Using the Past Knowledge to Improve Sentiment Classification

Qi Qin[1,2] **, Wenpeng Hu**[3]**, Bing Liu**[2,*]
[1] Center for Data Science, AAIS, Peking University
[2] Wangxuan Institute of Computer Technology, Peking University
[3] Department of Information Science, Peking University
{qinqi, wenpeng.hu}@pku.edu.cn, dcsliub@gmail.com

Abstract

This paper studies sentiment classification in the lifelong learning setting that incrementally learns a sequence of sentiment classification tasks. It proposes a new lifelong learning model (called L2PG) that can retain and selectively transfer the knowledge learned in the past to help learn the new task. A key innovation of this proposed model is a novel parameter-gate (p-gate) mechanism that regulates the flow or transfer of the previously learned knowledge to the new task. Specifically, it can selectively use the network parameters (which represent the retained knowledge gained from the previous tasks) to assist the learning of the new task t. Knowledge distillation is also employed in the process to preserve the past knowledge by approximating the network output at the state when task $t-1$ was learned. Experimental results show that L2PG outperforms strong baselines, including even multiple task learning.

1 Introduction

A typical sentiment analysis (SA) or social media company that provides sentiment analysis services has to work for a large number of clients (Liu, 2012). Each client normally wants to study people's opinions about a particular category of products or services, which we also call a *domain*. If we regard each such study/project as a task, we can model a SA company's working on a large number of studies/projects for clients as performing a sequence of SA tasks. A natural question that one would ask is whether after analyzing opinions about a number of products or services (tasks), the SA system of the company can do better on a new task by retaining the knowledge learned from the past/previous tasks and selectively transfer the prior knowledge to the new task to help it learn better. The answer should be yes because words and phrases used to express opinions or sentiments in different domains are similar and thus can mostly be shared or transferred across domains, although different domains do have domain specific sentiment expressions. This is a *lifelong learning* setting (Thrun, 1998; Silver et al., 2013; Chen and Liu, 2016). This paper focuses on *lifelong sentiment classification* (Chen et al., 2015).

Problem Definition: We consider incrementally learning a sequence of supervised sentiment classification (SC) tasks, $1, ..., t, ...$ Each task t has a training dataset $D^t_{train} = \{x^t_i, y^t_i\}^{n_t}_{i=1}$, where x^t_i is an input instance and y^t_i is its label, and n_t is the number of training examples of the tth task. Our goal is to design a lifelong learning algorithm $f(\cdot; \theta^t)$ or neural network that can retain the knowledge learned in the past and selectively transfer the knowledge to improve the learning of each new task t. It is assumed that after each task is learned, its training data is deleted and thus not available to help learn any subsequent tasks. This is a common scenario in practice because clients usually want to ensure the confidentiality of their data and don't want their data shared or used by others.

This problem is clearly related a *continual learning* (CL) (Chen and Liu, 2018; Parisi et al., 2019; Li and Hoiem, 2017; Wu et al., 2018; Schwarz et al., 2018; Hu et al., 2019; Ahn et al., 2019), which also aims to learn a sequence of tasks incrementally. However, the main objective of the current CL techniques is to solve the *catastrophic forgetting* (CF) problem (McCloskey and Cohen, 1989). That is, in learning each new task, the network parameters need to be modified in order to learn the new task. However, this modification can result in accuracy degradation for the previously learned tasks. In the problem defined above, our goal is

*Corresponding Author. This work was done when this author was on leave at Peking University. His current affiliation is University of Illinois at Chicago: liub@uic.edu.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1124–1133
November 16 - 20, 2020.

to forward transfer the past knowledge to improve the new task learning. We don't need to ensure the classifiers or models learned for previous tasks still work well.[1] However, as we will see in the experiment section, the proposed method is able to outperform the current state-of-the-art CL algorithms. Although there is some existing work on lifelong sentiment classification (Chen et al., 2015; Wang et al., 2019) based on naive Bayes. Our deep learning model is based on an entirely different approach and it performs markedly better.

To solve the proposed lifelong sentiment classification problem using a single neural network, two objectives have to be achieved. The first objective is to selectively transfer some pieces of knowledge learned in the past to assist the new task learning. Knowledge selection is critical here because not every piece of the past knowledge is useful (some even harmful) to the new task. The second objective is to preserve the knowledge learned in the past during learning the new task because if many pieces of previous knowledge are corrupted due to updates made in learning a new task, future tasks will not be able to benefit from them.

This paper proposes a novel model, called L2PG (*Lifelong Learning with Parameter-Gates*), to achieve the objectives. To achieve the first objective, we propose a novel mechanism called the **parameter-gate** (**p-gate**) to give suitable importance values to the network parameters representing the past knowledge according to how useful they are to the new task and transfer them to the new task to enable it to learn better. We split the parameters θ^t of the proposed model $f(\cdot;\theta^t)$ into three subsets: **(1)** the shared parameters $\theta^{s,t}$, **(2)** the task classification parameters $\theta^{c,t}$ and **(3)** the p-gate parameters, where the shared parameters $\theta^{s,t}$ and p-gate parameters are continuously updated with the learning of each new task t. $\theta^{c,t}$ remains unchanged for task t once the task is learned/trained. In learning a new task t, we only randomly initialize the task classification parameters $\theta^{c,t}$, and use an **input p-gate** to select parameters (or knowledge) from the shared parameters $\theta^{s,t-1}$ of the network state after learning task $t-1$ that are helpful to the new task t and use a **block p-gate** to block part of the previous training step parameters of $\theta^{s,t}$ that are not useful (or harmful) to task t.

To achieve the second objective, knowledge distillation (Hinton et al., 2015) is used to ensure that the updated network can preserve the previous model's knowledge in learning the new task.

This paper makes three main contributions:

- It proposes a novel deep learning model L2PG that uses a novel p-gate mechanism and knowledge distillation for lifelong sentiment classification. To the best of our knowledge, this approach has not been reported in the existing lifelong or continual learning literature.
- Unlike traditional gates that regulate the feature information flow through the sequence chain, the goal of the proposed p-gates is to select useful parameters (which represent the learned knowledge from previous tasks) to be transferred to the new task to make it learn better. In other words, p-gates regulate the knowledge transfer from the past to the present.
- It creates a 3-class sentence level sentiment classification corpus from reviews of 10 diverse product categories for lifelong learning evaluation. Such evaluations need many tasks. To our knowledge, no existing sentence sentiment classification corpus fits this need.

Experimental results show that L2PG outperforms state-of-the-art baselines including multi-task learning, which optimizes all the tasks at the same time.

2 Related Work

Our work is related to sentiment classification (Liu, 2012), lifelong learning and continual learning. For sentiment classification, recent deep learning models have been shown to outperform traditional methods (Kim, 2014; Devlin et al., 2018; Shen et al., 2018; Zhang et al., 2019; Qin et al., 2020). However, these models don't retain or transfer the knowledge to new tasks.

Lifelong learning: Most relevant to our work is lifelong learning (Thrun, 1998; Silver et al., 2013; Ruvolo and Eaton, 2013; Chen and Liu, 2014, 2016). For lifelong sentiment classification, Chen et al. (2015) used naive Bayes to leverage word probabilities under different classes in old tasks/domains as priors to help optimize the new task learning. Wang et al. (2019) worked similarly but their method can improve the model of a previous task without retraining. Xia et al. (2017) proposed a voting method but their method works on the same data from different time periods. Lv et al. (2019) proposed a model using two networks, one

[1] *Lifelong learning* and *continual learning* are often regarded as the same. Here, we follow (Thrun, 1998) and make this distinction.

for knowledge retention and one for feature learning. But it was shown to be weaker than (Wang et al., 2019). L2PG has a very different approach and performs markedly better. Wang et al. (2018) studied aspect level sentiment classification, which is not the goal of L2PG. However, to the best of our knowledge, none of these methods used gated mechanisms to regulate the transfer of knowledge in the lifelong learning process.

Continual learning: It is similar to lifelong learning, but its main goal is to overcome *catastrophic forgetting* to ensure learning of a new task will not forget the models learned for previous tasks (McCloskey and Cohen, 1989; Goodfellow et al., 2013). For example, LWF (Li and Hoiem, 2017) uses knowledge distillation loss to ensure that after learning a new task, it can still approximate the performance of the old tasks. EWC (Kirkpatrick et al., 2017) introduces constraints to control parameter changes when learning a new task. HAT (Serrà et al., 2018) masks units that are important to previous tasks by a hard attention. PGMA (Hu et al., 2019) generates a subset of parameters. Two reviews of continual learning can be found in (Chen and Liu, 2018; Parisi et al., 2019). Our lifelong learning setting focuses on transferring the past knowledge to the current task. We don't ensure that the models learned in the past still work well after learning a new task. Although Progressive Networks (Rusu et al., 2016) also tries to help future learning through knowledge transfer, but it is not scalable as its network size scales quadratically in the number of tasks.

Knowledge Distillation Loss was proposed in (Hinton et al., 2015) for transferring knowledge in a large model to a smaller one. LWF uses knowledge distillation to help deal with forgetting. Dhar et al. (2019) proposed an information preserving penalty, attention distillation loss, to preserve the information about existing classes. This setting is different from ours as it incrementally learns more classes. Each of our tasks is an independent sentiment classification problem with multiple classes.

3 The Proposed L2PG Model

The working of the proposed model L2PG in learning the new task t is illustrated in Figure 1. Our learner $f(\cdot;\theta^t)$ consists of three modules and two loss functions. The first module is the *shared knowledge module* **(SK)**, which consists of a CNN (*i.e.*, convolutional neural network) with various filters. It contains the shared knowledge across tasks in its parameters $\theta^{s,t}$. The second module is the *task classification module* **(TC)** with parameters $\theta^{c,t}$, which is a fully connected layer for the classification of task t. There is one TC for each task and it is fixed once t is learned. The third module is the *p-gate module* **(PG)**.

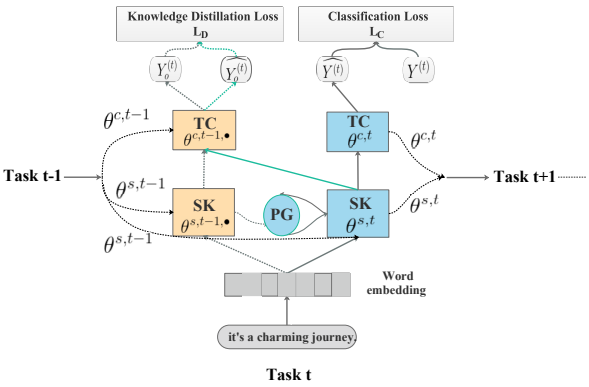

Figure 1: The proposed L2PG model. In learning task t, the parameters in the yellow boxes are temporary copies of the parameters of task $t-1$ (a superscript $\bullet$ is used to indicate a copy) and are not changed (they are deleted after learning task t). The parameters in the blue boxes and blue disk are updated. Green lines are for knowledge distillation.

In learning each new task t, a temporary copy of SK and of TC (in the yellow boxes of Figure 1) are made from the state of the network after task $t-1$ was learned. For clarity, we use the superscript $\bullet$ to indicate a copy of something. For example, $\theta^{s,t-1,\bullet}$ and $\theta^{c,t-1,\bullet}$ denote the copies of $\theta^{s,t-1}$ and $\theta^{c,t-1}$ respectively. They are fixed and not updated during the learning of task t. SK (in the blue box) and PG (in the blue disk) are updated in learning task t, and are also used in testing. The goal of PG is to identify useful knowledge for task t from the parameters $\theta^{s,t-1,\bullet}$ of SK after task $t-1$ training and to block the unhelpful or harmful knowledge in SK (see Sec. 3.3) for the current task. Knowledge distillation is used to ensure that in learning task t, the knowledge gained from the previous tasks are not forgotten. Updating the parameters of SK, TC and PG are done through back propagation. The two loss functions used are knowledge distillation loss and cross entropy loss.

3.1 Shared Knowledge Module (SK)

Let the training data of task t be D^t_{train}, and an instance of it with length L (after padding or cutting) be x^t_i with label y^t_i. Training of SK (in the blue box of Figure 1) for the new task t starts with SK of

the task $t-1$ model $f(\cdot;\theta^{t-1})$. After training of task t, $f(\cdot;\theta^{t-1})$ becomes SK of the model $f(\cdot;\theta^{t})$ for task t. During training, the input instance goes through SK to get advanced features to be used by task t's TC module. Let $V_{ij}^t \in \mathbf{R}^k$ be the word vector corresponding to the jth word of x_i^t and $X_i^t \in \mathbf{R}^{L\times k}$ be the embedding matrix of x_i^t. SK receives X_i^t from the input layer, and then extracts advanced features $\mathbf{C}_i^t$ in the form of a n-gram, i.e.,

$$\mathbf{C}_i^t = [c_1, c_2, ..., c_{L-n+1}] = [c_j]_{j=1}^{L-n+1} \quad (1)$$

where c_j represents the output produced by CNN's filter on $X_i^t[j : j+n-1, :]$. Mathematically, a convolution operation consists of a filter $W^t \in \mathbf{R}^{n\times k}$ and a bias $b^t \in \mathbf{R}$. c_j can be expressed as:

$$c_j = g(W^t \cdot X_i^t[j : j+n-1, :] + b^t) \quad (2)$$

where g is a nonlinear activation function such as **Relu**. We use a Maxpooling operation over the feature map and take the maximum value $\mathbf{C}_i^t = \max\{\mathbf{C}_i^t\}$ as the feature corresponding to this particular filter. The shared knowledge from SK of the current task t is

$$\mathbf{C}_i^t = \text{SK}(X_i; \theta^{s,t}) \quad (3)$$

where $\theta^{s,t}$ is the whole set of parameters of SK of the current task t.

3.2 Task Classification Module (TC)

Using Eq. 3 we obtain a high-level representation of the input instance x_i^t. Then, we pass the feature of x_i^t through TC of the task t to obtain the classification result,

$$\widehat{y_i^t} = \text{Softmax}(\mathbf{C}_i^t \cdot W_c^t + b_c^t) \quad (4)$$

where W_c^t, b_c^t are the weight and bias of the classifier. Like SK above, we refer the classifier from the TC module of the current task t as

$$\widehat{y_i^t} = \text{TC}(\mathbf{C}_i^t; \theta^{c,t}) \quad (5)$$

where $\theta^{c,t}$ is the set of all parameters of the TC (classifier) of the current task t. As mentioned earlier, TC is a fully connected layer (in the top blue box of Figure 1) and is randomly initialized.

3.3 P-Gate Module (PG)

Recall that in learning the new task t, the proposed p-gate mechanism (PG) selectively transfers some pieces of knowledge from the parameters $\theta^{s,t-1}$ after task $t-1$ is learned, i.e., $f(\cdot;\theta^{t-1})$, to the current task t. At the same time, PG also needs to block the knowledge that is not helpful to the current task or knowledge that may cause forgetting for previous tasks. We achieve the goals using two p-gates, an **input p-gate** and a **block p-gate**.

The **input p-gate** uses the Sigmoid function to determine what proportion of each parameter in the SK from the previous task should help the current task to learn. The input p-gate is formulated as,

$$z = \text{Sigmoid}(W_z \cdot \theta^{s,t-1,\bullet}) \quad (6)$$

where $\theta^{s,t-1,\bullet}$ is a copy of $\theta^{s,t-1}$, the parameters of the network state after task $t-1$ was learned (see the top yellow box in Figure 1), and W_z is the set of trainable input p-gate's parameters. $\theta^{s,t-1,\bullet}$ does not change during training. $z_{ij} \to 1$ means that the corresponding parameter is almost completely helpful to the learning of the current task, and $z_{ij} \to 0$ means that the parameter is of no help (or harmful) to the current task t.

The **block p-gate** blocks some SK's parameters from the previous training step $S-1$ in the training process of the current task t. $\theta_{S-1}^{s,t}$ serves as the initial parameters of $\theta_S^{s,t}$ of the current training step S. The block p-gate is formulated as,

$$b = \text{Sigmoid}(W_b \cdot \theta_S^{s,t}) \quad (7)$$

where W_b is the set of trainable block p-gate's parameters. $b_{ij} \to 0$ means that the current parameter almost certainly has a negative effect on the next learning or may lead to forgetting. Both the input p-gate's parameters W_z and block p-gate's parameters W_b are trained by minimizing the loss function of the current task t's classification module TC.

After this step of training using a batch of examples for task t is completed, SK's parameters of step S is revised by the following combination operation, i.e., the trained $\theta_S^{s,t}$ is replaced,

$$\theta_S^{s,t} := z * \theta^{s,t-1,\bullet} + b * \theta_S^{s,t} \quad (8)$$

This operation is to reduce the interference of the new task t on the existing knowledge learned in the past and cause forgetting.

After the parameter combination and revision is done, the training goes to the next step/iteration $S+1$ using another batch of data. Note that this combination and replacement operation is not used if S is the last step of an epoch.

3.4 Objective of Optimization

In order for the model to retain old knowledge during the learning process, we use the *knowledge distillation loss* in (Hinton et al., 2015) to encourage the outputs of one network to approximate the outputs of another, similar to LWF. Therefore, when we start training task t, we first use $f(\cdot;\theta^{t-1})$ to get the softmax output $Y_o^t = \{y_{oi}^t\}_{i=1}^{n_t}$ of all training instances of t and $\widehat{Y_o^t} = \{\widehat{y_{oi}^t}\}_{i=1}^{n_t}$ is the softmax outputs of SK of task t combining TC of task $t-1$, which are used to build a knowledge distillation loss. Let $Y^t = \{y_i^t\}_{i=1}^{n_t}$ be all ground truth labels of task t and $\widehat{Y^t} = \{\widehat{y_i^t}\}_{i=1}^{n_t}$ be the softmax outputs of $f(\cdot;\theta^t)$ used to build the cross entropy loss. n_t is the number of training examples of task t.

We now present the L2PG's optimization goals when sequentially learning each new task t.

Knowledge Distillation Loss: It is defined as:

$$L_D(Y_o^t, \widehat{Y_o^t}) = -\sum_{i=1}^{n_t} y_{oi}'^t \cdot log(\widehat{y_{oi}'^t}) \quad (9)$$

$$y_{oi}'^t = \frac{(y_{oi}^t)^{1/K}}{\sum_j (y_{oj}^t)^{1/K}}, \widehat{y_{oi}'^t} = \frac{(\widehat{y_{oi}^t})^{1/K}}{\sum_j (\widehat{y_{oj}^t})^{1/K}}. \quad (10)$$

where K is a hyperparameter and Hinton et al. (2015) suggests $K > 1$, which increases the weight of smaller logit values and encourages the network to better encode similarities among classes.

Classification Loss: The classification loss of the current learner $f(\cdot;\theta^t)$ for task t is cross entropy of $\widehat{Y^t}$ and Y^t,

$$L_C(Y^t, \widehat{Y^t}) = -\sum_{i=1}^{n_t} y_i^t \cdot log(\widehat{y_i^t}) \quad (11)$$

So, the total loss is

$$L = L_C(Y^t, \widehat{Y^t}) + \lambda L_D(Y_o^t, \widehat{Y_o^t}) + \beta R(\theta^t) \quad (12)$$

where λ and β are hyperparameters, $R(\theta^t)$ is the regularization term (we use L2 regularizer), and θ^t includes $\theta^{c,t}, \theta^{s,t}, W_b$ and W_z.

The algorithm of L2PG for training the new task t is given in Algorithm 1, which is self-explanatory.

Algorithm 1 L2PG - Learning the new task t

Input: Training set D_t^{train} of task t, and shared parameters $\theta^{s,t-1,\bullet}$ and task classification parameters $\theta^{c,t-1,\bullet}$ from task $t-1$.
Initialize:
  $\theta_0^{s,t} \leftarrow \theta^{s,t-1}$ // 0 denotes training step
  $\theta_0^{c,t} \leftarrow \mathbf{Random}(|\theta^{c,t}|)$
for each training step $S = 0, 1, ..., M$ **do**
  Sample one batch X_S^t from D_t^{train};
  // compute outputs for loss L_D
  $Y_o^t = f(X_S^t; \theta^{s,t-1,\bullet}, \theta^{c,t-1,\bullet})$;
  $\widehat{Y_o^t} = f(X_S^t; \theta_S^{s,t}, \theta^{c,t-1,\bullet})$;
  // compute output for loss L_C
  $\widehat{Y^t} = f(X_S^t; \theta_S^{s,t}, \theta_S^{c,t})$;
  Update parameters:
  Parameters $\theta_S^{s,t}$, $\theta_S^{c,t}$, W_z and W_b are updated by minimize Eq. 12;
  // Use the trained p-gate parameters to
  // select the knowledge for the next step
  $z = \text{Sigmoid}(W_z \cdot \theta^{s,t-1,\bullet})$;
  $b = \text{Sigmoid}(W_b \cdot \theta_S^{s,t})$;
  $\theta_S^{s,t} := z * \theta^{s,t-1,\bullet} + b * \theta_S^{s,t}$
end for

4 Experiments

We now evaluate L2PG and compare it with two main types of baselines, i.e., those under lifelong sentiment classification and those under continual learning for dealing with catastrophic forgetting.

4.1 Datasets

We carried out experiments on two datasets. The first dataset is for document level sentiment classification with two classes, positive and negative. It consists of reviews of 16 diverse kinds of products (domains) commonly used in multi-task text classification (Liu et al., 2017). The reviews of the first 14 products are from Amazon.com. The remaining two are about movie reviews (IMDB and MR). The number of training and testing samples for each product (or task) is about 1,400 and 400, respectively. We call this dataset **Mix-16**, which gives us 16 tasks, one per product category/domain.

The second dataset is for sentence-level sentiment classification and is created by us. It consists of review sentences of 10 types of products/domains crawled from Amazon.com, which gives us 10 tasks. Each sentence is labeled with *positive*, *negative* or *neutral*. The sentences with conflict opinions (e.g., both positive and negative) are not used. Sentence sentiment classification of each domain forms a task. The review sentences of each product are annotated by two annotators independently. We trained all the annotators and provided them with an annotation instruction document. After training, each of them was asked to

Dataset	$Avg.L$	$Train$	$Test$	$\|V\|$
Air conditioner	15	1,018	439	2,714
Diaper	17	1,065	459	2,685
Stove	15	1,084	467	2,813
Headphone	15	1,186	510	3,476
Bike	16	1,021	441	3,097
Luggage	17	1,211	520	3,380
Smartphone	16	1,187	511	3,778
GPS	17	1,318	567	3,976
TV	16	1,346	579	4,053
Hotel	16	1,466	630	4,015

Table 1: Dataset statistics of Amazon-10. $Avg.l$: Average sentence length. $Train$, $Test$: number of training and test sentences respectively. $|V|$: Vocabulary size.

perform annotation of 50 sentences to assess their annotation quality. They started their annotation only after we were satisfied with their annotations. After they completed their annotations, sentences with disagreements were identified and discussed by the annotators to come to an agreement. The Kappa score for annotator agreement was 0.7947.

Note that we are aware that there are some existing sentence sentiment classification data, but each of them is only from reviews of a single product. We are unable to create many different domain tasks from them to suit lifelong learning. Furthermore, they mostly have only two classes, positive and negative, which do not reflect all review sentences because many review sentences express no sentiment (neutral), e.g., "*I bought this camera yesterday*." That is why we created the new dataset with 10 different categories of products, which give us 10 tasks for lifelong learning.[2] We denote this dataset as **Amazon-10**.

4.2 Baselines

We consider the following baselines for comparison with the proposed L2PG model. The feature extraction module (*e.g.*, SK of L2PG) of all models including L2PG uses CNN (Kim, 2014) and each classifier is a fully connected layer (e.g., TC of L2PG for each task).

I-CNN: I-CNN is a single-task CNN classifier, where one CNN model performs each task independently, no sharing of knowledge across tasks.

S-CNN: S-CNN is I-CNN but uses one CNN model (one feature extractor and one classifier) to incrementally learn all tasks. No mechanism is used to deal with knowledge transfer or forgetting.

LWF-T: This is a continual learning model based on *Learning without forgetting* (LWF) (Li and Hoiem, 2017). It uses knowledge distillation to deal with catastrophic forgetting. Since LWF was originally designed for image classification, we modified it for text classification using the same model as the above, i.e., CNN for the shared parameter module, one fully connected layer for each task's classifier (each task has its own classifier). When training the new task, the parameters of the task-specific classifiers of the previous tasks are fixed. We denote this LWF model as **LWF-T**.

HAT: This is a well-known algorithm for continual learning that deals with catastrophic forgetting (Serrà et al., 2018). Since HAT (or UCL below) was also designed for image classification, we again adapted it for text. HAT has almost no forgetting for image classification.

UCL: This is a latest continual learning model (Ahn et al., 2019) that improves HAT.

LSC: This is the naive Bayes-based lifelong sentiment classification model in (Chen et al., 2015).

LNB: LNB (Wang et al., 2019) is similar to LSC but is able to improve the model of a previous task without retraining. The system in (Lv et al., 2019) is not compared as it performed poorer than LNB (Wang et al., 2019).

MTL: This is a multi-task learning baseline using CNN as the shared knowledge module as L2PG and each task has its own task-specific classifier like L2PG, HAT, and UCL.[3] In (Li and Hoiem, 2017), MTL's performance was regarded as the upper bound of continual learning because the training data of all tasks are available during training. But for L2PG, after each sentiment classification task is learned, its data is assumed deleted.

Training details. For all models in our experiments, the word embedding are randomly initialized as 300-dimension vectors and then modified during training. We use filter sizes of [3,4,5] with 100 feature maps each in the CNN module, and dropout rate of 0.5. In L2PG, we set mini-batch size to 50, learning rate to 0.001, temperature $T = 2$ and $\lambda, \beta = 1$. We use the same feature extractor CNN and classifier as other models. For HAT and UCL, we modified their code for text and optimized their parameters (their original parameters performed poorly for text), but we did not change their algorithms. HAT and UCL need 300

[2] Our code and the newly created dataset can be found from https://github.com/Qqinmaster/L2PG

[3] Note that we use a comparable architecture for MTL to other baseline models for fair comparisons. It is not the state-of-the-art model reported in the literature, which uses more sophisticated architectures and achieves better results.

	I-CNN	S-CNN	LWF-T	HAT	UCL	LSC	LNB	MTL	L2PG
Health	84.05 (± 0.99)	86.40 (± 0.58)	86.00 (± 0.50)	84.50 (± 1.53)	87.00 (± 0.61)	87.50	88.25	86.80 (± 0.69)	88.45 (± 0.69)
Toys	84.30 (± 0.97)	86.40 (± 0.52)	86.95 (± 0.48)	84.75 (± 0.52)	86.55 (± 0.70)	86.25	89.50	85.70 (±0.21)	88.00 (± 0.88)
Electronics	82.60 (± 0.55)	85.05 (± 0.45)	86.85 (± 1.50)	82.75 (± 2.16)	85.69 (± 0.82)	82.75	78.50	87.35 (±0.38)	88.15 (± 1.17)
Books	79.45 (± 0.48)	82.60 (± 0.91)	81.10 (± 0.91)	77.75 (± 2.43)	81.20 (± 1.36)	78.75	79.00	80.15 (±0.58)	84.05 (± 0.78)
Music	77.60 (± 0.45)	80.90 (± 1.19)	80.85 (± 1.05)	76.50 (± 2.46)	81.50 (± 0.89)	80.00	79.25	80.50 (±0.66)	82.25 (± 1.00)
Baby	84.90 (± 0.99)	86.20 (± 0.72)	86.40 (± 0.72)	85.35 (± 2.27)	86.45 (± 1.10)	82.00	83.75	86.25 (±0.47)	88.75 (± 0.47)
Magazines	90.70 (± 0.62)	90.45 (± 0.76)	92.20 (± 0.37)	89.75 (± 1.26)	91.14 (± 0.52)	92.75	88.25	91.70 (±0.67)	92.20 (± 0.48)
MR	66.00 (± 1.45)	68.50 (± 0.98)	70.25 (± 0.68)	58.70 (± 1.18)	68.28 (± 0.88)	70.00	71.50	70.60 (±0.91)	71.35 (± 1.10)
Sports	84.00 (± 0.35)	87.85 (± 0.63)	85.30 (± 0.93)	86.00 (± 1.28)	86.80 (± 0.82)	87.00	86.00	88.40 (±0.89)	87.20 (± 1.30)
Kitchen	83.45 (± 0.57)	86.90 (± 0.49)	86.30 (± 0.54)	81.50 (± 1.84)	86.30 (± 0.52)	85.00	85.25	88.60 (±0.34)	89.30 (± 0.41)
Apparel	85.75 (± 0.35)	86.95 (± 0.89)	86.55 (± 0.82)	84.00 (± 1.23)	85.47 (± 0.65)	84.75	86.25	87.20 (±0.37)	86.90 (± 0.76)
IMDB	74.45 (± 1.07)	76.20 (± 0.51)	77.35 (± 0.89)	71.75 (± 1.25)	78.60 (± 1.20)	80.19	79.95	76.35 (±1.36)	80.65 (± 0.93)
Software	85.35 (± 1.42)	87.40 (± 0.49)	87.15 (± 0.84)	81.50 (± 1.25)	86.55 (± 0.76)	87.00	83.75	87.40 (±0.49)	88.15 (± 1.17)
Vido	80.00 (± 0.71)	84.00 (± 0.47)	83.70 (± 0.67)	81.00 (± 1.01)	83.35 (± 1.29)	81.75	81.50	87.40 (±0.49)	85.95 (± 0.60)
Camera	87.45 (± 1.28)	87.80 (± 0.72)	88.15 (± 0.68)	84.15 (± 0.25)	84.59 (± 0.51)	85.50	86.50	87.40 (±0.49)	88.60 (± 0.88)
DVD	78.20 (± 1.10)	79.95 (± 0.91)	80.00 (± 0.79	77.35 (± 0.75	79.85 (± 0.46)	80.75	81.00	81.05 (±1.02)	82.45 (± 0.33)
Average	81.77 (± 0.83)	83.97 (± 0.70)	84.07 (± 0.77)	80.46 (± 1.42)	83.71 (± 0.82))	83.25	83.01	84.56 (±0.63)	85.78 (± 0.81)

Table 2: Mix-16: Average accuracy (%) of each task (or domain) over 5 different task sequences for every candidate model under the lifelong learning setting. LSC and LNB don't have ±sd as they are task sequence independent.

	I-CNN	S-CNN	LWF-T	HAT	UCL	MTL	L2PG
Bike	64.44(± 0.79)	65.47 (± 1.04)	65.88 (± 1.07)	62.17 (± 4.42)	65.85 (± 0.81)	66.32 (± 0.79)	67.48 (± 0.47)
GPS	60.98 (±0.47)	66.03 (±1.72)	67.13 (±1.11)	60.49 (±3.45)	66.08 (±0.93)	64.93 (±1.50)	68.78 (±0.75)
Hotel	65.01 (± 0.71)	64.50 (± 1.03)	66.05 (± 1.15)	60.28 (±1.55)	66.44 (±0.88)	64.66 (±0.61)	68.73 (±1.48)
Luggage	69.23 (±0.35)	73.36 (±0.64)	73.42 (± 0.25)	70.16 (±1.23)	73.41 (±0.37)	73.22 (±0.51)	76.58 (±0.71)
Diaper	63.83 (±0.84)	65.94 (±1.22)	66.33 (±1.49)	62.77 (±1.35)	64.74 (±0.51)	66.12 (±0.88)	68.05 (±1.26)
Smartphone	60.61 (±1.18)	66.76 (±0.55)	67.73 (±0.93)	60.43 (±3.63)	65.63 (±1.16)	66.10 (±1.28)	69.74 (±0.35)
Stove	67.23 (±0.94)	68.28 (±0.64)	69.89 (±1.05)	67.19 (±2.05)	68.24 (±0.43)	69.92 (±0.67)	70.67 (±1.28)
Headphone	62.74 (±0.62)	65.17 (±1.21)	65.61(± 0.95)	61.36 (±2.41)	65.90 (±0.68)	64.18 (± 1.14)	68.18 (±1.08)
TV	61.27 (±0.46)	64.43 (±0.36)	65.34 (±1.55)	61.18 (±1.37)	64.58 (±0.42)	64.18 (±0.65)	66.70 (±1.24)
Air-condition	61.63 (±0.67)	65.77 (±0.85)	66.22 (±0.79)	63.87 (±2.21)	67.10 (±1.27)	65.10 (±1.24)	69.66 (±1.02)
Average	63.70 (±0.71)	66.57 (±0.93)	67.36 (±1.04)	62.98 (±2.37)	66.80 (±0.75)	66.47 (±0.93)	69.46 (±0.97)

Table 3: Amazon-10: Average accuracy (%) of each task (or domain) over 5 different task sequences for every candidate model under the lifelong learning setting. LSC and LNB are not used here because their algorithms cannot handle more than 2 classes in a task.

and 100 epochs to achieve the best results respectively, but for others, 20 epochs are sufficient. For LSC and LNB, we use their original code. Note that LSC and LNB can only deal with two-class sentiment classification due to the limitation of its knowledge sharing mechanism. Thus we cannot run it on the second dataset which has three classes.

4.3 Results and Analysis

For our lifelong learning setting, we use 5 random task sequences to compute the accuracy as different task sequences may give different results.[4] For each sequence, each task (also a domain) is used as the last task in turn to collect its test result. This is because we are only interested in improving the accuracy of the current/new task based on knowledge learned in the previous tasks. Table 2 and Table 3 give the mean accuracy of each task when it is the last task for Mix-16 and Amazon-10 respectively. The average accuracy of each column is given in the last row of each table. L2PG significantly outperforms every baseline on both datasets with p-value < 0.01 on paired t-test. Compared with I-CNN, L2PG increases the averaged accuracy by 4.01% on Mix-16 and 5.76% on Amazon-10. This is because I-CNN treats each task independently, but L2PG performs knowledge transfer. Even naive single continual learning of S-CNN outperforms I-CNN by 2.20%, 2.87% on the two datasets respectively. This shows that significant knowledge sharing exists in sentiment classification tasks.

Compare with Lifelong Sentiment Classification Models: LSC and LNB are only designed for 2-class lifelong sentiment classification. They cannot handle three classes in Amazon-10 and thus have no result for it. L2PG is only compared to LSC and LNB on Mix-16. In Table 2, we see that L2PG outperforms LSC and LNB by 2.53%, 2.77% respectively. One reason is that LSC and LNB are naive Bayes approaches, which cannot model the contextual relationship due to its conditional independence assumption on features (words). L2PG does not have this limitation.

Compare with Continual Learning Models: For continual learning models LWF-T, HAT and UCL, to be consistent with the lifelong setting of L2PG, we also take turns to put each task as the last and use the final model to get the accuracy of the last task (there is no forgetting for the last task). The average accuracy of L2PG on both datasets is

[4]Because LSC and LNB pairs are naive bayes based methods, they will not be affected by task order under lifelong learning setting.

System	Average
L2PG	69.46 ($\pm$0.97)
w/o L_D (L2GP-NK)	68.31 ($\pm$0.64)
w/o PG (LWF-T)	67.36 ($\pm$1.04)
w/o L_D or PG (S-CNN)	66.57 ($\pm$0.93)

Table 4: Ablation experiments on Amazon-10. For each system, the result is the average of all tasks' accuracy in the lifelong learning setting, where L_D is the knowledge distillation loss.

markedly higher than these models. For example, on Amazon-10, L2PG's average accuracy is 2.20% higher than LWF-T, 6.48% higher than HAT and 2.66% higher than UCL. As we can see, continual learning models LWF-T and UCL (the latest algorithm) that only deals with catastrophic forgetting also achieve better results than I-CNN as the tasks are similar and share a great deal of knowledge (HAT is markedly worse). However, since they do not have specific mechanisms to perform knowledge transfer, they are weaker than L2PG.

Compare with MTL: Under the condition that the same CNN is used as the feature extractor and a fully connected layer is used as a task-specific classifier for each task, L2PG is on average 1.22% better than MTL on Mix-16 and 2.99% better than MTL on Amazon-10. MTL is often considered the upper bound of continual learning because it trains all the tasks together. However, its loss is the sum of the losses of all tasks, which does not mean it optimizes for every individual task. L2PG in the lifelong learning setting tries to do the best for the new/current task.

Ablation Experiments and Analysis: To show the usefulness of each component of L2PG, we perform ablation experiments on the Amazon-10 data without using knowledge distillation loss, the p-gate modele (PG), or both. Their results are given in Table 4.

When only removing the knowledge distillation loss from L2PG (w/o L_D), which we call L2PG-NK, the average accuracy drops by about 1.15%, which indicates that using knowledge distillation loss to actively preserve the old knowledge is useful. When only removing the p-gate module from L2PG (w/o *PG*), which is actually LWF-T, the average accuracy drops by about 2.10%, which shows that our PG mechanism can choose and transfer the right knowledge to the new task. Without both knowledge distillation loss and PG (w/o L_D or PG), which is actually S-CNN, the result is much worse. Comparing L2PG-NK with LWF-T and S-CNN, we can see L2PG-NK's average score is 0.95% higher than LWF-T, 1.74% higher than S-CNN, which indicates that even without distillation loss, the PG mechanism can effectively retain the past knowledge and use it effectively.

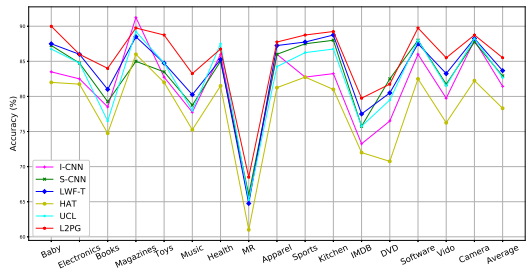

Figure 2: Mix-16: accuracy of its 16 tasks of continual learning. LSC, LNB and MTL are not used as they don't work in the continual learning setting.

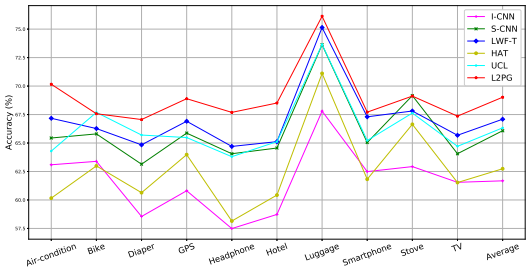

Figure 3: Amazon-10: accuracy of its 10 tasks of continual learning. LSC, LNB & MTL do not work in the continual learning setting.

4.4 L2PG in the Continual Learning Setting

Here we run L2PG as a continual learning system. Like LWF-T, HAT and UCL, after all tasks are learned, L2PG is tested on every task's test data (note, in the lifelong learning setting, we only test on the last task). The continual learning results on the two datasets are presented in Figures 2 and 3, where six models are compared, namely, I-CNN, S-CNN, LWF-T, HAT, UCL and L2PG. From the figures, we observe that L2PG actually can outperform all the other five models. This is due to the fact that L2PG encourages knowledge transfer, while the continual learning systems LWF-T, HAT and UCL only focus on preserving the past knowledge.

5 Conclusion

This paper proposed an effective model L2PG for lifelong sentiment classification. L2PG not only can retain what it has learned, but also selectively transfer the past knowledge to learn the new task

better. The key component is the proposed parameter gate (p-gate) mechanism that is able to select the right previously learned knowledge or parameters to transfer to the new task. Knowledge distillation is also employed to maintain the knowledge or models learned for the previous tasks. Empirical evaluation showed L2PG outperforms strong baselines in lifelong learning, continual learning, and even multi-task learning.

References

Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. 2019. Uncertainty-based continual learning with adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 4394–4404.

Zhiyuan Chen and Bing Liu. 2014. Topic modeling using topics from many domains, lifelong learning and big data. In *ICML*.

Zhiyuan Chen and Bing Liu. 2016. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 10(3):1–145.

Zhiyuan Chen and Bing Liu. 2018. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207.

Zhiyuan Chen, Nianzu Ma, and Bing Liu. 2015. Lifelong learning for sentiment classification. In *ACL-2015*, pages 750–756.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. 2019. Learning without memorizing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5138–5146.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. 2013. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao, Jinwen Ma, Dongyan Zhao, and Rui Yan. 2019. Overcoming catastrophic forgetting for continual learning via model adaptation. In *ICLR*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *Eprint Arxiv*.

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

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. *arXiv preprint arXiv:1704.05742*.

Guangyi Lv, Shuai Wang, Bing Liu, Enhong Chen, and Kun Zhang. 2019. Sentiment classification by leveraging the shared knowledge from a sequence of domains. In *DASFAA*, pages 795–811.

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

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. 2019. Continual Lifelong Learning with Neural Networks: A Review. *Neural Networks*, 113:54–71.

Qi Qin, Wenpeng Hu, and Bing Liu. 2020. Feature projection for improved text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

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

Paul Ruvolo and Eric Eaton. 2013. ELLA: An efficient lifelong learning algorithm. In *ICML*, pages 507–515.

Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. 2018. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*.

Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. 2018. Overcoming catastrophic forgetting with hard attention to the task. *arXiv preprint arXiv:1801.01423*.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin.

2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. *ACL-2018*.

Daniel L Silver, Qiang Yang, and Lianghao Li. 2013. Lifelong machine learning systems: Beyond learning algorithms. In *2013 AAAI spring symposium series*.

Sebastian Thrun. 1998. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer.

Hao Wang, Bing Liu, Shuai Wang, Nianzu Ma, and Yan Yang. 2019. Forward and backward knowledge transfer for sentiment classification. *arXiv preprint arXiv:1906.03506*.

Shuai Wang, Guangyi Lv, Sahisnu Mazumder, Geli Fei, and Bing Liu. 2018. Lifelong learning memory networks for aspect sentiment classification. In *IEEE BigData 2018*.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, Zhengyou Zhang, and Yun Fu. 2018. Incremental classifier learning with generative adversarial networks. *arXiv preprint arXiv:1802.00853*.

Rui Xia, Jie Jiang, and Huihui He. 2017. Distantly supervised lifelong learning for large-scale social media sentiment analysis. *IEEE Transactions on Affective Computing*, 8(4):480–491.

Kai Zhang, Hefu Zhang, Qi Liu, Hongke Zhao, Hengshu Zhu, and Enhong Chen. 2019. Interactive attention transfer network for cross-domain sentiment classification. *AAAI-2019*.