Class Incremental Learning for Intent Classification with Limited or No Old Data

Debjit Paul[*] EPFL debjit.paul@epfl.ch

Daniil Sorokin Amazon Alexa AI dsorokin@amazon.de

Judith Gaspers Amazon Alexa AI gaspers@amazon.de

Abstract

In this paper, we explore class-incremental learning for intent classification (IC) in a setting with limited old data available. IC is the task of mapping user utterances to their corresponding intents. Even though class-incremental learning without storing the old data yields high potential of reducing human and computational resources in industry NLP model releases, to the best of our knowledge, it hasn't been studied for NLP classification tasks in the literature before. In this work, we compare several contemporary class-incremental learning methods, i.e., BERT warm start, L2, Elastic Weight Consolidation, RecAdam and Knowledge Distillation within two realistic class-incremental learning scenarios: one where only the previous model is assumed to be available, but no data corresponding to old classes, and one in which limited unlabeled data for old classes is assumed to be available. Our results indicate that among the investigated continual learning methods, Knowledge Distillation worked best for our class-incremental learning tasks, and adding limited unlabeled data helps the model in both adaptability and stability.

1 Introduction

In real-world scenarios, NLP models are regularly updated to incorporate new functionality that either covers new data distributions or includes new output classes (Diethe et al., 2018). We focus on the latter scenario in this work, and we consider a feature expansion use case for a spoken language understanding (SLU) task in a voice assistant, such as Siri or Alexa. In particular, we focus on the task of intent classification (IC), which is a common sub-task in SLU that aims to map an input user utterance to an intent supported by the system (for instance, PLAYMUSIC for *play the best songs from Madonna*). With feature expansion, new intents are to be added to the SLU model over time to support new output classes that corresponds to new user-facing features in a voice assistant.

A regular feature expansion process in production results in a series of consecutive SLU model releases that are trained for the same intent classification task, in which new output classes are being added to the model. Having regular model releases, it is wasteful of computing and human resources to start from scratch every time and not re-use the previous release model in one form or another. Moreover, as our research community moves towards building NLP systems that are environmentally friendly (e.g., requiring less training time and computational resources), class incremental learning (C-IL) becomes an important research direction. Hence, the problem of class incremental learning (C-IL) has been studied by the community for a while now (Kirkpatrick et al., 2017); yet, most of the previous work has focused on computer vision tasks (Cheng et al., 2019) with limited attention to NLP (Cao et al., 2020; Thorne and Vlachos, 2021).

A C-IL setup for regular releases of a model offers both challenges and opportunities. On the one hand, updating the previous model with new classes introduces a stability-plasticity problem (Mermillod et al., 2013), where the new model should both retain the knowledge about the old classes and learn the new classes on the same level of accuracy. On the other hand, re-using the old model to build the next release might help to reduce the reliance on old training data and speed up the training of the next release (Ash and Adams, 2020).

In this work, we compare contemporary solutions for incremental learning on a C-IL problem for the IC task following a production-inspired feature expansion cycle. We assume a number of consecutive model releases, where each one adds a number of new output classes and the corresponding training data. The training data for the old classes is either limited or not available at all in

[*]Work done at Amazon Alexa AI

line with the privacy requirements of a real-life setup, where there might be restrictions on storing or using older data.

We simulate this setup with the large intent classification OOS (Out-of-scope) dataset (Larson et al., 2019). Furthermore, we restrict our experiments to methods that do not increase the capacity of the final model with time. This is an important restriction in a real-world scenario where runtime constraints do not allow to make the model larger.

Our main contributions are:

1. To best of our knowledge, we are the first to explore different incremental learning methods for class-incremental learning in SLU in a data-scarce scenario.

2. We focus on the restriction of a production-like setup, where previous models might be available, but not the data and the runtime restriction prohibit making the model larger with time.

3. We present in-depth experiments on C-IL in two data-scenarios: without old data being available and with limited old, but unlabelled data being available. In the experiments, we compare the techniques that were previously proposed for fine-tuning and task-incremental learning and apply them repeatedly to extend the same model with new classes.

2 Related Work

2.1 Spoken language understanding

SLU has been mostly approached with deep learning methods (Mesnil et al., 2013) and specifically with large pre-trained models (Zhang and Wang, 2016; Chen et al., 2019; Louvan and Magnini, 2020). To improve the overall SLU performance, the community has investigated semi-supervised learning and paraphrasing to bootstrap new features and to overcome the class imbalance problem (Cho et al., 2019; Sokolov and Filimonov, 2020). The most research on SLU assumes that the number of classes is static, while in a real production SLU system, new classes are added on a regular basis. In contrast, in this work, we propose to focus on a C-IL scenario, where new classes are added to the system and the models needs to adapted in the absence of the previous training data.

2.2 Class-incremental learning

In a production environment, C-IL is a challenging problem since normally the new classes are only a small fraction of the classes in the new data.

Most approaches for incremental learning have been developed in the context of task-incremental learning and computer vision problems. In this paper, we study the importance of such methods on C-IL in NLP. There are several approaches that use regularization terms together with the classification loss in order to mitigate catastrophic forgetting. Few methods concentrate on the weights and estimate an importance metric for each parameter in the network (Kirkpatrick et al., 2017; Thorne and Vlachos, 2021) to decide what to update, while others focus on preventing the activation drift (Li and Hoiem, 2018).

Many previous works on continual learning have also focused on learning from a continuous stream of data (Biesialska et al., 2020) or on an incremental learning of new tasks (Kanwatchara et al., 2021) and languages (Castellucci et al., 2021). Payan et al. (2021) discuss a single-task continual learning setup and simulated a passive data extension scenario where new examples are coming in for all output classes on a public dataset. Similarly, Ash and Adams (2020) evaluate a batch-learning setup, where each model iteration is warm-started from the previous step and the whole training data is always available, while some new data is added across all output classes in each batch.

Finally, Wu et al. (2022) experiment on 5 different tasks to investigate the behaviour of fine-tuned large pre-trained models when the number of output classes grows with time. This setup is close to ours, as we also look at an expanding set of output intents in models. Uniquely, we focus on data-scarce scenarios, where old data might not be available anymore, which is motivated by privacy and production requirements of voice assistants.

3 Challenges

The fundamental obstacles to effective C-IL are conceptually simple, but in practice very challenging to overcome. These challenges originate from the sequential training of tasks and the requirement that at any moment the learner must be able to classify all classes from all previously learned tasks. Incremental learning methods must balance retaining knowledge from previous tasks while learning new knowledge for the current task. This problem is called the stability-plasticity dilemma (Mermillod et al., 2013). A naive approach to class incremental learning which focuses solely on learning the new task will suffer from catastrophic forgetting: a drastic drop in the performance on previous tasks. Preventing catastrophic forgetting leads to a second important problem of class incremental learning, i.e., the problem of intransigence: the resistance to learn new tasks. Class incremental learning methods need to balance between keeping knowledge about old classes and learning new classes at the same level of accuracy.

4 Method

We consider a C-IL scenario in which an existing NLP classification model is updated over time. We assume a number of consecutive model updates, and for each update, a certain number of new classes and corresponding training data become available, which need to be supported. Since, to the best of our knowledge, C-IL has not yet been explored for NLP classification tasks, this study includes a diverse set of techniques, which have shown promising results on other incremental learning tasks. In the following, we first describe our C-IL scenario in more detail. Subsequently, we discuss the considered basic classification model architecture and the C-IL strategies afterwards.

4.1 Class-incremental learning scenario

We assume that a task-specific classification model M_0 is available, which may already cover a comparatively large number of different output classes. Furthermore, we assume that we have access to an input stream of datasets D, each comprising labeled data for new classes: $D = [D_1, \ldots, D_n]$ with $D_i = \{(x_{i,j}, y_{i,j})\}_{j=1}^{|D_i|}$, where $x_{i,j_1}, \ldots, x_{i,j_n}$ is an utterance with n tokens, and $y_{i,j}$ is a sentence-level intent label. Each $D_i \in D$ comprises labeled data belonging to k_i new classes, i.e., classes which have not been observed during training of M_0 or in any of the previous datasets $D_1, \ldots, D_{i-1}$. For each dataset D_i, we perform a model update to integrate the new classes starting from the previous model M_{i-1}, yielding model M_i. More specifically, we assume that for each model update M_i

1. only the previous model M_{i-1} is available,
2. the dataset D_i comprising data belonging to k_i new classes is available, and
3. the datasets from previous iterations $D_1, \ldots, D_{i-1}$ are not available anymore.

In some of our experiments, we additionally assume that unlabeled data for classes from previous iterations are available.

In this class-incremental learning scenario, our goal is to add new classes over time, such that

1. a reasonable performance is attained on the data belonging to the new classes and
2. there is no catastrophic forgetting (performance degradation) on old classes.

4.2 Basic classification model architecture

Following the current state-of-the-art for production SLU models (Chen et al., 2019; Gaspers et al., 2021b,a; Weld et al., 2021), we consider classification model architectures that leverage large pre-trained masked language models (MLM) for the models M_i. In particular, we assume that a model M_i consists of an MLM encoder and a task-specific classification head with softmax on top. We use cross-entropy loss for the classification task.

4.3 Class-incremental learning methods

We have selected five methods from different categories, including popular C-IL approaches based on weight regularization and on data regularization, which we summarize in the following.

Warm start (BERT) A naive C-IL strategy is simply to continue fine-tuning the previous model on new data. Recently, it has been shown that pre-trained language models can effectively transfer task-agnostic knowledge to task-specific knowledge, and fine-tuning is a commonly used technique to mitigate model biases (Du et al., 2021). Therefore, "warm starting" the optimization rather than initializing randomly from scratch may be useful for quick adaptation to the new data and incorporating the new classes. However, on the downside, a direct pre-train-then-fine-tune approach is prone to catastrophic forgetting of previous knowledge (Ash and Adams, 2020).

Weight regularization-based approaches focus on preventing weight drift to consolidate previous knowledge when learning a new task. We include three methods that fall into this category: L2, Elastic Weight Consolidation (EWC) and RecAdam. As noted in the introduction, we only consider methods that do not increase model size with time, as this would be not restricted in a real-life runtime environment.

L2 When training on a new task, the importance of each parameter is used to penalize changes to them. Therefore, in addition to the classification loss (cross-entropy), we add an L2 regularised loss:

$$L(\theta^i) = L_{\text{cross}}(*) + \sum_{p}^{N} (\theta_p^{i-1} - \theta_p^i)^2, \qquad (1)$$

where N = total number of parameters of $|\theta^{i-1}|$.

EWC Kirkpatrick et al. (2017) proposed *Elastic Weight Consolidation* (EWC) that is a weight regularization method that penalizes parameter updates according to the model's sensitivity. The model sensitivity is calculated as a diagonal approximation of the Fisher Information Matrix F_i, which captures the importance of the model at the minimum after each task is learned, while ignoring the influence of those parameters along the learning trajectory in weight space. The modified loss with EWC is defined as:

$$L(\theta^i) = L_{\text{cross}}(*) + \sum_{p}^{N} \frac{\lambda}{2} F_p (\theta_p^{i-1} - \theta_p^i)^2 \qquad (2)$$

where N = total number of parameters of $|\theta^{i-1}|$.

RecAdam. Recently, Chen et al. (2020) proposed RecAdam to address the problem of sequential transferring regime of deep pre-trained LMs. They assume that the pre-training data is not available during fine-tuning on a new task. They propose an optimizer that consists of two modules: Pre-training Simulation and Objective Shifting, where the former allows the model to learn source tasks without pre-training data, and the latter allows the model to focus on target tasks. Recadam was motivated by EWC and it keeps a copy of the pretrained parameters and accesses them at each training step. Recadam introduces a quadratic penalty between pretrained and fine-tuned weights in the optimization objective to prevent deviating from the pre-trained model weights. In our class incremental learning setting, we also keep the model's parameters trained on previous classes and access them for training on new class data.

Knowledge distillation. Knowledge distillation (KD) was initially proposed by Hinton et al. (2015) to encourage the outputs of one model to approximate the outputs of another, and hence it can be applied to prevent activation drift. In this study, we adopted the Teacher-Student framework for KD. In our case, at C-IL stage i, the teacher is the model M_{i-1} which was already trained on the previous datasets. The student model is a clone of the teacher, which is fine-tuned using a combined loss function:

$$L = (1 - \lambda) L_{\text{cross}} + \lambda L_{KD}, \qquad (3)$$

where L_{cross} is the task-specific cross-entropy loss, which is computed on the new dataset D_i. The knowledge distillation loss L_{KD} is computed as the cross-entropy between the output probability distributions provided by the student and teacher models. λ is a hyper-parameter balancing the two losses. We set the hyper-parameter λ to $\frac{k_{old}}{k_{old}+k_{new}}$ where k is the number of classes.

Note in all of the above setups we initialize (warm-start) the next model M^i with the previous M^{i-1} model.

5 Experimental set up

In our experiments, we study two C-IL scenarios: without any old data available and assuming that a small amount of unlabeled data for old classes can be accessed. In the following, we first discuss both scenarios in more detail. Subsequently, we describe the dataset and class-incremental learning simulation, and finally the experimental settings.

5.1 Class-incremental learning scenarios

In our experiments, we study the following two class-incremental learning scenarios:

1. **Core C-IL scenario with no previous data.** This scenario can be referred to as the "true" class-incremental learning scenario described in section 4.1 and this is the most extreme scenario for incremental learning. We assume that at C-IL stage *i* we only have access to the previous model M_{i-1} and the new dataset D_i covering new classes. However, **no data** corresponding to previous classes is available. We study this scenario because storing data for long is often impossible and training a model from scratch is expensive in real-world scenarios. Therefore, we assume that we only have a previously trained model available but no old exemplars (i.e., no data corresponding to old classes) in this setting.

	Limited unlabeled data		Core C-IL no previous data		
IS	Train	TC	Train	TC	Dev/Test
0	5000	1-50	5000	1-50	1000/1500
1	1250	1-70	1000	51-70	1400/2100
2	1350	1-90	1000	71-90	1800/2700
3	1450	1-110	1000	91-110	2200/3300
4	1550	1-130	1000	111-130	2600/3900
5	1650	1-150	1000	131-150	3000/4500

Table 1: Data statistics for the IL setup on the OOS datase. **IS**: Incremental stage, **Train**: available training data at each stage, **TC**: Included training classes.

2. **Limited unlabeled data availability C-IL scenario**. In this scenario, we assume that a limited amount of **unlabeled** data for older classes is available. While we are aiming for a technique that succeeds in scenario 1, this is a very challenging problem to solve. On the other hand, in real-world settings, some unlabeled data of previous classes sometimes can be collected again or stored for a short period. Thus, scenario 2 is also reasonable and included for comparison regarding what performance can be reached when a small amount of unlabeled data of previous classes is available vs. unavailable. In this scenario, we additionally leverage semi-supervised learning and we use the previous model M_{i-1} to label the unlabeled data at the incremental stage i. Note that unlabeled data are much less expensive than labeled data as no annotators are needed, and it is preferable w.r.t. privacy concerns, as no human needs to look into the utterances to annotate them.

In both cases, we assume that an initial model M_0 is trained first, covering 50 intent classes. Subsequently, five incremental learning stages are conducted, and in each stage 20 new classes are added.

5.2 Data

We evaluate our models using the out-of-scope (OOS) dataset (Larson et al., 2019) for intent classification. The OOS dataset comprises English user queries which were annotated with intents. It contains 150 intent classes, and our goal is that model's learn these 150 classes incrementally.

For class-incremental learning experiments, we first randomly selected 50 classes and used all corresponding training and development data for training the initial model M_0. Next, we randomly split the remaining classes into 5 groups, each comprising 20 classes, and we created datasets by collecting the data for the groups of classes. This process resulted in datasets covering 5 incremental stages (IS), which we split to create training, development and test sets for each incremental stage.

To simulate the second C-IL scenario with limited unlabeled data amounts for old classes, we set 5% of data aside per class and dropped labels. Next, we use the previously trained model (M_{i-1}) to annotate the unlabeled data for the next incremental stage and add the annotated data to the new labeled training data.

The data splits and statistics are reported in Table 1. The number of exemplars per class in each incremental phase is uniform.

5.3 Hyperparameter details and metrics

We use the BERT-large-uncased model from the Huggingface Transformers package. The BERT architecture type is widely used in practical applications[1] and we limit our experiments to this common architecture type and instead choose to evaluate multiple scenarios and C-IL methods (see Section 5). We train five random initializations of each model, reporting the mean accuracy for all of the experiments. This results in over 180 experimental runs for the two setups, the five tested methods and the oracle. For fine-tuning, the learning rate and regularization strength are selected through 5-fold cross-validation on the BERT warm start train data, selecting the model with the highest training accuracy. We choose the regularization strength λ from $\{10^6, 2{*}10^6, 10^7, 2{*}10^7\}$ and three learning rates in $\{2*10^6, 4*10^6, 6*10^6\}$. We also use the following hyperparameters: (a) Embedding dimension: 768, (b) Optimizer: AdamW, and (c) Gradient Norm: 10.0. Our main evaluation metric at each incremental step is the standard multi-class accuracy. We report overall accuracy results, as well as the break down for old and new classes.

6 Experimental Results

Recall that the core challenge with class incremental learning methods is to balance retaining knowledge from old exemplars while learning new knowledge for the new exemplars. In our experiments, we investigate how the different incremental learning methods perform on both preventing *catastrophic forgetting* and *intransigence* i.e., resistance to learn

[1]C.f. the number of downloads in the last months for BERT models `https://huggingface.co/models?sort=downloads&search=bert`

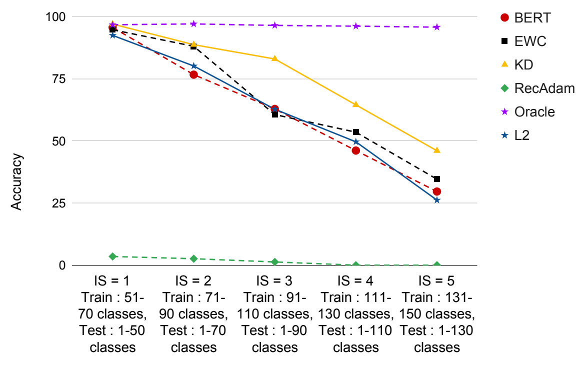

Figure 1: Average accuracy of the C-IL methods on all seen classes so far until the previous incremental phase for the core C-IL scenario.

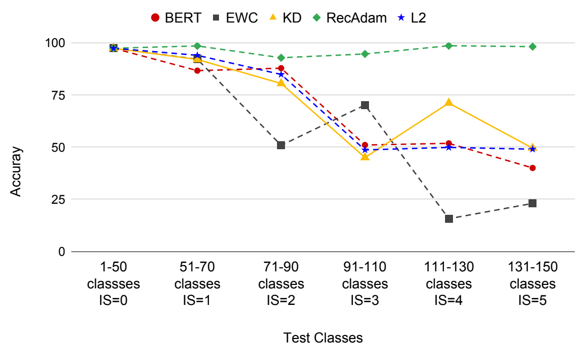

Figure 2: Accuracy of the C-IL methods methods on new class labels at different IL stages for the core C-IL scenario.

about new exemplars. Where appropriate, we also include performance of an "oracle"; contrasting with the C-IL techniques, the oracle has access to all of the previously available labeled datasets. It should be seen as an upper bound, indicating what performance would be possible if older labeled data could be kept for model training.

6.1 Core C-IL scenario

Forgetting over time We first study the performance of the C-IL methods in addressing the catastrophic forgetting problem. Figure 1 compares the different C-IL methods on old class labels at different incremental learning stages. Firstly, we observe that the KD method performs consistently better than other C-IL methods in the trend of accuracy at different incremental learning stages. Next, we find that RecAdam suffers the most significant performance drop. Interestingly, the performance of EWC until incremental stage (IS) 3 is comparable to BERT WARM-START and L2. However, as the number of incremental stages increases, the performance of BERT WARM-START and L2 comparatively drops more.

We also report the upper bound results denoted as *Oracle* where models are trained with all training data of the classes learned so far. The gap in performance of KD and EWC appears unbridgeable after IS 2. This suggests that (1) only constraining old parameters does not suffice to prevent forgetting and (2) there is a positive effect of the distribution information of previous features in C-IL.

Performance on new exemplars To analyze the effectiveness of different C-IL models more concretely, we explore how they affect the new class label's accuracy. Figure 2 shows the performance of different methods on newer class labels at different incremental phases. We find that RecAdam performs better on the new class labels. The RecAdam optimizer was designed to improve the model's performance on the fine-tuning task by not deviating too much from the pretrained model parameters. This could be an intuitive reason for RecAdam's strong performance on newer class labels. This also aligns with the poor performance of RecAdam on the older exemplars (see Figure 1).

We observe that BERT WARM-START performs better than KD on new class labels at the earlier incremental stages. However, when the number of stages increases, the KD method outperforms BERT WARM-START and EWC in learning about the new class labels and remembering previous labels (see Figure 1). This indicates that the performance of the KD method is better in reducing the forgetting problem and performs better on newer class labels when we increase the number of incremental stages.

Analysis Tables 2a-2e present the performance with respect to *average Accuracy* and *Forgetting Rates* of each method at different incremental stages. The last column (red cells) of each table represents the *average accuracy*:

$$\overline{A} = \frac{1}{N+1}\sum_{i=0}^{N} A_i, \qquad (4)$$

where A_i is accuracy on the test dataset D_i^{test} comprising data belonging to the k_i batch of classes. For example, the $\overline{A}$ column at M_1, M_2,.., M_5 represents the performance on classes 1-70, 1-90, .., 1-150 respectively. The diagonal cells represent the performance of each method on new class labels (this is also the data visualized in Figure 2). In this case, *IS*=1, *IS*=2,.., *IS*=5 means the performance on classes $51-70$, $71-90$, .., $131-150$ respectively.

The lower triangle (**blue cells**) of each table represents *Forgetting Rates* (lower is better) : $F = A_{i,i} - A_{i,j}$, where $j < i$. Similar to (Liu et al., 2020), we define a forgetting rate, denoted as F, by calculating the difference between the accuracy of the old model (M_k) and the new model (M_{k+i}) on the same test data. For example, in Table 2a, the M_1 model at *IS*=0 (i.e., performance on 1-50 classes) forgets 1.81 accuracy points with respect to the model M_0.

The bold numbers represent the best performing model at different incremental stages. There are two main observations: (1) For class incremental learning, we hypothesize that the model is prone to suffer from more severe forgetting as the incremental stage increases. We find that although there was some big drop after training on the 3^{rd} incremental stage, KD forgetting rate is low. Interestingly, the forgetting rate for EWC is relatively low. We find that with the EWC method the results at some incremental phases have negative forgetting rates suggesting that a new model (such as M_3, M_4, M_5) performs better than the corresponding previous model for some old classes. One intuitive reason could be that the performance on the new labels for EWC is comparatively poor compared to BERT WARM-START and the KD method. KD maintains stable performance as the number of incremental stages increases. Especially after training on the 4^{th} and 5^{th} stage, the forgetting increment was relatively small, which demonstrated the robustness of KD. (2) After each individual phase, the learned model M_i is evaluated on the test data $D^{test}_{o:i}$, where $0 : i$ denotes all seen classes so far. We observe that initially all methods except RECADAM work well but as the incremental phase increases to 5^{th}, the KD method gains $+14.2$ pp.

	Forgetting Rate						
	IS=0	**IS=1**	**IS=2**	**IS=3**	**IS=4**	**IS=5**	$\overline{A}$
M_0	97.4						
M_1	**1.81**	86.66					92.3
M_2	9.55	38.36	87.83				80.6
M_3	22.15	35.66	24.83	51			59.3
M_4	30.03	63.66	45.33	31.5	51.8		49.0
M_5	**38.03**	71.5	71.67	46	43.97	40.0	29.1

(a) BERT Warm Start

	Forgetting Rate						
	IS=0	**IS=1**	**IS=2**	**IS=3**	**IS=4**	**IS=5**	$\overline{A}$
M_0	97.4						
M_1	2.74	92.16					93.57
M_2	**3.81**	18.16	50.83				81.2
M_3	21.55	65.33	-5.17	70.16			56.08
M_4	29.43	64.16	-4.33	**29.16**	15.66		47.41
M_5	39.7	77.5	26.83	56.66	-12.67	23.0	29.6

(b) EWC

	Forgetting Rate						
	IS=0	**IS=1**	**IS=2**	**IS=3**	**IS=4**	**IS=5**	$\overline{A}$
M_0	97.4						
M_1	2.6	92					**94.6**
M_2	6.88	**8**	80.5				**82.6**
M_3	**11,75**	**11.5**	**1.84**	45			**72.66**
M_4	**20.55**	**18**	**11.65**	23.4	71.11		**55.17**
M_5	40.7	**44.17**	**39.39**	**31.34**	**16.45**	49.5	**43.8**

(c) KD

	Forgetting Rate						
	IS=0	**IS=1**	**IS=2**	**IS=3**	**IS=4**	**IS=5**	$\overline{A}$
M_0	97.4						
M_1	5	94					92.8
M_2	10.14	31.84	84.3				81.18
M_3	19.87	54	35.8	48.66			60.1
M_4	24.4	64	38	34.5	49.83		49.66
M_5	48.8	81	67.3	45.86	34.08	49	29.22

(d) L2

	Forgetting Rate						
	IS=0	**IS=1**	**IS=2**	**IS=3**	**IS=4**	**IS=5**	$\overline{A}$
M_0	97.2						
M_1	93.5	**98.5**					30.66
M_2	93.47	98.5	**92.8**				22.70
M_3	95	98.5	92.8	**94.66**			18.30
M_4	97.4	98.5	92.8	94.66	**98.66**		15.17
M_5	97.4	98.5	92.8	94.66	98.66	**98.16**	13.08

(e) RecAdam

Table 2: Performance of each method with respect to average accuracy (last column) and forgetting rates (lower triangle) at different IS. For average accuracy higher is better, for forgetting rates lower is better.

6.2 Limited unlabeled data available

Motivated by our findings in the supervised setting, in this scenario, we assume that we have a small amount of unlabelled data corresponding to previous class labels available. We use the previously trained model to automatically annotate the unlabelled data and add this data into the training set for training the model for the next phase.

Figure 3 presents the results of the different C-IL methods on class labels seen so far until the previous incremental learning stages. Firstly, we observe that the performance of all C-IL methods benefits from the extra data. Interestingly, the performance of the EWC method is much more stable than without any previously labelled data. Moreover, the KD method still performs relatively bet-

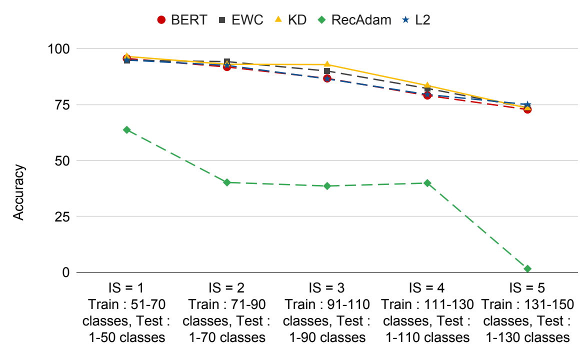

Figure 3: Average accuracy in the setting where limited unlabeled data is available for the C-IL methods on all seen classes so far until the previous incremental phase.

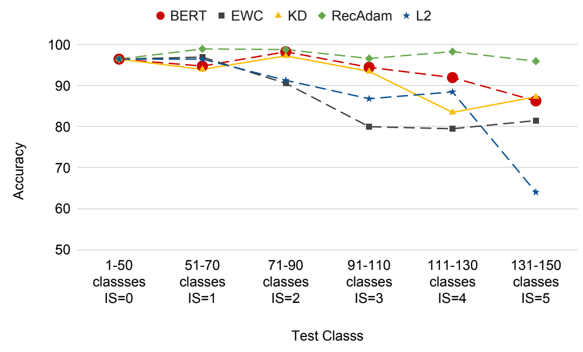

Figure 4: Accuracy in the setting where limited unlabeled data is available for the C-IL methods on new class labels at different incremental learning stages.

ter than other methods in preventing the forgetting problem. Figure 4 depicts the performance on the newer class labels at each incremental phase. BERT WARM-START works better than the KD and EWC methods, indicating that extra noisy data helps BERT WARM-START more. Similar to Figure 2 we observe that RECADAM performs better on newer class labels, indicating that it is over-fitted to newer exemplars.

The average accuracy results are shown in Table 3. The EWC and KD methods perform best by a large margin. EWC shows a small gain for the first couple of incremental phases compared with KD and L2. However, the gain increases as more incremental phases are conducted. Regarding the final incremental classifier on all classes, the KD method outperforms EWC, L2 and BERT WARM-START by 0.8%, 2% and 2.44% respectively.

7 Conclusion

In this paper, we explored class-incremental learning for the intent classification task. In particular, we compared several methods, i.e., BERT warm start, L2, Elastic Weight Consolidation, RecAdam and Knowledge Distillation. We compared performance within two class-incremental learning scenarios: one where only the previous model was assumed to be available, but no data corresponding to old classes, and one in which limited unlabeled data for old classes was assumed to be available.

We are the first to benchmark these methods in the challenging incremental learning setup for intent classification motivated by real-life restrictions where no old data might be available. We presented extensive experiments on the out-of-scope dataset for intent classification. Among the investigated continual learning methods, Knowledge Distillation worked best for our class-incremental learning tasks, and adding limited unlabeled data helped the model in both adaptability and stability. We plan to add token-level slot prediction task to our setup in the future and include further MLM models beyond just the BERT architecture.

Test classes	70	90	110	130	150
Method	M_1	M_2	M_3	M_4	M_5
BERT Warm-start	95.52	91.5	86.81	80.2	73.06
EWC	95.4	93.4	88.18	81.7	74.7
KD	95.7	91.7	89.18	82.5	75.5
RecAdam	73.8	53.22	49.18	48.89	14.2
L2	95.38	92.25	86.5	80.8	73.5

Table 3: *Average Accuracy* of different C-IL methods in the setting where limited unlabeled data is available.

8 Ethical considerations

The experiments presented in this paper are performed with publicly available models and methods on a public dataset and can be verified independently. Our experimental setup is motivated by a real-life problem of regular SLU model releases, where it is critical to maintain the performance on old classes and not to introduce new errors into the existing model so that no user is negatively affected by the changes. The incremental learning techniques discussed here have a potential to improve user privacy, as they do not rely on storing the old data. Training only on the data incremental has also a significant impact on model training times and resource usage and, as a consequences, improves the environmental impact of a model release pipeline.

References

Jordan Ash and Ryan P Adams. 2020. On warm-starting neural network training. *Advances in Neural Information Processing Systems*, 33:3884–3894.

Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. 2020. Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6523–6541, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Pengfei Cao, Yubo Chen, Jun Zhao, and Taifeng Wang. 2020. Incremental event detection via knowledge consolidation networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 707–717, Online. Association for Computational Linguistics.

Giuseppe Castellucci, Simone Filice, Danilo Croce, and Roberto Basili. 2021. Learning to solve NLP tasks in an incremental number of languages. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 837–847, Online. Association for Computational Linguistics.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. BERT for Joint Intent Classification and Slot Filling. *arXiv preprint arXiv:1902.10909*.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7870–7881, Online. Association for Computational Linguistics.

Hao Cheng, Dongze Lian, Bowen Deng, Shenghua Gao, Tao Tan, and Yanlin Geng. 2019. Local to global learning: Gradually adding classes for training deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4748–4756.

Eunah Cho, He Xie, and William M Campbell. 2019. Paraphrase generation for semi-supervised learning in nlu. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 45–54.

Tom Diethe, Tom Borchert, Eno Thereska, Borja Balle, and Neil D Lawrence. 2018. Continual learning in practice. In *Proceedings of the NeurIPS 2018 workshop on Continual Learning*.

Li Du, Xiao Ding, Ting Liu, and Bing Qin. 2021. Learning event graph knowledge for abductive reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5181–5190, Online. Association for Computational Linguistics.

Judith Gaspers, Quynh Do, Tobias Röding, and Melanie Bradford. 2021a. The impact of domain-specific representations on BERT-based multi-domain spoken language understanding. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 28–32, Kyiv, Ukraine. Association for Computational Linguistics.

Judith Gaspers, Quynh Do, Daniil Sorokin, and Patrick Lehnen. 2021b. The Impact of Intent Distribution Mismatch on Semi-Supervised Spoken Language Understanding. In *Proc. Interspeech 2021*, pages 4708–4712.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Kasidis Kanwatchara, Thanapapas Horsuwan, Piyawat Lertvittayakumjorn, Boonserm Kijsirikul, and Peerapon Vateekul. 2021. Rational LAMOL: A rationale-based lifelong learning framework. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2942–2953, Online. Association for Computational Linguistics.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521 – 3526.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Zhizhong Li and Derek Hoiem. 2018. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:2935–2947.

Yaoyao Liu, Anan Liu, Yuting Su, Bernt Schiele, and Qianru Sun. 2020. Mnemonics training: Multi-class incremental learning without forgetting. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12242–12251.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 480–496, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Martial Mermillod, Aurélia Bugaiska, and Patrick BONIN. 2013. The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in Psychology*, 4.

Grégoire Mesnil, Xiaodong He, Li Deng, and Yoshua Bengio. 2013. Investigation of Recurrent-Neural-Network Architectures and Learning Methods for Spoken Language Understanding. In *Interspeech*, pages 3771–3775, Lyon, France.

Justin Payan, Yuval Merhav, He Xie, Satyapriya Krishna, Anil Ramakrishna, Mukund Sridhar, and Rahul Gupta. 2021. Towards realistic single-task continuous learning research for NER. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3773–3783, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alex Sokolov and Denis Filimonov. 2020. Neural machine translation for paraphrase generation. *arXiv preprint arXiv:2006.14223*.

James Thorne and Andreas Vlachos. 2021. Elastic weight consolidation for better bias inoculation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 957–964, Online. Association for Computational Linguistics.

H. Weld, X. Huang, S. Long, J. Poon, and S. C. Han. 2021. A survey of joint intent detection and slot-filling models in natural language understanding.

Tongtong Wu, Massimo Caccia, Zhuang Li, Yuan-Fang Li, Guilin Qi, and Gholamreza Haffari. 2022. Pretrained language model in continual learning: A comparative study. In *International Conference on Learning Representations*.

Xiaodong Zhang and Houfeng Wang. 2016. A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding. In *Proceedings of the Twenty-Fifth IJCAI*, page 2993–2999, New York, NY, USA.