Backward Compatibility During Data Updates by Weight Interpolation

Raphael Schumann[*1], **Elman Mansimov**[2], **Yi-An Lai**[2],
Nikolaos Pappas[2], **Xibin Gao**[2], **Yi Zhang**[2]
[1]Computational Linguistics, Heidelberg University, Germany [2]AWS AI Labs
rschuman@cl.uni-heidelberg.de
{mansimov,yianl,nppappa,gxibin,yizhngn}@amazon.com

Abstract

Backward compatibility of model predictions is a desired property when updating a machine learning driven application. It allows to seamlessly improve the underlying model without introducing regression bugs. In classification tasks these bugs occur in the form of negative flips. This means an instance that was correctly classified by the old model is now classified incorrectly by the updated model. This has direct negative impact on the user experience of such systems e.g. a frequently used voice assistant query is suddenly misclassified. A common reason to update the model is when new training data becomes available and needs to be incorporated. Simply retraining the model with the updated data introduces the unwanted negative flips. We study the problem of regression during data updates and propose Backward Compatible Weight Interpolation (BCWI). This method interpolates between the weights of the old and new model and we show in extensive experiments that it reduces negative flips without sacrificing the improved accuracy of the new model. BCWI is straight forward to implement and does not increase inference cost. We also explore the use of importance weighting during interpolation and averaging the weights of multiple new models in order to further reduce negative flips.

1 Introduction

In conventional software development it is an established routine to identify and fix regression bugs before deploying a new version. Regression bugs describe defects in already existing features and are particularly sensible for end users because accustomed workflows are affected. In machine learning driven applications however the main focus usually lies on improving the underlying model and regression is rarely measured, let alone actively

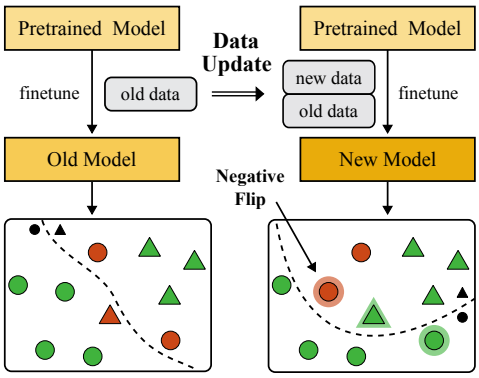

Figure 1: The left column shows the common workflow of finetuning a pretrained model on a given dataset in order to learn a classifier. A data update occurs when new data becomes available and is added to the existing data. The right column depicts the finetuning of the pretrained model on the updated (old and new) data. After expanding the training set, the new model makes more correct predictions compared to the old model. Despite this, the prediction of some instances are flipped from the correct label to an incorrect one. These so called regression errors hinder the adoption of the new model. Our work proposes to interpolate weights of old and new model in order to reduce those negative flips during data updates.

mitigated. This prevents backward compatibility of e.g. visual search systems (Shen et al., 2020) or virtual voice assistants (Cai et al., 2022) and leads to humans loosing trust in AI systems (Bansal et al., 2019; Srivastava et al., 2020). Previous work on mitigating regression in machine learning models focuses on cases where the model architecture (Yan et al., 2021; Cai et al., 2022) or pretraining procedure (Xie et al., 2021) is updated. For example, updating a finetuned BERT model (Devlin et al., 2019) to a RoBERTa based model (Liu et al., 2019) which is finetuned on the same task specific data. Such fundamental modifications are done rather infrequently and it is more common to update the training data of a model in order to improve a deployed system. One such type of machine learning

[*]Work done while interning at AWS AI Labs. Correspondence to: Elman Mansimov <mansimov@amazon.com>

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 2846–2861

based system that undergoes frequent data updates are virtual assistants and chatbots. A data update consists of additional labeled utterances and commonly aims to improve classification performance or to support new classes. Training a new model on the updated data introduces regression in the form of negative flips. As depicted in Figure 1, a negative flip is a data point that was correctly classified by the old model and is now classified incorrectly by the new model. This happens despite the overall better accuracy of the new model. From a user's perspective it seems as if the virtual assistant or chatbot got worse because familiar utterances are suddenly misinterpreted. On the other hand, the overall better accuracy is only perceived over time. The negative user impact of regression and abundance of data updates motivates us to study the mitigation of regression during data updates in multi-class text classification. The outlined data update setting can be categorized as a continual learning problem. But in two key aspects it is distinct from the commonly studied continual learning setting (Parisi et al., 2019). (i) We assume full access to the old training data. As such, catastrophic forgetting in terms of accuracy drop is avoided by joint training. (ii) We instead measure forgetting/interference by number of negative flips between old model and new model.

To reduce negative flips during data updates, we propose Backward Compatible Weight Interpolation (BCWI) in this paper. BCWI describes the interpolation between the weights of the old model and the weights of the new model. The interpolation largely recovers the prediction pattern of the old model without hurting the improved accuracy of the new model. The method is informed by recent success of weight interpolation for robust finetuning (Wortsman et al., 2022b) and model patching (Ilharco et al., 2022). While these works focus on avoiding catastrophic forgetting in terms of task accuracy, we are interested in reducing negative flips while maintaining high accuracy. We further introduce FisherBCWI which uses the Fisher information matrix as importance weighting (Kirkpatrick et al., 2017; Matena and Raffel, 2021) and SoupBCWI which employs soup ensembles (Wortsman et al., 2022a) to further reduce negative flips. The proposed methods do not modify the training process and do not increase inference cost. We describe BCWI and its variants in detail and empirically show on three datasets and two update scenarios (adding i.i.d. data and adding new classes) that they reduce negative flips by up to three times while maintaining the improved accuracy of the new model. This property of weight interpolation has not been explored before and constitutes a substantial step towards regression free data updates.

2 Related Work

Mitigating Regression Previous work focuses mainly on reducing negative flips when updating the model architecture or the pretraining procedure. In these settings the available data is static and not affected by the update as in our work. Negative flips are either minimized by training with distillation loss while using the old model as teacher (Yan et al., 2021; Xie et al., 2021; Jiang et al., 2022; Hidey et al., 2022) or ensembling techniques (Yan et al., 2021; Xie et al., 2021; Zhao et al., 2022; Deng et al., 2022; Liu et al., 2022b). Cai et al. (2022) introduces a method specifically for structured prediction that uses the old model to rerank the output beams of the new model. Model regression is also known as prediction churn or jitter (Milani Fard et al., 2016; Toneva et al., 2018; Liu et al., 2022b). Our proposed method uses weight interpolation in order to move the new model closer to the old model post training.

Weight Interpolation Weight interpolation and weight averaging are known to improve classification performance in different settings. Averaging the weights of multiple model checkpoints along a cyclic learning rate schedule leads to better classification generalization (Izmailov et al., 2018). Averaging the weights of multiple models, initialized by the same pretrained model and finetuned with different hyperparameter, improves accuracy in classification tasks (Wortsman et al., 2022a) and out-of-distribution generalization (Rame et al., 2022). Weight interpolation is also used to merge the task specific accuracy of a finetuned model with the zero-shot capability of its ancestor model (Wortsman et al., 2022b; Ilharco et al., 2022). Looking beyond simple averaging, Matena and Raffel (2021) use the Fisher information matrix to scale each model weight by importance. We use the same importance weighting for the FisherBCWI method. To the best of our knowledge, we are the first to explore weight interpolation for mitigating data update regression.

Continual Learning Continual learning studies the problem of incrementally adding new knowl-

edge to a model while avoiding catastrophic forgetting (Ratcliff, 1990; McCloskey and Cohen, 1989). Knowledge arrives in the form of new tasks, additional classes or data with shifted distribution (Lange et al., 2022). Catastrophic forgetting is measured by accuracy drop on previous data and tasks. It arises from the imposed constraint that one has none or limited access to previous data when new knowledge is incorporated. The constraint is motivated by analogy of how humans learn over time (McCloskey and Cohen, 1989) or storage feasibility (Sodhani et al., 2022). We instead allow access to the old data because the amount is manageable and we do not focus on simulating lifelong learning. This setting reinforces the need to measure catastrophic forgetting and interference not only in terms of overall accuracy, but also in terms of reducing negative flip rate.

Weight regularization is a common way to prevent catastrophic forgetting by preventing the model weights to deviate too far from the old model. Prior Weight Decay (Wiese et al., 2017; Lee et al., 2020) moves the current model weights in the direction of the weights of the old model during each training step. Mixout (Lee et al., 2020) randomly replaces a subset of the current weights with the weights of the old model at each training step. Kirkpatrick et al. (2017) introduce Elastic Weight Consolidation (EWC) which uses the diagonal Fisher information matrix to weigh the importance of each model parameter in L2 regularization. We show in our experiments that the weight regularization techniques also reduce the number of negative flips.

3 Problem Formulation: Regression in Data Updates

In order to measure regression in classification models, Yan et al. (2021) introduced negative flip rate (NFR):

$$\text{NFR} = \frac{1}{N}\sum_{i}^{N} \mathbb{1}[f_{\theta_{old}}(x_i) = y_i \wedge f_{\theta_{new}}(x_i) \neq y_i], \quad (1)$$

where $f_{\theta_{old}}$ is the old model and $f_{\theta_{new}}$ the new, updated model. NFR is measured on a given regression set with N input and label pairs (x, y). Negative flips are instances that are predicted correctly by the old model and are incorrectly predicted by the new model. Consequently, NFR is the ratio of negative flips to the total number of instances in the regression set, i.e., the development or test set. We formulate the problem of minimizing regression during data updates in the following way. A deployed model with weights θ_{old} was trained by finetuning a pretrained model θ_{pre} on currently available data D_{old}:

$$\theta_{old} = \arg\min_{\theta} \mathcal{L}(\theta|\theta_{pre}, \mathcal{D}_{old}), \quad (2)$$

where $\mathcal{L}$ is the classification loss. We now obtain additional data and update the available data to get $\mathcal{D}_{upd} = \mathcal{D}_{old} \cup \mathcal{D}_{new}$. This larger dataset allows us to train a new model that achieves better classification performance than the old model. A straight forward way to do so is to finetune the initial pretrained model on the updated data (old and new data):

$$\theta_{new\_target} = \arg\min_{\theta} \mathcal{L}(\theta|\theta_{pre}, \mathcal{D}_{upd}). \quad (3)$$

This is the process depicted in Figure 1 and, unfortunately, leads to high negative flip rate which in turn limits the compatibility between the old and new model. The goal of this work is to find a method that emits a new model which produces minimal negative flips while achieving the same classification performance as the target model:

$$\begin{aligned} \theta^* &= \arg\min_{\theta} \mathcal{R}(\theta, \theta_{old}) \\ s.t.\ &\mathcal{M}(\theta^*) \approx \mathcal{M}(\theta_{new\_target}), \end{aligned} \quad (4)$$

where $\mathcal{R}$ is the regression metric and $\mathcal{M}$ measures the classification performance. In our work these are negative flip rate and accuracy, respectively. To account for variance, the equality of classification metrics can be defined as e.g. overlapping confidence intervals.

4 Proposed Method: Backward Compatible Weight Interpolation

We start with the intuitive observation that negative flips are reduced when using the old model as the starting point for finetuning the new model:

$$\theta_{new} = \arg\min_{\theta} \mathcal{L}(\theta|\theta_{old}, \mathcal{D}_{upd}). \quad (5)$$

Next we interpolate between the weights of the old and new model:

$$\theta_{\text{BCWI}} = \alpha\theta_{old} + (1-\alpha)\theta_{new}, \quad (6)$$

where $\alpha \in [0.0, 1.0]$ is the interpolation parameter and regulates the trade-off between classification performance and negative flip rate. A larger α moves the model closer to the old model, reducing

negative flip rate but ultimately sacrifices the improved classification performance. We empirically show that in all but one of the conducted experiments there exists an $\alpha > 0$ that results in a model that achieves the same classification performance as the target model while significantly reducing negative flips. We call this method Backward Compatible Weight Interpolation (BCWI).

4.1 FisherBCWI

The interpolation with a single parameter might not be optimal because not every model weight is equally contributing to a model's predictions. The importance of each weight can be quantified by the diagonal of the empirical Fisher information matrix (Kirkpatrick et al., 2017; Matena and Raffel, 2021):

$$F_{old} = \frac{1}{c}\sum_{i}^{N}(\nabla_{\theta_{old}} \log p(y_i|x_i))^2, \quad (7)$$

where c is a normalization constant and $\nabla_{\theta_{old}}$ is the gradient in respect to the weights of the old model. By using $F_{old} \in \mathbb{R}^{|\theta_{old}|}$ as the importance factor for each parameter in the old model we get:

$$\theta_{\text{FisherBCWI}} = \frac{\alpha F_{old}\theta_{old} + (1-\alpha)\theta_{new}}{\alpha F_{old} + (1-\alpha)}, \quad (8)$$

where all operations are elementwise. The interpolation is focused on weights that are important for the old model and thus minimizes interference with the weights of the new model.

4.2 SoupBCWI

Ensembling the logits of multiple new models reduces negative flips (Yan et al., 2021; Xie et al., 2021). The inference cost increases linearly with each new model in the ensemble and makes it impracticable for many applications. To alleviate this, we employ a soup ensemble (Wortsman et al., 2022a) of new models. A soup ensemble is formed by averaging the weights of multiple models that were individually finetuned from the same pretrained model. We find that the soup ensemble of new models is reducing negative flips. This is complementary to BCWI as we show by interpolating the ensemble weights towards the weights of the old model:

$$\theta_{\text{SoupBCWI}} = \alpha\theta_{old} + (1-\alpha)\frac{1}{M}\sum_{j}^{M}\theta_{new_j}, \quad (9)$$

	Add_Data Scenario			Add_Classes Scenario			
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**	**#C**
MASSIVE							
old	1,000	333	4,000	1,222	409	3,258	47
+ new	500	167	-	278	91	742	13
= updated	1,500	500	4,000	1,500	500	4,000	60
Banking77							
old	700	233	4,000	927	310	3,713	70
+ new	300	100	-	73	23	287	7
= updated	1,000	333	4,000	1,000	333	4,000	77
AG News							
old	120	60	4,000	225	113	3,000	3
+ new	180	90	-	75	37	1,000	1
= updated	300	150	4,000	300	150	4,000	4

Table 1: The dataset splits for the Add_Data and Add_Classes update scenarios constructed for the respective datasets. The *updated* data is the *old* data in addition to the *new* data. The test set is only updated when new classes are added. The #C-column lists the number of classes in the respective data portion of the AC scenario. The AD scenario includes all classes.

where M is the number of new models. Each new model is finetuned according to Equation 5 and each with a different random seed. In the next section, we motivate the data update scenarios that we use to demonstrate the effectiveness of the proposed methods.

5 Data Update Scenarios

The data that is available to train a given classification model changes over time. This can be due to several reasons. More labeled data for the existing classes is obtained by annotating instances from the initial source or from observed queries. Data for new classes is added to support additional downstream features or classes are split up to allow for more fine-grained classification. The retraining of an existing model on the evolved data basis is called *data update*. In this work, we focus on two isolated data update scenarios that cover two common use cases, namely adding i.i.d. data and adding new classes. We simulate the two scenarios in order to study the prevalence and mitigation of regression during data updates.

Add_Data Scenario In the Add_Data (AD) scenario, the amount of available data is increased by adding new instances for the current set of classes. This is the most basic type of data update and aims at improving the classification performance of the derived model. The additional data is usually obtained by annotating more instances from the initial data source or from the observed model queries.

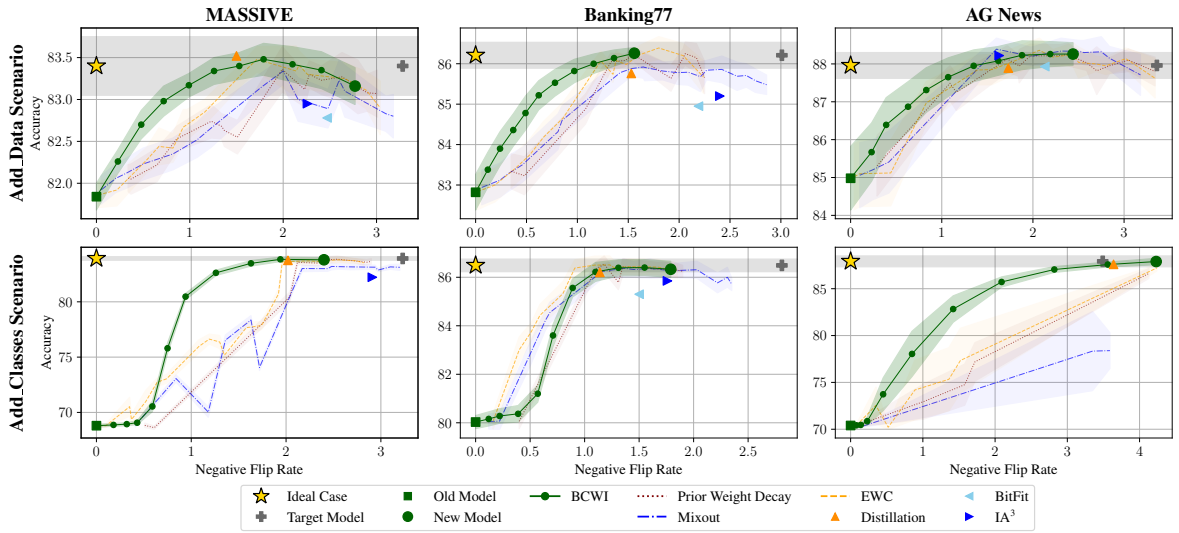

Figure 2: Results for the AD and AC scenario evaluated on our test sets of MASSIVE (FitzGerald et al., 2022), Banking77 (Casanueva et al., 2020) and AG News (Zhang et al., 2015). The gray horizontal bar is spanned by the 95% confidence interval of the target model and indicates the level of accuracy a model should reach. Baselines are Prior Weight Decay (Lee et al., 2020), Mixout (Lee et al., 2020), EWC (Kirkpatrick et al., 2017), Distillation (Xie et al., 2021), BitFit (Ben Zaken et al., 2022) and IA^3 (Liu et al., 2022a). Identical markers belong to the same method evaluated with different trade-off parameters. Markers to the right of the target model are cut off. For BCWI this is α in 0.1 steps where 0.0 is equivalent to the new model and 1.0 is equivalent to the old model. The trade-off parameters for the baselines are listed in Appendix A. The ideal case for a new model is to have zero negative flips while maintaining the target accuracy. BCWI consistently produces a model that is closer to the ideal case than any of the baseline methods and is more stable across different trade-off parameters.

While in the latter case the distribution can shift over time, we assume i.i.d. data for this scenario.

Add_Classes Scenario In the Add_Classes (AC) scenario, we study data updates that consists of adding new classes and corresponding instances to the existing data. This is necessary when the text classification based system supports new features. For example, a virtual assistant is extended with a food delivery feature, a news classification model covers emerging topics or medical reports are classified according to new diseases codes.

5.1 Datasets and Splits

We simulate the two described data update scenarios for three datasets each. MASSIVE (FitzGerald et al., 2022) is a natural language understanding dataset with 60 intents covering basic domains of a virtual assistant. We use the English portion of the data. Banking77 (Casanueva et al., 2020) includes utterances with 77 intents for a virtual assistant limited to the banking domain. AG News (Zhang et al., 2015) is a document classification dataset that categorizes news articles into four topics. Table 1 lists the number of instances in the data splits for both scenarios across all three datasets. We only use a subset of the original data and randomly sample all splits from the training set of the respective dataset. The size of the splits was chosen such that the data update leads to a significant improvement of classification accuracy. In order to simulate the addition of new classes for the AC scenario, we limit its old data splits to a subset of the available classes.

6 Experiments

We evaluate our proposed BCWI method on the above described data update scenarios, each constructed for three different datasets.[1] We choose RoBERTa (Liu et al., 2019) as a pretrained model because it is widely used and a representative encoder-only transformer model. The old model and target model are trained by finetuning $\text{RoBERTa}_{\text{BASE}}$ on the old and updated data respectively (see Equation 2 and 3). The new model is trained by finetuning the old model on the updated data (see Equation 5). The empirical Fisher information matrix used by FisherBCWI and EWC is calculated on the training and development instances. The classification performance

[1] https://github.com/amazon-science/regression-constraint-model-upgrade/tree/main/nlp

	MASSIVE		Banking77		AG News	
Model	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**
Old Model	81.8 ±0.2	0.0 ±0.0	82.8 ±0.4	0.0 ±0.0	85.0 ±0.8	0.0 ±0.0
Target Model	83.4 ±0.4	3.3 ±0.4	86.2 ±0.4	3.0 ±0.3	88.0 ±0.1	3.4 ±0.3
New Model	83.2 ±0.2	2.8 ±0.2	86.3 ±0.1	1.6 ±0.1	88.3 ±0.3	2.4 ±0.3
BitFit	82.8 ±0.3	2.5 ±0.2	85.0* ±0.4	2.2 ±0.2	87.9 ±0.3	2.1 ±0.1
IA^3	83.0 ±0.2	2.3 ±0.1	85.2* ±0.4	2.4 ±0.2	88.2 ±0.5	**1.6** ±0.2
Distillation	83.5 ±0.2	**1.5** ±0.2	85.8 ±0.2	1.5 ±0.2	87.9 ±0.5	**1.7** ±0.3
PriorWD	83.4 ±0.3	2.0 ±0.2	85.8 ±0.3	1.3 ±0.1	88.1 ±0.4	**1.7** ±0.2
Mixout	83.0 ±0.2	1.8 ±0.2	85.8 ±0.3	1.4 ±0.1	88.4 ±0.4	**1.6** ±0.2
EWC	83.3 ±0.1	**1.6** ±0.1	86.1 ±0.2	1.2 ±0.1	87.9 ±0.4	**1.6** ±0.3
BCWI	83.4 ±0.1	**1.4** ±0.1	85.5 ±0.3	**0.8** ±0.1	88.0 ±0.4	**1.5** ±0.2

Table 2: Add_Data scenario results for BCWI and baselines. Hyperparameters are tuned on the dev set. '*' indicates that there is no overlap with the target accuracy. **Bold** NFR values have overlapping 95% confidence intervals with the best value (except old model).

	MASSIVE		Banking77		AG News	
Model	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**
Old Model	68.8 ±0.1	0.0 ±0.0	80.0 ±0.3	0.0 ±0.0	70.4 ±0.1	0.0 ±0.0
Target Model	83.9 ±0.2	3.2 ±0.2	86.5 ±0.3	2.8 ±0.3	87.9 ±0.5	**3.5** ±0.6
New Model	83.8 ±0.2	2.4 ±0.1	86.3 ±0.3	1.8 ±0.2	87.9 ±0.3	**4.2** ±0.5
BitFit	81.6* ±0.2	3.3 ±0.2	85.3* ±0.4	1.5 ±0.2	86.8* ±0.3	4.9 ±0.7
IA^3	82.2* ±0.4	2.9 ±0.2	85.8 ±0.4	1.8 ±0.2	87.1 ±0.3	4.9 ±0.3
Distillation	83.8 ±0.2	2.0 ±0.2	86.2 ±0.3	**1.1** ±0.1	87.6 ±0.2	**3.6** ±0.5
PriorWD	83.3* ±0.3	2.1 ±0.2	86.3 ±0.3	**1.1** ±0.1	87.4 ±0.4	**4.3** ±0.4
Mixout	83.0* ±0.2	2.4 ±0.1	86.2 ±0.3	1.2 ±0.1	87.6 ±0.4	5.0 ±0.5
EWC	83.6 ±0.3	2.0 ±0.1	86.4 ±0.3	**0.9** ±0.1	87.9 ±0.4	**4.3** ±0.4
BCWI	83.2* ±0.2	**1.4** ±0.1	86.0 ±0.4	**1.0** ±0.1	87.6 ±0.3	**3.6** ±0.4

Table 3: Add_Classes scenario results for BCWI and baselines. Hyperparameters are tuned on the dev set. '*' indicates that there is no overlap with the target accuracy. **Bold** NFR values have overlapping 95% confidence intervals with the best value (except old model).

of each model is reported as accuracy on the updated test set. Regression is measured as negative flip rate (see Equation 1) in respect to the classifications of the old model on the updated test set. Experiments are repeated ten times with different random seeds and we report the mean and 95% confidence interval. Detailed setup and tuning of hyperparameters can be found in Appendix A. The α-values in our experiments are tuned to reach the accuracy threshold on the development set.

Baselines We compare BCWI to distillation training where the old model is used as the teacher (Yan et al., 2021; Xie et al., 2021). Besides distillation, the method additionally applies higher weight to negative flip instances in the training set. This is a strong baseline and produces state-of-the-art results for reducing regression in architecture updates. We also compare our proposed approach to methods that are used to avoid catastrophic forgetting in continual learning. Prior weight decay (Wiese et al., 2017; Lee et al., 2020) moves the current weights towards the old model at each training step. Mixout (Lee et al., 2020) randomly replaces a subset of the weights at each training step with the weights of the old model. Kirkpatrick et al. (2017) introduce elastic weight consolidation (EWC) that uses the diagonal Fisher information matrix to weigh the importance of each model parameter in L2 regularization. BitFit (Ben Zaken et al., 2022) is a parameter efficient finetuning method that only touches the bias terms of a model. IA^3 (Liu et al., 2022a) introduces additional parameters to scale the outputs of the key and value layer in multi-head attention as well as the position-wise feed-forward networks. The proper model weights are frozen. All baselines are trained by finetuning the old model according to Equation 5.

6.1 Results

We first discuss the results for the AD scenario shown in the top row of Figure 2. Looking at the MASSIVE plot, we see that the new model (see Equation 5) yields lower NFR than the target model (see Equation 3) while achieving similar accuracy. The horizontal gray bar is spanned by the 95% confidence interval around the accuracy of the target model and indicates the area of accuracy that fulfills the constraint in Equation 4. The dots along the green line are BCWI models evaluated at decreasing α-values with step size 0.1, starting from α=1.0 which is equivalent to the old model on the bottom left to α=0.0 which is equivalent to the new model. For all three datasets there is a BCWI model that lies within the gray area and has lower negative flip rate than the new model. The weight regularization baselines are evaluated with different regularization strength and the individual markers for Prior Weight Decay, Mixout and EWC are connected. The plots reveal that the baselines are competitive at accuracy levels close to the new model but drop faster than BCWI when approaching low negative flip rate. The numerical results in Table 2 show that BCWI can reduce negative flips by up to three times over the target model while maintaining the accuracy.

The second row in Figure 2 features the BCWI and baseline plots for the AC scenario. The green line which connects individual BCWI models follows an S-shaped curve with an inflection point

	MASSIVE		Banking77		AG News	
Model	**ACC**	**NFR↓**	**ACC**	**NFR↓**	**ACC**	**NFR↓**
Add_Data Scenario						
BCWI	83.4 ±0.1	1.4 ±0.1	85.5 ±0.3	0.8 ±0.1	88.0 ±0.4	1.5 ±0.2
FisherBCWI	83.5 ±0.2	2.0 ±0.2	85.5 ±0.3	0.6 ±0.1	88.1 ±0.4	1.9 ±0.3
SoupBCWI-2	83.5 ±0.1	**1.1** ±0.1	85.6 ±0.4	0.7 ±0.1	88.0 ±0.4	1.2 ±0.2
SoupBCWI-4	83.6 ±0.1	**0.9** ±0.1	85.4 ±0.4	0.6 ±0.1	88.0 ±0.3	**1.1** ±0.1
SoupBCWI-8	83.6 ±0.2	**0.8** ±0.1	85.4 ±0.3	0.6 ±0.1	88.0 ±0.3	**1.1** ±0.1
SoupBCWI-16	83.5 ±0.2	**0.7** ±0.1	85.4 ±0.4	0.6 ±0.1	87.9 ±0.4	**0.9** ±0.1
Add_Classes Scenario						
BCWI	83.2 ±0.2	1.4 ±0.1	86.0 ±0.4	1.0 ±0.1	87.6 ±0.3	3.6 ±0.4
FisherBCWI	82.9 ±0.2	1.2 ±0.1	85.7 ±0.5	**0.7** ±0.1	87.5 ±0.2	3.3 ±0.4
SoupBCWI-2	83.0 ±0.3	1.2 ±0.1	85.8 ±0.4	0.8 ±0.1	87.9 ±0.3	3.8 ±0.3
SoupBCWI-4	82.9 ±0.2	**1.1** ±0.1	85.8 ±0.3	**0.6** ±0.1	87.9 ±0.3	3.8 ±0.4
SoupBCWI-8	82.9 ±0.3	**1.0** ±0.1	85.8 ±0.3	**0.5** ±0.1	87.7 ±0.3	3.5 ±0.3
SoupBCWI-16	82.9 ±0.2	**1.0** ±0.1	85.8 ±0.3	**0.5** ±0.1	87.9 ±0.3	3.8 ±0.3

Table 4: Results for FisherBCWI and SoupBCWI in comparison with BCWI. **Bold** NFR values are lower than those of BCWI and without overlapping 95% confidence intervals.

near $\alpha \geq 0.5$ (on AG News the lower end of the S is compressed along the x-axis). This is because the old model is not trained on the new classes and their accuracy drops rapidly to zero once the old model weights dominate. For each dataset in the AC scenario there is a BCWI model that lies within the target accuracy and yield lower NFR than the new model. On AG News none of the α-values within the gray area result in lower NFR than the target model. The numerical values in Table 3 show that BCWI is as good as or better than the baselines in reducing regression at the same accuracy level.

BCWI Variants We discuss the results for the BCWI variants proposed in Section 4.1 and 4.2 in this paragraph. FisherBCWI uses the diagonal Fisher information matrix as importance weighting when interpolating between old and new model. In Table 4 we can see that it produces less negative flips than vanilla BCWI in the majority of experiments. This shows that studying interpolation schemes beyond linear is a promising research direction to further reduce negative flips. The results for SoupBCWI, also presented in Table 4, reveal that interpolating the weights of a soup ensemble (Wortsman et al., 2022a) with the weights of the old model significantly reduces negative flips. The effect slows down after more than four new models in the soup ensemble. We present the full trade-off trajectories for FisherBCWI and SoupBCWI in Appendix B.

	Additional Memory	Training Time	Tune Trade-Off	Inference Cost
EWC	$\lvert F\rvert + \lvert\theta_{old}\rvert$	$t(F)$ + 1.9x	retrain	1x
Prior WD	$\lvert\theta_{old}\rvert$	1.1x	retrain	1x
Mixout	$\lvert\theta_{old}\rvert$	1.6x	retrain	1x
BCWI	-	1x	post training	1x
FisherBCWI	-	$t(F)$ + 1x	post training	1x
SoupBCWI	-	Mx	post training	1x
Ensemble	-	Mx	post training	Mx

Table 5: Properties of the proposed methods in comparison to considered baselines. *Additional Memory*: Number of additional values that need to be held in GPU memory during training. *Training Time*: Factor by which the training time of the new model is increased. *Tune Trade-Off*: Weather it is necessary to retrain the model in order to tune the accuracy-NFR trade-off. *Inference Cost*: Factor by which inference cost is increased. F is the diagonal Fisher information matrix with size $|\theta|$ and with compute time $t(F)$ roughly equal to one epoch. M is the number of new models in the (soup) ensemble.

7 Analysis

Method Properties In this section we discuss the training and inference resources required by BCWI and the utilized baselines listed in Table 5. The weight regularization baselines have higher GPU memory requirements because they need to access the weights of the old model at each training step. The calculations necessary to evaluate the regularization terms amount to $1.1 - 1.9\times$ longer training time. EWC additionally keeps the Fisher information matrix in memory, which is pre-calculated before training. The pre-calculation takes the time of roughly one epoch of training. This is also necessary for the FisherBCWI method. In order to tune the regularization strength of EWC, Prior WD and Mixout, the model needs to be retrained entirely. On the other hand, the α-value of BCWI is tuned after the training is completed, by interpolating the converged model weights. This property of BCWI is a big advantage because it saves training resources and allows to quickly adjust to e.g. user complaints about too many regression errors in an updated model.

Output Ensemble of Old and New Model The weight interpolation between old and new model can be seen as an ensemble in weight space. But in contrast to output ensembles, it does not increase inference cost. In an output ensemble, the input needs to be passed through both models and the final prediction is a combination of the two output probability distributions. This renders output en-

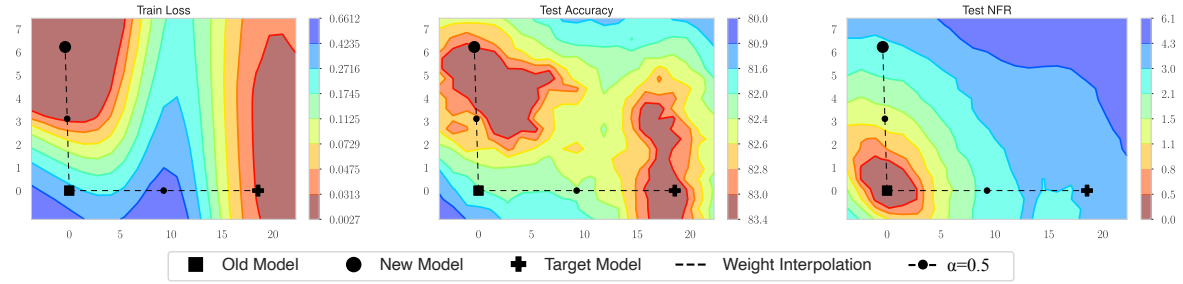

Figure 3: Visualization of training loss, test accuracy and negative flip rate for the AD scenario on MASSIVE. Visualization technique from Izmailov et al. (2018). The x and y axes denote euclidean distance. On the bottom left of each plot is the old model and dotted lines represent points along the linear interpolation towards the new model and target model.

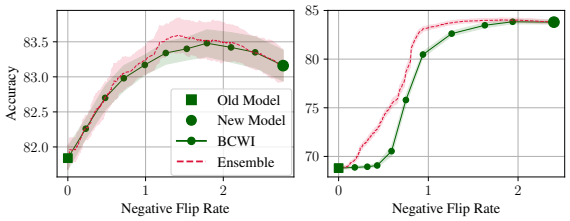

Figure 4: BCWI in comparison with output ensemble of old and new model on MASSIV. The ensemble is calculated as the weighted average of output probabilities. AD scenario is on the left and AC scenario on the right.

sembles impracticable in many applications, especially real-time systems. Figure 4 shows the graphs for weight interpolation and weighted average of output probabilities. The trajectories in the AD scenario are similar and in the AC scenario the output ensemble performs slightly better. This highlights that BCWI conveys most of the improvements in regression mitigation, but without the downside of increased inference cost from running the old and new model.

Loss Landscape To better understand BCWI, we visualize the loss and error landscapes for the old, new and target model in Figure 3. The left plot shows the cross-entropy loss on the updated training data. The new model and target model, both trained on the updated data, achieve equally low loss. Because the new model is initialized by the old model (see Equation 5), it stays within the same loss basin. The target model, initialized by the pretrained model (see Equation 3), diverges more from the old model and ends up in a different local minimum. Thus interpolation between the old model and target model faces a high loss barrier and in turn low test accuracy. The distance between old model and target model is three times larger than the distance between old model and new model. According to Rame et al. (2022) this leads to a large locality term and makes the models less "averagable". A potential way to alleviate this is permutating the weights (Ainsworth et al., 2022) of the target model such that it lies within the same basin as the old model. The plot in the middle shows the accuracy along the interpolation from old to new model and that small α-values maintain high accuracy. The interpolation towards the target model traverses low accuracy regions and only achieves high accuracy very close to the target model. The plot on the right shows that the area of low negative flip rate is centered around the old model. This explains the lower NFR for the new model opposed to the target model because the distance between the old and new model is smaller than between the old and target model. Interpolating the weights of the new model and the weights of the old model follows a monotonic decrease of negative flips. This allows to find a point in weight space that has low negative flips while maintaining high accuracy.

8 Conclusion

We studied the problem of regression during data updates in text classification. Retraining a model with a larger amount of training data increases accuracy but also introduces negative flips. We propose BCWI which describes the interpolation between the weights of the old model and the weight of the new model. We empirically show on three datasets and two update scenarios that BCWI models significantly reduce negative flips while not sacrificing accuracy. We compare BCWI to strong continual learning methods and achieve similar or better results, while not increasing training or inference cost. Another big advantage of BCWI is that the trade-off parameter α can be tuned without retrain-

ing the model. This saves additional training cost and only requires to store the weights of the old and new model. We extend BCWI by using the Fisher information matrix as importance factor in weight interpolation and show that it further reduces negative flips. Using multiple new models as in proposed SoupBCWI also reduces regression without increasing the inference cost. In principle BCWI is architecture and task agnostic with the possibility to explore effectiveness in applications such as image classification or natural language generation left for future work.

Limitations

We show the effectiveness of our method on three datasets, two of which are focused on intent detection. While in principle the method is task-agnostic, we didn't present results for more tasks or domains. Another limitation is that we did not show results for BCWI when the training data is updated multiple times and the new model is interpolated successively.

Ethics Statement

The proposed method relies on pretrained models and inherits their possible harmful biases. Given that the objective of BCWI is to change correct predictions as little as possible, it solidifies possible harmful predictions and biases of a finetuned model. Consequently, practitioners must scrutinize the annotated class labels of datasets that are used in combination with the BCWI method.

References

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha S. Srinivasa. 2022. Git re-basin: Merging models modulo permutation symmetries. *ArXiv*, abs/2209.04836.

Gagan Bansal, Besmira Nushi, Ece Kamar, Dan Weld, Walter Lasecki, and Eric Horvitz. 2019. Updates in human-ai teams: Understanding and addressing the performance/compatibility tradeoff. In *AAAI Conference on Artificial Intelligence*. AAAI.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Deng Cai, Elman Mansimov, Yi-An Lai, Yixuan Su, Lei Shu, and Yi Zhang. 2022. Measuring and reducing model update regression in structured prediction for nlp. *ArXiv*, abs/2202.02976.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Xiang Deng, Yun Xiao, Bo Long, and Zhongfei Zhang. 2022. Reducing flipping errors in deep neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(6):6506–6514.

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*, pages 4171–4186, Minneapolis, Minnesota.

Jack G. M. FitzGerald, Christopher Leo Hench, Charith S. Peris, Scott Mackie, Kay Rottmann, A. Patricia Domínguez Sánchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, L. Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and P. Natarajan. 2022. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages. *ArXiv*, abs/2204.08582.

Christopher Hidey, Fei Liu, and Rahul Goel. 2022. Reducing model churn: Stable re-training of conversational agents. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 14–25, Edinburgh, UK. Association for Computational Linguistics.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. 2022. Patching open-vocabulary models by interpolating weights. *ArXiv*, abs/2208.05592.

P Izmailov, AG Wilson, D Podoprikhin, D Vetrov, and T Garipov. 2018. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pages 876–885.

Heinrich Jiang, Harikrishna Narasimhan, Dara Bahri, Andrew Cotter, and Afshin Rostamizadeh. 2022. Churn reduction via distillation. In *International Conference on Learning Representations*.

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.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. 2022. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:3366–3385.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. *ArXiv*, abs/1909.11299.

Haokun Liu, Derek Tam, Muqeeth Mohammed, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022a. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*.

Huiting Liu, Avinesh P.V.S, Siddharth Patwardhan, Peter Grasch, and Sachin Agarwal. 2022b. Model stability with continuous data updates.

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

Michael Matena and Colin Raffel. 2021. Merging models with fisher-weighted averaging. *ArXiv*, abs/2111.09832.

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

Mahdi Milani Fard, Quentin Cormier, Kevin Canini, and Maya Gupta. 2016. Launch and iterate: Reducing prediction churn. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning {bert}: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

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.

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, patrick gallinari, and Matthieu Cord. 2022. Diverse weight averaging for out-of-distribution generalization. In *Advances in Neural Information Processing Systems*.

Roger Ratcliff. 1990. Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*, 97(2):285–308.

Yantao Shen, Yuanjun Xiong, Wei Xia, and Stefano Soatto. 2020. Towards backward-compatible representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6367–6376.

Shagun Sodhani, Mojtaba Farmazi, Sanket Vaibhav Mehta, Pranshu Malviya, Mohamed Abdelsalam, J. Janarthanan, and Sarath Chandar. 2022. An introduction to lifelong supervised learning. *ArXiv*, abs/2207.04354.

Megha Srivastava, Besmira Nushi, Ece Kamar, Shital Shah, and Eric Horvitz. 2020. An empirical analysis of backward compatibility in machine learning systems. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '20, page 3272–3280, New York, NY, USA. Association for Computing Machinery.

Mariya Toneva, Alessandro Sordoni, Rémi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. 2018. An empirical study of example forgetting during deep neural network learning. *ArXiv*, abs/1812.05159.

Georg Wiese, Dirk Weissenborn, and Mariana Neves. 2017. Neural domain adaptation for biomedical question answering. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 281–289, Vancouver, Canada. Association for Computational Linguistics.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022a. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Mitchell Wortsman, Gabriel Ilharco, Mike Li, Jong Wook Kim, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. 2022b. Robust fine-tuning of zero-shot models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7949–7961.

Yuqing Xie, Yi-An Lai, Yuanjun Xiong, Yi Zhang, and Stefano Soatto. 2021. Regression bugs are in your model! measuring, reducing and analyzing regressions in NLP model updates. 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 6589–6602, Online. Association for Computational Linguistics.

Sijie Yan, Yuanjun Xiong, Kaustav Kundu, Shuo Yang, Siqi Deng, Meng Wang, Wei Xia, and Stefano Soatto. 2021. Positive-congruent training: Towards regression-free model updates. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14294–14303.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Yue Zhao, Yantao Shen, Yuanjun Xiong, Shuo Yang, Wei Xia, Zhuowen Tu, Bernt Shiele, and Stefan 0 Soatto. 2022. Elodi: Ensemble logit difference inhibition for positive-congruent training. *ArXiv*, abs/2205.06265.

A Experiment Details

	(Range of) Hyperparameters		
Prior WD	0.01, 0.1, 1.0, 10.0, 100, 200, 1e3, 2e3, 4e3, 1e4, 1e5		
Mixout	0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.98, 0.99, 0.999		
EWC	1e-5, 1e-4, 1e-3, 0.01, 0.1, 1.0, 2.0, 5.0, 10.0, 50.0 100, 1e3, 1e4		
BitFit & IA^3	E: 8, 12, 16; LR: 1e-4, 1e-3, 1e-2		
LR Schedule	linear		
Warmup Ratio	0.1		
Batch Size	16		
Adam ϵ	1e-6		
Adam β_1	0.9		
Adam β_2	0.98		
Adam Bias Corr.	True		
Dropout	0.1		
Weight Decay	0.01		
Clip grad. norm	5.0		
	MASSIVE	**Banking77**	**AG News**
Old Model:			
Epochs	16	16	8
Learning Rate	6e-5	6e-5	6e-5
Target Model:			
Epochs	16	16	8
Learning Rate	6e-5	6e-5	6e-5
New Model:			
Epochs	3, 6, 10	3, 6, 10	2, 3, 6
Learning Rate	3e-5, 6e-5	3e-5, 6e-5	3e-5, 6e-5

Table 6: Hyperparameter for the different datasets and methods.

We list the hyperparameters used for training the different models in Table 6. The selection of hyperparameter largely follows (Mosbach et al., 2021). We use the RoBERTa$_{\text{BASE}}$ model from HuggingFace[2]. The best learning rate and number of epochs is selected on the development set based on accuracy and NFR. Although there are no extensive experiments, we noticed that BCWI is largely insensitive to hyperparameter selection. The focus can remain on optimizing accuracy and BCWI handles regression after the successful training. The interpolation parameter α is tuned on the development set by choosing the largest α-value that does not cause the accuracy to drop below a chosen threshold (see Table Table 9 and 10). The regularization strength of the baselines is tuned in the same way by selecting the strongest regularization parameter that does not sacrifice accuracy below that threshold on the dev set. We use a V100 GPU and finetuning takes around 20 minutes.

	MASSIVE		**Banking77**		**AG News**	
Model	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**
Old Model	81.8 ±0.2	0.0 ±0.0	82.8 ±0.4	0.0 ±0.0	85.0 ±0.8	0.0 ±0.0
Target Model	83.4 ±0.4	3.3 ±0.4	86.2 ±0.4	3.0 ±0.3	88.0 ±0.1	3.4 ±0.3
New Model	83.2 ±0.2	2.8 ±0.2	86.3 ±0.1	1.6 ±0.1	88.3 ±0.3	2.4 ±0.3
Ensemble-2	83.8 ±0.3	2.2 ±0.2	86.4 ±0.2	1.4 ±0.2	88.4 ±0.2	2.4 ±0.4
Ensemble-4	84.0 ±0.2	2.0 ±0.1	86.5 ±0.2	1.4 ±0.2	88.6 ±0.2	2.3 ±0.3
Ensemble-8	84.2 ±0.2	1.8 ±0.1	86.5 ±0.2	1.3 ±0.2	88.7 ±0.2	2.2 ±0.3
Ensemble-16	84.3 ±0.2	1.7 ±0.1	86.4 ±0.2	1.3 ±0.2	88.8 ±0.2	2.1 ±0.2
Soup-2	83.7 ±0.3	2.2 ±0.2	86.3 ±0.2	1.4 ±0.2	88.5 ±0.2	2.3 ±0.4
Soup-4	83.9 ±0.3	1.9 ±0.1	86.4 ±0.2	1.4 ±0.1	88.6 ±0.3	2.2 ±0.3
Soup-8	84.0 ±0.2	1.8 ±0.1	86.4 ±0.2	1.3 ±0.2	88.8 ±0.2	2.2 ±0.3
Soup-16	84.1 ±0.2	1.7 ±0.1	86.3 ±0.2	1.3 ±0.1	88.9 ±0.2	2.1 ±0.2

Table 7: Results for the Add_Data scenario on the test set. *Ensemble-M* is the output ensemble of *M* new models formed by averaging the probabilities. *Soup-M* is the soup ensemble of *M* new models formed by averaging the model weights.

	MASSIVE		**Banking77**		**AG News**	
Model	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**
Old Model	68.8 ±0.1	0.0 ±0.0	80.0 ±0.3	0.0 ±0.0	70.4 ±0.1	0.0 ±0.0
Target Model	83.9 ±0.2	3.2 ±0.2	86.5 ±0.3	2.8 ±0.3	87.9 ±0.5	3.5 ±0.6
New Model	83.8 ±0.2	2.4 ±0.1	86.3 ±0.3	1.8 ±0.2	87.9 ±0.3	4.2 ±0.5
Ensemble-2	83.9 ±0.2	2.2 ±0.1	86.8 ±0.3	1.4 ±0.2	88.0 ±0.3	4.2 ±0.4
Ensemble-4	84.2 ±0.2	2.0 ±0.1	86.9 ±0.2	1.2 ±0.1	88.0 ±0.3	4.3 ±0.4
Ensemble-8	84.2 ±0.2	1.9 ±0.2	87.0 ±0.3	1.1 ±0.1	88.0 ±0.3	4.2 ±0.3
Ensemble-16	84.3 ±0.2	1.9 ±0.2	87.1 ±0.3	1.0 ±0.1	88.1 ±0.3	4.2 ±0.3
Soup-2	83.9 ±0.3	2.1 ±0.1	86.7 ±0.3	1.4 ±0.2	88.0 ±0.3	4.2 ±0.3
Soup-4	84.0 ±0.2	1.9 ±0.1	86.8 ±0.3	1.1 ±0.1	88.0 ±0.3	4.2 ±0.4
Soup-8	84.1 ±0.2	1.8 ±0.2	86.9 ±0.2	1.0 ±0.1	88.1 ±0.3	4.2 ±0.4
Soup-16	84.1 ±0.2	1.8 ±0.2	86.9 ±0.2	1.0 ±0.1	88.1 ±0.3	4.1 ±0.4

Table 8: Results for the Add_Classes scenario on the test set of the three datasets. *Ensemble-M* is the output ensemble of *M* new models formed by averaging the probabilities. *Soup-M* is the soup ensemble of *M* new models formed by averaging the model weights.

B Additional Results

In Table 9 and 10 we show the dev set results for Table 2 and 3. The hyperparameter for the respective method was tuned to reach the accuracy threshold on the dev set.

We present the plots for FisherBCWI results in Figure 6. Results for Soup ensembles and probability ensembles of new models are listed in Table 7 and 8. They achieve the same accuracy and NFR which means that soup ensembles are as good as probability ensembles in reducing regression without increasing inference cost.

In the analysis in Section 7, we show that trajectory of BCWI closely follows the probability ensemble of old and new model. In Figure 5. In

[2] https://huggingface.co/roberta-base

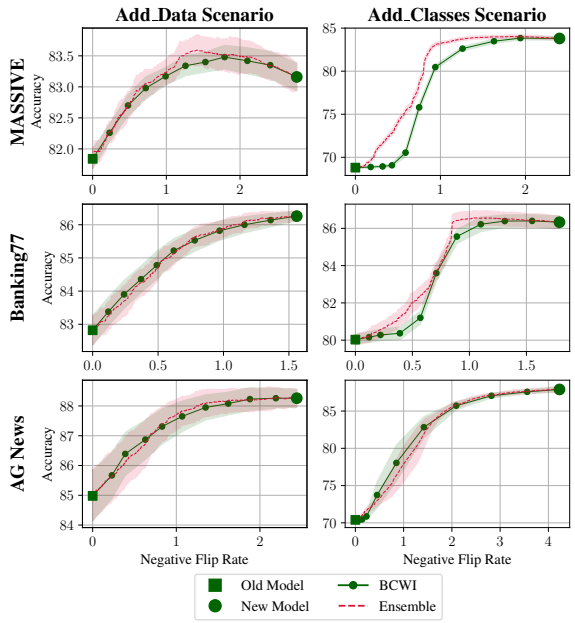

Figure 5: Plots comparing BCWI with the probability ensemble of old and new model.

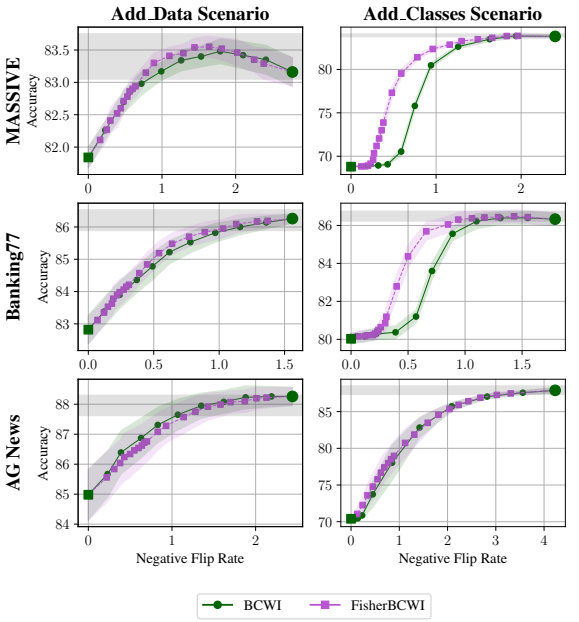

Figure 6: Plots for FisherBCWI in comparison with vanilla BCWI. The gray area indicates the target accuracy level.

the AC scenario the probabilities for new classes predicted by the old model are set to zero, because it was only trained on the old classes.

C Access to Old Data

For our main experiments we assume full access to the old data. This allows us to train the new model without catastrophic forgetting. To complement these results, we also show the behavior of BCWI when the new model is trained only on the new data (i.e. no access to the old data). The results are presented in Figure 8 and show that for the AD scenario the more restrictive setting has negative impact on Banking77 but achieves similar results for MASSIVE and AG News. In the AC scenario, the new model has significantly lower accuracy which can be attributed to catastrophic forgetting, because the new model is finetuned on new classes only. The interpolation towards the old model improves accuracy but does not reach the same accuracy as finetuning the new model on old and new data.

D Datasets and Scenarios

Detailed label distribution and number of instances for the AD and AC scenarios for all three datasets are visualized in Figure 9. The plots also show which classes are added for each dataset in the AC scenario.

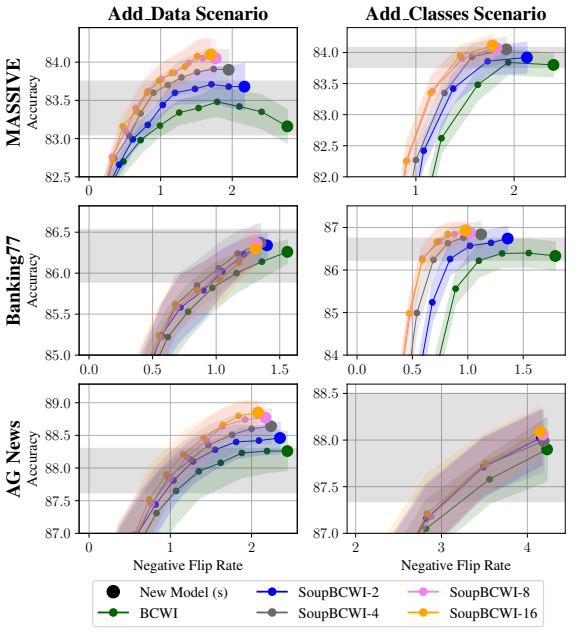

Figure 7: Plots for SoupBCWI in comparison with vanilla BCWI. The gray area indicates the target accuracy level.

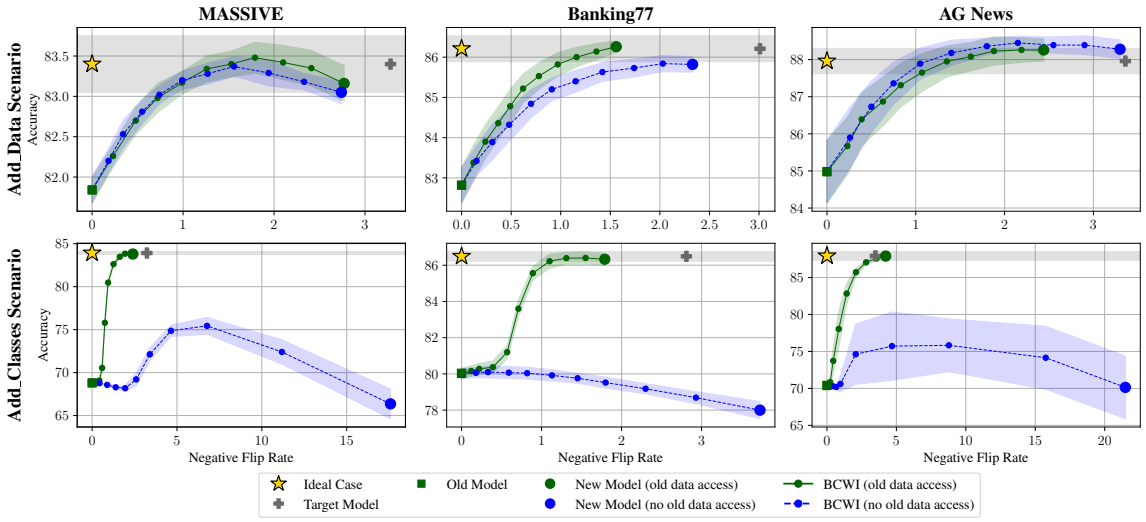

Figure 8: Comparison of BCWI with and without access to the old data during training of the new model. Results for the AD and AC scenario evaluated on our test sets of MASSIVE (FitzGerald et al., 2022), Banking77 (Casanueva et al., 2020) and AG News (Zhang et al., 2015). The gray horizontal bar is spanned by the 95% confidence interval of the target model and indicates the level of accuracy a model should reach. The α-values are in 0.1 steps where 0.0 is equivalent to the new model and 1.0 is equivalent to the old model. The ideal case for a model is to have zero negative flips while maintaining the target accuracy.

	MASSIVE					Banking77					AG News				
	λ	dev		test		λ	dev		test		λ	dev		test	
Model		**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**		**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**		**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**
Old Model	-	80.4 ±0.8	0.0 ±0.0	81.8 ±0.2	0.0 ±0.0	-	83.2 ±0.9	0.0 ±0.0	82.8 ±0.4	0.0 ±0.0	-	84.1 ±1.5	0.0 ±0.0	85.0 ±0.8	0.0 ±0.0
Target Model	-	82.2 ±0.4	3.0 ±0.6	83.4 ±0.4	3.3 ±0.4	-	86.4 ±0.7	2.8 ±0.8	86.2 ±0.4	3.0 ±0.3	-	88.5 ±1.1	3.1 ±0.9	88.0 ±0.1	3.4 ±0.3
New Model	-	82.0 ±1.0	2.4 ±0.5	83.2 ±0.2	2.8 ±0.2	-	86.1 ±0.8	**1.1** ±0.3	86.3 ±0.1	1.6 ±0.1	-	89.5 ±0.8	1.3 ±0.3	88.3 ±0.3	2.4 ±0.3
ACC Threshold		$\geq$ 81.8					$\geq$ 85.8					$\geq$ 89.0			
PriorWD	100	81.8 ±0.7	**1.7** ±0.3	83.4 ±0.3	2.0 ±0.2	200	86.1 ±0.7	**0.8** ±0.3	85.9 ±0.3	1.3 ±0.1	1e3	89.5 ±0.8	**0.8** ±0.5	88.1 ±0.4	**1.7** ±0.2
Mixout	0.2	81.8 ±0.5	2.2 ±0.4	83.0 ±0.2	2.6 ±0.2	0.9	86.1 ±0.7	**0.9** ±0.2	85.8 ±0.3	1.4 ±0.1	0.95	89.7 ±0.9	**0.9** ±0.6	88.4 ±0.4	**1.6** ±0.2
EWC	0.01	82.0 ±0.8	**1.8** ±0.4	83.3 ±0.3	2.1 ±0.2	0.01	86.4 ±1.0	**0.8** ±0.2	86.1 ±0.2	1.4 ±0.1	1.0	88.9 ±1.0	**0.9** ±0.6	87.9 ±0.4	**1.6** ±0.3
BCWI	0.45	81.8 ±0.7	**1.2** ±0.3	83.4 ±0.1	**1.4** ±0.1	0.4	85.8 ±0.8	**0.6** ±0.2	85.5 ±0.3	**0.8** ±0.1	0.35	89.0 ±0.8	**0.8** ±0.4	88.0 ±0.4	**1.5** ±0.2
FisherBCWI	0.2	81.9 ±0.8	**1.8** ±0.3	83.5 ±0.2	2.0 ±0.2	0.6	85.8 ±0.9	**0.6** ±0.3	85.5 ±0.3	**0.6** ±0.1	0.2	89.2 ±0.8	**1.0** ±0.4	88.1 ±0.4	**1.9** ±0.3
SoupBCWI-2	0.45	81.8 ±0.7	1.0 ±0.4	83.5 ±0.1	1.1 ±0.1	0.4	85.8 ±1.0	0.6 ±0.3	85.6 ±0.4	0.7 ±0.1	0.45	89.1 ±0.6	0.7 ±0.4	88.0 ±0.4	1.2 ±0.2
SoupBCWI-4	0.5	81.9 ±0.4	0.8 ±0.2	83.6 ±0.1	0.9 ±0.1	0.45	85.8 ±0.9	0.5 ±0.2	85.4 ±0.4	0.6 ±0.1	0.45	89.1 ±0.8	0.6 ±0.4	88.0 ±0.3	1.1 ±0.1
SoupBCWI-8	0.5	81.9 ±0.4	0.8 ±0.3	83.6 ±0.2	0.8 ±0.1	0.45	85.8 ±1.0	0.5 ±0.2	85.4* ±0.3	0.6 ±0.1	0.45	89.1 ±0.7	0.7 ±0.4	88.0 ±0.3	1.1 ±0.1
SoupBCWI-16	0.55	81.8 ±0.5	0.6 ±0.2	83.5 ±0.2	0.7 ±0.1	0.45	85.9 ±0.9	0.5 ±0.2	85.4 ±0.4	0.6 ±0.1	0.5	89.1 ±0.8	0.6 ±0.4	87.9 ±0.4	0.9 ±0.1

Table 9: Results for the Add_Data scenario. The trade-off parameter λ (or α for BCWI) is tuned on the dev set to be above the accuracy threshold. The threshold is set as 90% of dev accuracy from old to new model. "*" indicates that the accuracy does not overlap with accuracy of the target model. **Bold** NFR values have overlapping 95% confidence intervals with the best value. The old model and SoupBCWI is not under consideration when selecting the best NFR value.

	MASSIVE					Banking77					AG News				
	λ	dev		test		λ	dev		test		λ	dev		test	
Model		**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**		**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**		**ACC↑**	**NFR↓**	**ACC↑**	**NFR↓**
Old Model	-	67.1 ±0.5	0.0 ±0.0	68.8 ±0.1	0.0 ±0.0	-	82.9 ±0.7	0.0 ±0.0	80.0 ±0.3	0.0 ±0.0	-	67.9 ±0.5	0.0 ±0.0	70.4 ±0.1	0.0 ±0.0
Target Model	-	81.6 ±0.6	3.9 ±0.5	83.9 ±0.2	3.2 ±0.2	-	89.0 ±0.6	2.2 ±0.4	86.5 ±0.3	2.8 ±0.2	-	86.8 ±1.5	**1.5** ±0.7	87.9 ±0.5	**3.5** ±0.6
New Model	-	81.5 ±0.6	3.0 ±0.3	83.8 ±0.2	2.4 ±0.1	-	88.6 ±0.5	1.7 ±0.6	86.3 ±0.3	1.8 ±0.2	-	87.9 ±0.6	**1.3** ±0.8	87.9 ±0.3	**4.2** ±0.5
ACC Threshold		$\geq$ 80.8					$\geq$ 88.3					$\geq$ 86.9			
PriorWD	200	81.3 ±0.7	2.1 ±0.3	83.3 ±0.3	2.1 ±0.2	200	89.4 ±0.7	**0.6** ±0.2	86.3 ±0.3	1.1 ±0.1	1e4	87.5 ±1.4	**1.2** ±0.6	87.4 ±0.4	4.3 ±0.4
Mixout	0.7	81.0 ±0.5	2.6 ±0.3	83.0 ±0.2	2.4 ±0.1	0.95	89.0 ±0.8	**0.9** ±0.4	86.2 ±0.3	1.2 ±0.1	0.8	88.0 ±1.1	**1.9** ±0.9	87.6 ±0.4	5.0 ±0.5
EWC	0.01	81.6 ±0.5	2.2 ±0.3	83.6 ±0.3	2.0 ±0.1	0.01	89.3 ±0.7	**0.6** ±0.2	86.4 ±0.3	**0.9** ±0.1	1e-5	88.0 ±0.6	**1.3** ±0.8	87.9 ±0.4	4.3 ±0.4
BCWI	0.25	81.2 ±0.5	**1.7** ±0.3	83.2 ±0.2	**1.4** ±0.1	0.35	88.8 ±0.5	**0.8** ±0.3	86.0 ±0.4	1.0 ±0.1	0.1	86.9 ±0.6	**1.1** ±0.6	87.6 ±0.3	**3.6** ±0.4
FisherBCWI	0.5	81.3 ±0.5	**1.2** ±0.2	82.9 ±0.2	**1.2** ±0.1	0.7	88.5 ±0.5	**0.6** ±0.2	85.7 ±0.5	**0.7** ±0.1	0.05	86.9 ±0.7	**1.0** ±0.6	87.5 ±0.2	**3.3** ±0.4
SoupBCWI-2	0.25	81.3 ±0.6	1.3 ±0.2	83.0 ±0.3	1.2 ±0.1	0.35	88.6 ±0.8	0.8 ±0.4	85.8 ±0.4	0.8 ±0.1	0.05	87.3 ±0.8	1.3 ±0.9	87.9 ±0.3	3.8 ±0.3
SoupBCWI-4	0.25	81.3 ±0.5	1.2 ±0.1	82.9 ±0.2	1.1 ±0.1	0.35	88.3 ±0.9	0.8 ±0.4	85.8 ±0.3	0.6 ±0.1	0.05	87.4 ±0.9	1.5 ±0.9	87.9 ±0.3	3.8 ±0.4
SoupBCWI-8	0.25	81.3 ±0.5	1.2 ±0.2	82.9 ±0.3	1.0 ±0.1	0.35	88.6 ±0.7	0.7 ±0.3	85.8 ±0.3	0.5 ±0.1	0.1	86.9 ±0.8	1.3 ±0.7	87.7 ±0.3	3.5 ±0.3
SoupBCWI-16	0.25	81.3 ±0.5	1.1 ±0.2	82.9 ±0.2	1.0 ±0.1	0.35	88.5 ±0.9	0.5 ±0.3	85.8 ±0.3	0.5 ±0.1	0.05	87.7 ±0.8	1.3 ±0.7	87.9 ±0.3	3.8 ±0.3

Table 10: Results for the Add_Classes scenario. The trade-off parameter λ (or α for BCWI) is tuned on the dev set to be above the accuracy threshold. The threshold is set as 95% of dev accuracy from old to new model. "*" indicates that the target accuracy on the test set is not reached. **Bold** NFR values have overlapping 95% confidence intervals with the best value. The old model and SoupBCWI is not under consideration when selecting the best NFR value.

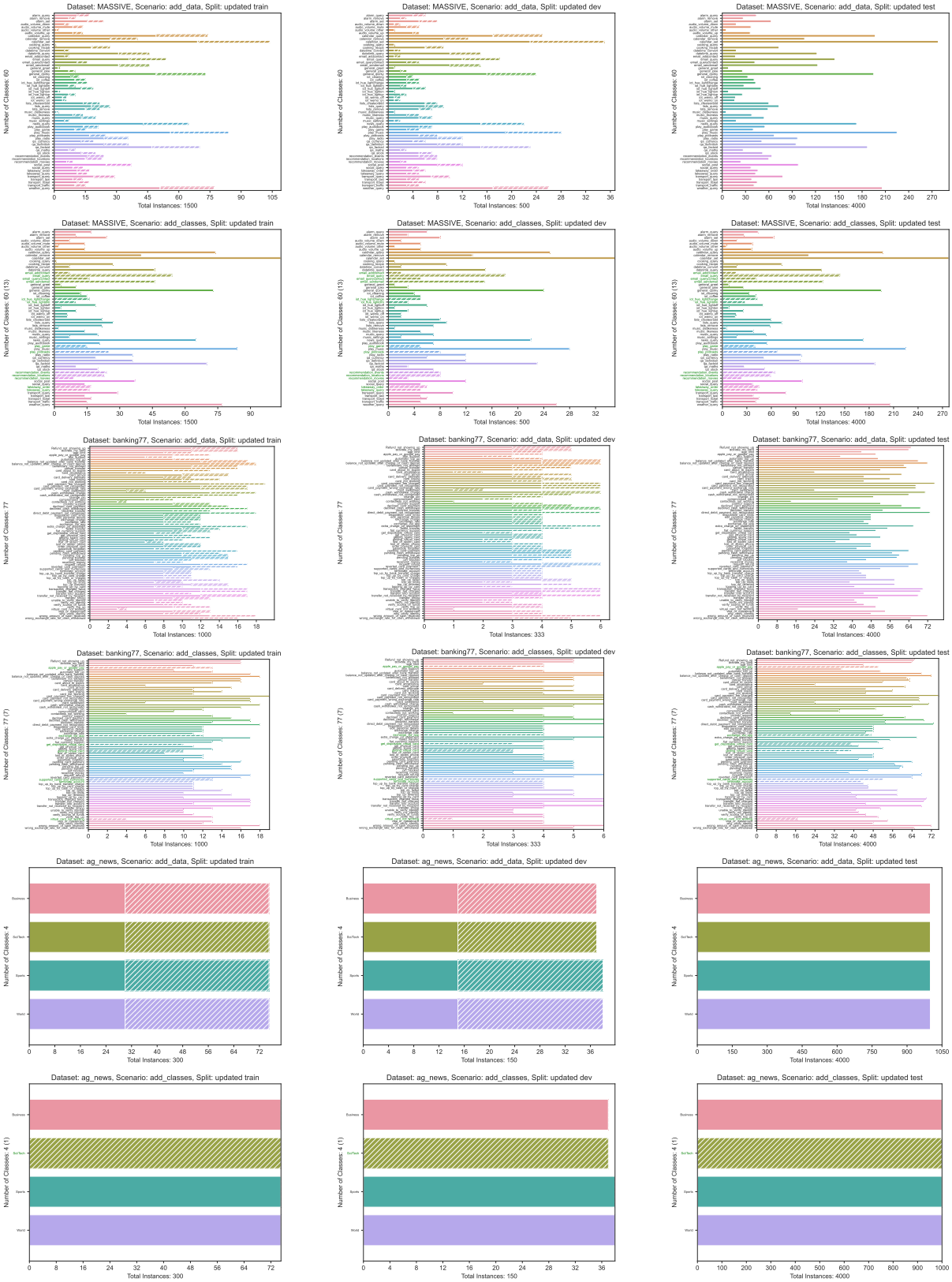

Figure 9: Add_Data scenario and Add_Classes scenario for MASSIVE (FitzGerald et al., 2022), Banking77 (Casanueva et al., 2020) and AG News (Zhang et al., 2015). Striped bars indicate added instances. Added class names are printed in green.