Learning Without Forgetting: Online Adaptation of Transformer-based NERC Models

Pablo Turón[1,*], Montse Cuadros[1]

[1]Fundación Vicomtech, Basque Research and Technology Alliance (BRTA), Mikeletegi 57, 20009 Donostia-San Sebastián (Spain)

Abstract

Transformer-based Named Entity Recognition and Classification (NERC) models are typically trained under static scenarios, making them vulnerable to concept drift as data evolves. While retraining can mitigate this issue, it is often computationally expensive, and updating only with new data risks catastrophic forgetting. To balance adaptation and retention, Online Learning (OL) enables continuous, incremental updates. In this work, we apply OL strategies to transformer-based NERC models on different datasets, exploring different update sizes and methods such as regularization, Low-Rank Adaptation (LoRA), Elastic Weight Consolidation (EWC), and replay. The results indicate that while OL facilitates adaptation to evolving data, all tested update strategies still exhibit a trade-off between adaptation and forgetting.

Keywords

Online Learning, Transformers, NERC

1. Introduction

Artificial intelligence (AI) models tend to lose performance over time due to data drift, which occurs when the distribution of incoming data gradually diverges from the original training dataset. To maintain performance, models must be adapted to the new data, either by fully retraining on the combined dataset or by incrementally updating with recent samples. Full retraining is computationally expensive and requires careful balancing between old and new data to ensure proper weighting. Incremental updates, meanwhile, can lead to catastrophic forgetting, where the model fits new data and rapidly loses knowledge from the original training. These challenges highlight the difficulty of keeping AI models accurate and up to date in dynamic environments.

To address the challenges of data drift and catastrophic forgetting, Online Learning (OL) provides a framework in which AI models are updated continuously as new data arrives, rather than relying on full retraining. In this paradigm, models integrate new information incrementally, allowing them to adapt to evolving data distributions while retaining previously acquired knowledge. Unlike other approaches such as continual learning, OL does not assume explicit task boundaries, task transitions, or periodic retraining over accumulated data, but instead performs immediate updates as data arrives. Compared with full retraining, OL is computationally more efficient and naturally balances old and new knowledge, making it particularly suitable for dynamic environments.

Within this framework, a variety of strategies have been developed to guide model updates and mitigate knowledge loss. Some approaches focus on parameter tuning and regularization, updating model weights while constraining significant deviations from previously learned representations. Techniques such as Elastic Weight Consolidation (EWC) [1] aim to preserve prior knowledge by penalizing changes to important parameters. Other methods increase model capacity; for example, Low-Rank Adaptation (LoRA) [2] introduces additional trainable parameters to learn from new data without substantially altering the original weights. Replay-based strategies maintain performance on earlier distributions by periodically revisiting a subset of previously observed samples.

Despite their effectiveness, each of these approaches presents inherent trade-offs and limitations. Their performance is highly sensitive to factors such as hyperparameter configuration, the volume of data available at each update step, and the specific implementation choices employed. While OL has been extensively studied in general machine learning settings, its potential for fine-tuning large transformer-based models remains relatively underexplored—especially for structured prediction tasks like Named Entity Recognition and Classification (NERC), a token-level sequence prediction problem with strong interdependencies between labels.

This gap is particularly important because transformer models, which have become the dominant paradigm in NERC, are typically fine-tuned in static settings under the assumption of a fixed training distribution. In practice, however, real-world NERC systems must contend with constantly evolving data: emerging entities, domain shifts, and subtle changes in language use can all erode model performance over time. Bridging the strengths of OL with transformers' contextual capabilities could therefore offer a promising path toward more adaptive and resilient NERC systems.

In this work, we investigate the application of OL strategies to the fine-tuning of transformer-based models for NERC tasks. Our goal is to evaluate whether OL can effectively adapt these models to dynamic data distributions while mitigating catastrophic forgetting and maintaining computational efficiency. We analyze different OL techniques within this context and assess their impact on model performance under continuous data updates.

This paper is organized as follows. Section 2 reviews the related work. Section 3 describes the datasets used in this study and how it was adapted for experimentation. Section 4 outlines the experimental framework, while Section 5 presents the conducted experiments and their results. Finally, Section 6 discusses the findings, followed by the conclusions in Section 7.

SEPLN 2026: 42nd International Conference of the Spanish Society for Natural Language Processing, León, Spain, 22-25 September 2026.

*Corresponding author.

pturon@vicomtech.org (P. Turón); mcuadros@vicomtech.org (M. Cuadros)

0000-0002-5563-1120 (P. Turón); 0000-0002-3620-1053 (M. Cuadros)

2. Related Work

NERC is a core task in natural language processing that identifies and classifies text spans into predefined entity categories. Modern approaches are predominantly based on transformer architectures, which provide strong contextual representations and have achieved state-of-the-art performance across benchmarks [3]. However, these models are typically trained under static data distributions, while real-world applications are characterized by evolving data, including domain shifts and emerging entities.

This mismatch motivates the need for methods capable of adapting to data drift while preserving previously learned knowledge. In this context, a key challenge is catastrophic forgetting, where updating a model with new data degrades its performance on earlier data [4]. OL addresses this issue by enabling incremental updates, aiming to balance adaptation to new data and retention of prior knowledge [5].

A common strategy to mitigate forgetting is based on replay, where a small subset of past data is stored and revisited during training [6, 7]. Variants such as reservoir sampling are often used to maintain a representative memory, and replay can be combined with distillation to further preserve previous behavior [8]. While effective, these methods introduce memory overhead.

Regularization-based approaches instead constrain model updates to preserve important parameters. EWC penalizes changes to weights deemed important for previous data using importance estimates such as the Fisher information [1, 9]. Distillation-based methods similarly encourage consistency with a previous model without requiring access to past data [10]. While these strategies are generally memory-efficient, they can encounter difficulties when faced with substantial distributional shifts between tasks.

Architecture-based methods modify the model's structure to incorporate new knowledge, typically by adding parameters while preserving the existing ones. Progressive architectures exemplify this approach but can result in continuous model growth over time [11]. A more parameter-efficient alternative is Low-Rank Adaptation (LoRA), which introduces a small set of trainable low-rank matrices while keeping the base model frozen, enabling effective adaptation without significant model expansion [2].

Although OL has been extensively studied, its use in transformers remains limited. Most transformer-based approaches are trained in offline settings, with few works exploring their adaptation to streaming data. Existing methods show that transformers can be incrementally updated using standard optimization techniques combined with replay mechanisms or architectural modifications [12, 13]. While several studies have explored continual learning for NERC [14], the applicability of online learning to structured prediction problems such as NERC remains largely underexplored.

Overall, existing OL approaches—replay, regularization, and architectural adaptation—offer complementary strategies to balance knowledge retention and adaptation, but have been predominantly validated on classification tasks. Their effectiveness in structured prediction settings such as transformer-based NERC remains underexplored, particularly with evolving label distributions and emerging entities. Consequently, it is unclear how well these methods transfer to streaming scenarios. This work addresses this gap by investigating representative OL strategies for transformer-based NERC under streaming conditions.

3. Dataset

To the best of our knowledge, there is currently no publicly available NERC dataset explicitly designed to capture concept drift. This limitation motivated us to compile and analyze a collection of existing NERC datasets suitable for our study. In particular, we gathered several publicly available Spanish-language medical datasets from shared tasks, including *eHealth-KD*, *LivingNER*, and *MedProcNER* [15, 16, 17], as well as the English *Legal-NER* dataset [18], chosen to introduce cross-domain variability. These datasets differ substantially in both size and types of annotated entities (see Table 1), providing a diverse foundation for evaluating our method. A detailed description of each dataset is presented in subsection 3.1.

Dataset	Train_{base}	Train_{OL}	Dev	Test
Medline_{EKD}	650	390	130	130
Wikinews_{EKD}	200	120	40	40
eHealth-KD_{ALL}	850	510	170	170
LivingNER	1,157	695	221	225
MedProcNER	511	306	100	102
Legal-NER	4,718	2,830	944	943

Table 1
Overview of NERC datasets used for the experiments

To enable our experiments, we recomputed the data splits for each dataset, partitioning them into four subsets: base fine-tuning *(Train_{base})*, OL updates *(Train_{OL})*, development, and test sets. The splits were constructed to ensure that all entities are represented in each subset. They follow a 50%, 30%, 10%, and 10% distribution, respectively, as illustrated in Figure 1, and are used within the experimental framework described in Section 4. Although this partitioning does not fully reflect a realistic scenario—since the Train_{OL} subset shares the same data distribution as Train_{base}—it allows us to systematically evaluate how model updates affect the initial training under different OL strategies.

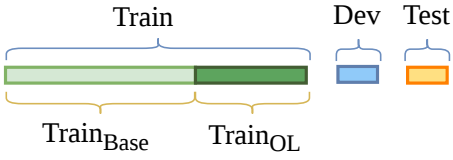

Figure 1: Data split representation.

3.1. Datasets Overview

The datasets used in this study cover Spanish medical texts and an English legal corpus, with varying sizes and levels of complexity. A detailed description of each dataset is provided below.

eHealth-KD is a Spanish-language corpus for extracting semantic information from electronic health records. It annotates four entities: *Concept*, *Action*, *Predicate*, and *Reference*, using MedlinePlus *(Medline_{EKD})* and Wikinews *(Wikinews_{EKD})* data sources, covering medical and general

health content. Experiments can use each source separately or combined *(eHealth-KD$_{ALL}$)*.

LivingNER focuses on the recognition, normalization, and categorization of species, pathogens, and food-related entities within clinical case reports drawn from over ten medical specialties. The NERC task targets the classification of entities as either *SPECIES* (non-human organisms) or *HUMAN*. Beyond entity recognition, this dataset also enables linking each identified entity to the *NCBI Taxonomy*, which facilitates both specialized entity identification and the classification of clinical impacts, such as foodborne or nosocomial infections.

MedProcNER is a dataset constructed from clinical texts, specifically intended for the identification of medical procedures. It includes clinical case reports across multiple medical specialties, including cardiology, oncology, and psychiatry. Each procedure mentioned in the texts is carefully annotated, normalized, and mapped to *SNOMED CT* terminology [19].

Legal-NER is an English corpus for named entity recognition in Indian court judgments, focusing on legal entities such as actors, institutions, and references. Although it includes annotations for both Judgements and Preambles, this study uses only the Judgement training set.

4. Experimental framework

As introduced in Section 3, the dataset is divided into four subsets: one for initial base fine-tuning *(Train$_{base}$)*, one for OL updates *(Train$_{OL}$)*, and separate sets for development and testing. Within this framework, we first fine-tune and evaluate a base transformer model on the target task using *Train$_{base}$*. We then iteratively update this model using an OL strategy applied to successive data samples from *Train$_{OL}$*, where each update step starts from the model obtained in the previous iteration, until the entire OL dataset has been processed, as shown in Figure 2.

At each iteration i, the model is first evaluated on the current data sample *Train$_{OL_i}$*. It is then fine-tuned on this subset—hereafter referred to as an OL update—using the selected OL algorithm, and subsequently re-evaluated on *Train$_{OL_i}$*, as well as on *Train$_{base}$*, and the test set. This procedure allows us to assess whether the model exhibits forgetting of the original data and whether it successfully adapts to newly introduced data.

To represent various OL scenarios, we evaluate different update granularities by splitting the OL data into chunks of different sizes: (1) extreme OL, where the model is updated after each sample, (2) moderate OL, where updates are performed every 5 samples, and (3) conservative OL, where updates are performed in batches of 50 samples.

For the base fine-tuning, we used `bert-base-multilingual-cased` as our foundation, chosen for its ability to handle multiple languages across the datasets in our experiments. For each dataset, the model was fine-tuned independently for 50 epochs—sufficient to flatten the training loss and reach a stable state—with early stopping patience of 5. Training was conducted with a batch size of 8, a learning rate of 10^{-5} and two warmup epochs using *fp16*.

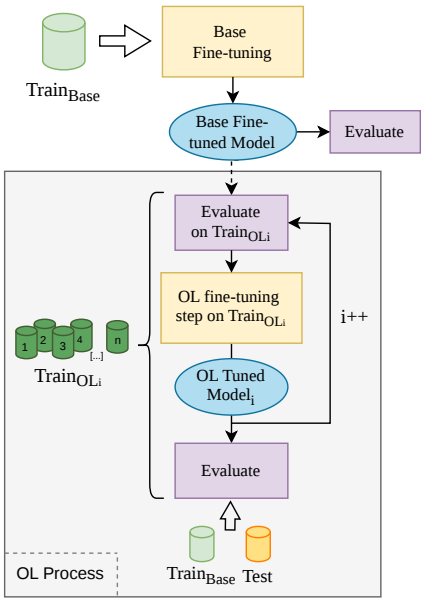

Figure 2: Online Learning workflow by experiment.

The deployed system supports three OL algorithms and one replay strategy. The algorithms are:

Regularization. Simple regularization by standard fine-tuning with tunable hyperparameters such as learning rate, number of epochs, regularization strength, and optional layer freezing.

Low-Rank Adaptation (LoRA). LoRA fine-tunes a pre-trained model by freezing its original weights and learning low-rank updates $\Delta W = \frac{\alpha}{r} AB$, with $r \ll d$, where $A \in \mathbb{R}^{d \times r}$, $B \in \mathbb{R}^{r \times d}$, and d is the weight dimension, typically applied to attention projections. The rank r controls the adaptation capacity, and α scales the magnitude of the update. By training only a small number of parameters, LoRA enables memory-efficient fine-tuning while preserving the model's pre-existing knowledge, allowing it to adapt effectively to new tasks.

Elastic Weight Consolidation (EWC). fine-tunes a model by adding a regularization term that penalizes changes to weights important for previous tasks:

$$\mathcal{L}_{EWC} = \mathcal{L}_{new} + \frac{\lambda}{2} \sum_i F_i (\theta_i - \theta_i^*)^2,$$

where θ_i^* are previous task weights, F_i their importance (Fisher information), and λ controls regularization strength. This approach mitigates catastrophic forgetting in OL by constraining updates to critical parameters while allowing incremental adaptation.

Replay strategies. To mitigate catastrophic forgetting, we incorporate replay mechanisms during the OL updates. At each iteration i, the model is fine-tuned not only on the current sample or batch *Train$_{OL_i}$* but also on a selected subset of the original training data *Train$_{base}$*. These samples are chosen using the Active Learning *N-grams-based strategy* proposed in [20], which prioritizes the most informative instances for replay, and constitute a percentage of the *Train$_{base}$* dataset, denoted as $R\%$.

Tech.	R%	Conf	lr	Forgetting Rate (ΔF1/upd.)			Impact (% data)			$\Delta F1$
				Test	Train_{base}	OL	Impr.	Wors.	Equal	
Reg.	0	Baseline	10^{-5}	-0,0091	-0,01	0,1286	38,47	6,08	55,45	5,35
EWC	0	γ=0.8; λ=40	10^{-5}	-0,0166	-0,0021	0,1244	40,08	2,74	57,17	5,66
	0	γ=0.8; λ=50	10^{-5}	-0,0187	-0,0079	0,1311	41,06	3,48	55,44	5,83
	0	γ=0.99; λ=150	10^{-5}	-0,0079	-0,0056	0,1242	39,50	2,97	57,51	5,30
	10	γ=0.8; λ=40	10^{-5}	-0,0827	-0,0555	0,0196	44,17	3,81	52,03	6,96
	10	γ=0.8; λ=50	10^{-5}	**0,0203**	-0,0323	0,0888	45,80	4,18	50,02	6,43
	10	γ=0.99; λ=150	10^{-5}	-0,0084	-0,0188	0,0798	45,14	2,58	52,28	7,04
LoRa	0	r=4; α=8	10^{-3}	-0,1606	-0,141	**0,183**	43,46	15,20	41,34	5,80
	0	r=8; α=16	10^{-3}	-0,2056	-0,304	-1,3531	30,54	19,66	49,80	1,53
	5	r=4; α=8	10^{-3}	0,0087	-0,0477	0,1108	**48,25**	8,87	42,88	**8,19**
	5	r=8; α=16	10^{-3}	-0,1342	-0,1518	0,0896	47,08	13,99	38,92	6,28
	10	r=4; α=8	10^{-3}	-0,0133	-0,0495	0,1245	46,20	12,03	41,75	6,34
	10	r=8; α=16	10^{-3}	-0,1201	-0,174	-1,4722	45,82	15,67	38,49	5,06
Regul.	0	Fr. all but last 4	10^{-5}	-0,0406	-0,0085	0,1376	36,45	4,99	58,56	4,07
	0	Freeze all; 5 Ep.	10^{-2}	0,0108	**0,0153**	0,182	8,45	6,02	85,53	0,47
	5	No freeze	10^{-5}	0,0045	-0,0048	0,0164	42,87	3,66	53,47	6,33
	10	No freeze	10^{-5}	-0,0179	-0,0518	-0,0026	44,56	**2,31**	53,13	7,60

Table 2
Average results of experiments using an update size of one sample. Bold values indicate the best performance within each experiment, while underlined values indicate the best performance achieved by each technique.

5. Experiments and results

For this experimentation, we conducted multiple runs across all datasets using different configurations of each OL algorithm. In total, 135,903 OL updates were performed, spanning 135 experimental configurations (including all OL methods with different settings across datasets for each update size). To derive meaningful insights, we report average results per experiment configuration across datasets. For clarity and readability, we present only a subset of the most relevant results in Tables 2, 3, and 4, as including all tables would make the reporting excessively lengthy.

In all experiments, we start from the base fine-tuned model and sequentially apply small fine-tuning updates using the selected OL chunk size until all $\textit{Train}_{OL}$ data has been processed. Each update is trained for 10 epochs, which provides sufficient convergence and ensures stability across all datasets and OL update sizes evaluated.

The evaluated metrics include forgetting, quantified as the average change in F1 score per dataset before and after each update ($\Delta F1$/update). We also assess the effect of OL updates on OL samples by measuring the average post-update F1 improvement ($\Delta F1$). Additionally, we examine the proportion of samples whose F1 score improves (termed effectiveness), remains unchanged, or decreases after the update, reporting these as fractions of the total samples in each category. All metrics are computed across multiple data splits—the $\textit{Train}_{base}$ dataset, the test dataset, and previously seen OL samples—allowing us to evaluate how rapidly each OL algorithm forgets prior knowledge or gains performance on both old and newly introduced data over the course of successive updates.

As a baseline, we conduct a single experiment in which OL updates are applied using the same training configuration as the base fine-tuned model. For the EWC method, we report results for three configurations, all using a learning rate of 10^{-5}: a conservative setup with high $\gamma = 0.99$ and strong regularization of important weights ($\lambda = 150$), and two aggressive setups with $\gamma = 0.8$ and weaker constraints ($\lambda = 40$ and $\lambda = 50$). All EWC results are reported for both 0% and 10% replay scenarios.

For LoRA, we report two configurations. The first employs a small number of parameters for task adaptation, with $r = 4$ and $\alpha = 8$. The second configuration, using $r = 8$ and $\alpha = 16$, provides greater flexibility by increasing the number of trainable parameters. In this setup, as more layers are frozen, the most relevant results were obtained using a learning rate of 10^{-3}. We evaluated performance both without replay and with 5% and 10% replay.[1]

For regularization-based methods, we report the evaluation of multiple configurations and replay strategies, including standard fine-tuning and variants with different replay settings. Some experiments freeze all except the classification head, while others freeze all layers but the last four (denoted as *Fr. all but last 4*) using a higher learning rate over 5 epochs. This gives the model slightly more flexibility to adapt to the new task, enabling more pronounced updates over a few batches.

Thus, in these experiments, we expect to observe lower effectiveness when updating the model with fewer samples, as the resulting gradients exhibit higher variance and provide a poor estimate of the underlying data distribution. As more samples are incorporated into each update, the gradients become more stable and representative, leading to more consistent training steps, improving its effectiveness.

However, using multiple new samples can bias the averaged gradient toward the incoming data distribution. If this differs from the original training data, it may shift model

[1]We report only LoRA results using a learning rate of 10^{-3}, as this setting consistently yielded the best performance for LoRA in preliminary experiments, whereas higher learning rates led to degraded performance with other algorithms.

Tech.	R%	Conf	lr	Forgetting Rate (ΔF1/upd.)			Impact (% data)			$\Delta F1$
				Test	Train$_{base}$	OL	Impr.	Wors.	Equal	
Reg	0	Baseline	10^{-5}	0,0009	-0,0133	0,0411	32,36	4,86	62,78	1,48
EWC	0	γ=0.8; λ=40	10^{-5}	0,0006	-0,0119	0,0757	26,94	4,41	68,64	1,33
	0	γ=0.8; λ=50	10^{-5}	0,0089	-0,0092	0,0689	27,60	3,92	68,47	1,14
	0	γ=0.99; λ=150	10^{-5}	0,0026	-0,0096	0,0736	27,90	**3,36**	68,72	1,29
	10	γ=0.8; λ=40	10^{-5}	0,0109	-0,0428	-0,0144	**88,13**	4,85	7,03	9,06
	10	γ=0.8; λ=50	10^{-5}	0,0027	-0,0425	-0,0134	86,57	4,07	9,36	8,95
	10	γ=0.99; λ=150	10^{-5}	-0,0074	-0,0387	-0,0127	85,90	4,60	9,50	8,19
LoRa	0	r=4; α=8	10^{-3}	-0,0461	-0,104	0,0046	35,92	6,26	57,83	3,35
	0	r=8; α=16	10^{-3}	-0,084	-0,153	0,0363	34,80	5,59	59,61	4,43
	5	r=4; α=8	10^{-3}	-0,0472	-0,169	-0,1033	87,31	6,15	6,53	**11,41**
	5	r=8; α=16	10^{-3}	-0,1035	-0,2365	-0,0983	86,38	9,86	3,77	10,37
	10	r=4; α=8	10^{-3}	-0,0473	-0,2032	-0,1558	87,35	7,78	4,86	10,60
	10	r=8; α=16	10^{-3}	-0,1276	-0,3165	-0,2261	86,17	10,15	3,66	10,37
Regul.	0	Fr. all but last 4	10^{-5}	-0,0018	**-0,0083**	**0,0796**	29,19	6,01	64,79	1,13
	0	Freeze all; 5 Ep.	10^{-2}	-0,0054	-0,0053	0,067	18,68	7,77	73,53	0,38
	5	No freeze	10^{-5}	0,0014	-0,0321	0,0699	79,81	4,13	16,06	7,22
	10	No freeze	10^{-5}	**0,0135**	-0,0462	0,0252	86,93	4,62	8,45	9,15

Table 3
Average results of experiments using an update size of five samples. Bold values indicate the best performance within each experiment, while underlined values indicate the best performance achieved by each technique.

parameters and increase forgetting. This effect is more pronounced on the *Train*$_{base}$, since F1 variation is measured relative to a checkpoint trained on this data.

5.1. Experiment results with updates of 1 sample

The results presented in Table 2 support our initial hypothesis, as no algorithm achieved an effectiveness above 50%, although most methods improved upon the baseline effectiveness. Additionally, several algorithms reduced the baseline forgetting rates, with some even exhibiting positive rates (i.e., improvements over previously learned data), thereby contributing to overall performance gains.

In this scenario, LoRA performs best with higher learning rates, achieving an effectiveness of 48.25% and a $\Delta F1$ of 8.19, while maintaining competitive forgetting test rate with a slightly positive rate. These results are obtained with $r = 4$ and $\alpha = 8$, using an intermediate replay level that enables adaptation without significantly altering the underlying model. In contrast, higher replay levels tend to hide new samples reducing effectiveness. However, this gain comes at the cost of a forgetting rate in the *Train*$_{base}$ approximately four times higher than the baseline.

In the case of EWC, we observe a balanced outcome, with high effectiveness (45.8%–45.14%) and a good trade-off between effectiveness and forgetting. This indicates that the method effectively protects relevant weights and is not adversely affected by replay (10%), achieving its best balance in the most conservative setting. The regularization-based approach also yields solid results: with 10% replay, the configuration with the highest improvement shows a slight increase in forgetting, while 5% replay improves forgetting at the cost of a lower proportion of improved samples.

5.2. Experiment results with updates of 5 samples

Higher effectiveness is clearly observed in this experimental setting (see Table 3), where several OL algorithms substantially outperform the baseline, achieving values of up to 88.13% compared to the baseline's 32.36%. This improvement, however, comes at the cost of increased forgetting, particularly on the *Train*$_{base}$, highlighting the inherent trade-off between effectiveness and the forgetting rate.

Replay strategies play a pivotal role in mitigating this trade-off: none of the experiments conducted without replay achieved effectiveness above 36%, demonstrating the importance of retaining prior knowledge. Among the algorithms tested, EWC stands out by achieving the highest effectiveness, accompanied by a large $\Delta F1$ and a near-best test forgetting rate on test set, which remains positive. At the same time, higher values of γ and λ constrain the learning process, limiting overall performance.

Regularization-based approaches benefit significantly from increased replay levels, which allow them to reach effectiveness close to the top results while maintaining a more balanced trade-off between forgetting rate, effectiveness, and $\Delta F1$ when using small learning rates. LoRa, on the other hand, attains effectiveness comparable to the best-performing methods but exhibits higher forgetting rates than the other algorithms, suggesting that its gains in effectiveness come at the expense of memory retention.

5.3. Experiment results with updates of 50 samples

Table 4 further supports our initial hypothesis. In this scenario, even the baseline achieves 90% effectiveness, with improvements on the test dataset, while almost all OL algorithms approach (or reach) 100% effectiveness and also

Tech.	R%	Conf	lr	Forgetting Rate (ΔF1/upd.)			Impact (% data)			$\Delta F1$
				Test	Train_{base}	OL	Impr.	Wors.	Equal	
Reg	0	Baseline	10^{-5}	0,1303	-0,128	-0,4464	90,06	4,39	5,56	5,29
EWC	0	γ=0.8; λ=40	10^{-5}	0,0618	-0,202	-0,5476	96,14	1,75	2,09	5,90
	0	γ=0.8; λ=50	10^{-5}	0,1016	-0,2048	-0,4189	98,53	1,46	0	6,52
	0	γ=0.99; λ=150	10^{-5}	0,1218	-0,1443	**0,003**	98,24	1,75	0	5,76
	10	γ=0.8; λ=40	10^{-5}	0,3523	-0,1649	-0,0989	99,71	0,29	0	9,76
	10	γ=0.8; λ=50	10^{-5}	0,1905	-0,1527	-0,3621	**100**	**0**	0	8,72
	10	γ=0.99; λ=150	10^{-5}	0,3436	-0,189	-0,4405	**100**	**0**	0	**10,26**
LoRa	0	r=4; α=8	10^{-3}	0,0323	-0,9473	-1,2911	88,62	9,28	2,10	7,43
	0	r=8; α=16	10^{-3}	-0,4576	-1,334	-1,7357	89,47	8,77	1,75	9,44
	5	r=4; α=8	10^{-3}	0,1743	-0,4742	-1,2296	99,71	0,29	0	9,41
	5	r=8; α=16	10^{-3}	0,0797	-0,72	-1,2065	95,54	4,46	0	6,86
	10	r=4; α=8	10^{-3}	-0,3689	-0,8296	-1,3136	99,12	0,87	0	8,54
	10	r=8; α=16	10^{-3}	-0,086	-0,5363	-0,5961	96,05	3,94	0	5,07
Regul.	0	Fr. all but last 4	10^{-5}	0,0853	-0,1329	-0,4581	86,44	5,61	7,95	4,10
	0	Freeze all; 5 Ep.	10^{-2}	0,1768	**0,0248**	-0,1524	76,26	7,07	16,67	1,78
	5	No freeze	10^{-5}	-0,0121	-0,1555	-0,2987	99,71	0,29	0	7,22
	10	No freeze	10^{-5}	**0,3774**	-0,2244	-0,8206	**100**	**0**	0	9,60

Table 4
Average results of experiments using an update size of 50 samples. Bold values indicate the best performance within each experiment, while underlined values indicate the best performance achieved by each technique.

exhibit gains on the test set. However, this increased effectiveness comes at the cost of substantially higher forgetting rates on $\textit{Train}_{base}$ and $\textit{Train}_{OL}$, showing that, despite this configuration may help to the generalization, it increase the forgetting on the previously seen knowledge.

We observe that EWC successfully learns new features, achieving 100% (or near) effectiveness across all experiments, with high forgetting rates surronding loses of 0.15 F1 score by update in $\textit{Train}_{base}$, and 0.4 in $\textit{Train}_{OL}$. In this case, replay-based methods help mitigate forgetting all the configurations except in the most conservative configuration. Only the most conservative configuration without replay ($\gamma = 0.99$ and $\lambda = 150$) effectively controls forgetting, but at the cost of slightly reduced overall gains.

Regularization experiments also achieve effectiveness scores reaching 100%; however, the forgetting rates on both the training and test datasets are notably higher compared to the other methods. Nevertheless, it is interesting to note that the most aggressive configuration, which freezes all layers except the last one that uses a high learning rate during 5 epochs, reduces the forgetting rate, achieving an effectiveness of 76.26%.

Finally, we can see that LoRa does not succeed in this task. Despite achieving high effectiveness scores, it exhibits the highest forgetting rates across the entire experimentation.

6. Discussion

Overall, the results support the initial hypothesis that the number of samples used in each update affects the balance between adaptation and forgetting. With very few samples, models exhibit lower effectiveness due to higher gradient variance and less reliable updates. As the number of samples increases, gradients become more stable, leading to higher effectiveness across all methods. However, this gain is consistently accompanied by greater forgetting of previously learned data, as stronger updates push the model away from earlier representations. These findings underscore a clear and consistent trade-off between plasticity and stability across all experiments.

Among the evaluated methods, EWC shows the most balanced performance, achieving competitive effectiveness while effectively mitigating forgetting. Its performance remains relatively stable across configurations, although it is still affected by the magnitude of the updates. Regularization-based approaches also provide a reasonable trade-off in smaller updates, but their effectiveness in controlling forgetting decreases with updates of 50 samples, where higher effectiveness is accompanied by substantially higher forgetting.

In contrast, LoRA demonstrates competitive performance in the extreme OL scenario, where updates occur sample by sample. However, it experiences greater forgetting than other methods in the other OL scenarios. This tendency is especially pronounced in higher-update settings, where LoRA adapts well to new data but struggles to retain previously learned knowledge.

Overall, the choice of method should depend on the target scenario: approaches such as EWC are more suitable when preserving prior knowledge is important, while methods like LoRA are better suited for scenarios that prioritize rapid adaptation at the expense of increased forgetting.

7. Conclusions

Our study demonstrates how different OL strategies impact the performance of transformer-based NERC models when continuously updated with new data. We show that the choice of method directly affects the balance between retaining prior knowledge and adapting to new information, highlighting the practical challenges of catastrophic forgetting in dynamic domains.

We evaluated multiple datasets from medical and legal domains, reorganized to simulate realistic online updates. This allowed a systematic assessment of OL strategies under varying update sizes and data dynamics.

Results show that update size plays a central role in model behavior, independently of the specific OL strategy employed. Smaller updates tend to introduce higher variance and provide limited learning signal, while larger updates enable stronger adaptation to new data but also amplify forgetting. This pattern reflects the inherent stability–plasticity trade-off, establishing update size as a key factor that shapes the learning dynamics of all methods considered.

Under these conditions, the different OL strategies exhibit distinct behaviors. EWC consistently provides a strong balance between retaining prior knowledge and adapting to new information, particularly in low-update regimes. Regularization-based approaches perform reliably when updates are controlled, but their effectiveness diminishes as update size increases. In contrast, LoRA adapts effectively across different settings, although increasing the update size leads to greater forgetting. Replay-based strategies help mitigate forgetting by reinforcing past knowledge, but when applied too aggressively, they can limit the model's ability to incorporate new information. These differences highlight that each technique manages the stability–plasticity trade-off in a characteristic way, conditioned—but not determined—by update size.

Overall, OL is a promising approach for dynamic NERC tasks, but its effectiveness depends on update size, technique configuration, and replay strategy. No method is universally best; the choice should align with the desired balance between stability and adaptability.

Future work should focus on the collection or construction of datasets exhibiting naturally occurring temporal drift, enabling evaluation under more realistic conditions. Additionally, future research could explore alternative approaches not considered in this study, such as model distillation, to assess whether they can better manage the stability–plasticity trade-off in continuously evolving environments.

References

[1] J. Kirkpatrick, R. Pascanu, N. C. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, Proceedings of the National Academy of Sciences 114 (2017) 3521–3526. doi:10.1073/pnas.1611835114.

[2] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: Low-rank adaptation of large language models, 2021. URL: https://arxiv.org/abs/2106.09685. arXiv:2106.09685.

[3] J. Li, A. Sun, J. Han, C. Li, A Survey on Deep Learning for Named Entity Recognition, IEEE Transactions on Knowledge and Data Engineering 34 (2022) 50–70. doi:10.1109/TKDE.2020.2981314.

[4] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, Y. Bengio, An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks, 2015. URL: https://arxiv.org/abs/1312.6211. arXiv:1312.6211.

[5] Y. Guo, B. Liu, D. Zhao, Online Continual Learning through Mutual Information Maximization, in: K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, S. Sabato (Eds.), Proceedings of the 39th International Conference on Machine Learning, volume 162 of *Proceedings of Machine Learning Research*, PMLR, 2022, pp. 8109–8126. URL: https://proceedings.mlr.press/v162/guo22g.html.

[6] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, G. Wayne, Experience replay for continual learning, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, volume 32, Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

[7] H. Shin, J. K. Lee, J. Kim, J. Kim, Continual learning with deep generative replay, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 2994–3003.

[8] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, C. H. Lampert, iCaRL: Incremental Classifier and Representation Learning, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5533–5542. doi:10.1109/CVPR.2017.587.

[9] X. Liu, X. Chang, Elastic Weight Consolidation Done Right for Continual Learning, 2026. URL: https://arxiv.org/abs/2603.18596. arXiv:2603.18596.

[10] J. Houyon, A. Cioppa, Y. Ghunaim, M. Alfarra, A. Halin, M. Henry, B. Ghanem, M. V. Droogenbroeck, Online Distillation with Continual Learning for Cyclic Domain Shifts, 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2023) 2437–2446. URL: https://api.semanticscholar.org/CorpusID:257921185.

[11] S. A. Bidaki, A. Mohammadkhah, K. Rezaee, F. Hassani, S. Eskandari, M. Salahi, M. M. Ghassemi, Online Continual Learning: A Systematic Literature Review of Approaches, Challenges, and Benchmarks, 2025. URL: https://arxiv.org/abs/2501.04897. arXiv:2501.04897.

[12] J. Bornschein, Y. Li, A. Rannen-Triki, Transformers for Supervised Online Continual Learning, 2024. URL: https://arxiv.org/abs/2403.01554. arXiv:2403.01554.

[13] A. Douillard, A. Ramé, G. Couairon, M. Cord, DyTox: Transformers for Continual Learning with DYnamic TOken eXpansion, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

[14] N. Monaikul, G. Castellucci, S. Filice, O. Rokhlenko, Continual Learning for Named Entity Recognition, Proceedings of the AAAI Conference on Artificial Intelligence 35 (2021) 13570–13577. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17600. doi:10.1609/aaai.v35i15.17600.

[15] A. Piad-Morfis, S. Estevez-Velarde, Y. Gutierrez, Y. Almeida-Cruz, A. Montoyo, R. Muñoz, Overview of the eHealth Knowledge Discovery Challenge at IberLEF 2021, Procesamiento del Lenguaje Natural 67 (2021) 233–242.

[16] A. Miranda-Escalada, E. Farré-Maduell, S. Lima-López, D. Estrada, L. Gascó, M. Krallinger, Mention detection, normalization & classification of species, pathogens, humans and food in clinical documents: Overview of LivingNER shared task and resources, Procesamiento

del Lenguaje Natural (2022).
[17] S. Lima-López, E. Farré-Maduell, L. Gascó, A. Nentidis, A. Krithara, G. Katsimpras, G. Paliouras, M. Krallinger, Overview of MedProcNER task on medical procedure detection and entity linking at BioASQ 2023, in: Working Notes of CLEF 2023 – Conference and Labs of the Evaluation Forum, 2023.
[18] P. Kalamkar, A. Agarwal, A. Tiwari, S. Gupta, S. Karn, V. Raghavan, Named Entity Recognition in Indian court judgments, in: Proceedings of the Natural Legal Language Processing Workshop 2022, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), 2022, pp. 184–193. URL: https://aclanthology.org/2022.nllp-1.15. doi:10.18653/v1/2022.nllp-1.15.
[19] SNOMED International, SNOMED Clinical Terms (SNOMED CT) International Edition, https://www.snomed.org/snomed-ct, 2025. March 2025 Release.
[20] P. Turón, M. Cuadros, Perplexity, Uncertainty, and the Limits of Active Learning, in: Hybrid Artificial Intelligent Systems, Springer Nature Switzerland, Cham, 2026, pp. 315–327.