Video Piracy Websites Detection using Continual Learning with Elastic Weight Consolidation

Dianzhi Yu(✉)[1][0009−0008−9602−4547], Wentao Zhang[1][0009−0005−1192−8247], Zenglin Xu[2][0000−0001−5550−6461], and Irwin King[1][0000−0001−8106−6447]

[1] Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China
{dianzhi.yu,wtzhang}@link.cuhk.edu.hk, king@cse.cuhk.edu.hk
[2] Artificial Intelligence Innovation and Incubation Institute, Fudan University, Shanghai, China
zenglinxu@fudan.edu.cn

Abstract. With the rapid development of the Internet and video streaming, video piracy websites (VPWs) pose a global challenge to the copyright protection of digital media as the black market for pirated videos continues to grow. In this paper, we design a **V**ideo **P**iracy **W**ebsites **D**etection (**VPWD**) architecture and implement the system that supports the end-to-end process of possible VPW collection to detection. When designing the architecture, we find that there are currently no publicly available datasets containing the contents of English and Chinese VPWs. Therefore, we build a real-world dataset of features targeting VPWs, containing 14,254 possible English and Chinese VPWs, with 5,759 of them labeled. To the best of our knowledge, it is the ***largest*** dataset for VPW detection. We aim to solve the challenges of detecting VPWs. Given the constant emergence of new VPWs with different VPW-related keywords and languages, a model trained once is not enough for detection. We demonstrate that such a fixed model will suffer from performance drops of over 8%, when possible VPWs with new keywords or languages are added to the database, resulting in feature distribution change. Our VPWD architecture is continual learning-based, which can continually learn features of VPWs. As far as we are aware, it is the ***first*** work to utilize continual learning for piracy website detection. We conduct experiments to compare continual learning algorithms in our VPWD architecture and achieve an accuracy of 96.4% on average, which is 0.5% higher than without continual learning algorithms.

Keywords: Website Detection · Piracy Websites · Continual learning.

1 Introduction

The rapid development of the Internet benefits content creators and users, but it also facilitates the fast global growth of the black market of video piracy websites (VPWs), challenging the copyright protection of digital media. VPWs distribute video content without permission from copyright holders or legitimate

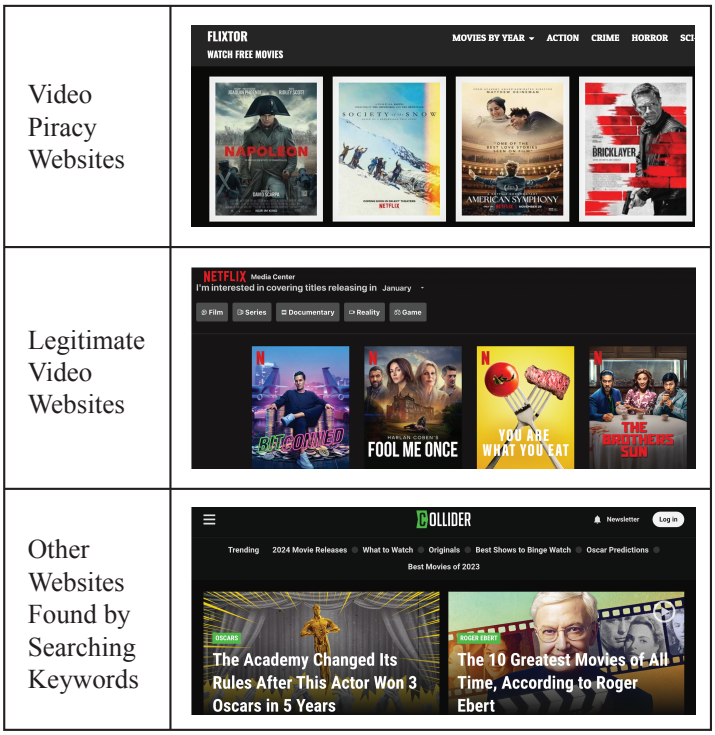


Fig. 1. Examples of VPWs and Non-VPWs.

purchasers, significantly financially impacting movies, TV shows, and other digital media industries [7]. VPWs are a hotbed for the growth of pirated content, accessed over 82 billion times worldwide in the first nine months of 2021 alone [1]. The United States is the country most affected by pirated digital media content, with about 11% of adults viewing pirated content in 2022 [3]. The U.S. economy loses at least USD $29.2 billion in revenue each year due to pirated material by estimation, with about 80% attributed to illegal live streams and videos [2]. China also suffers from VPWs. In 2021, China's VPW visits reached 5.9 billion, ranking among the top five globally [1]. China is expected to lose USD $9.78 billion through piracy in the movie and online TV revenue in 2022 [27].

In this paper, we design a **V**ideo **P**iracy **W**ebsites **D**etection (**VPWD**) architecture that supports the end-to-end process of active possible VPW collection to detection. VPWD can continually collect data from the Internet and continually learn features of VPWs to detect them with high accuracy.

When designing the architecture, we find that there are currently no publicly available datasets containing English and Chinese VPWs' content. Related datasets [18, 32], which focus on Chinese VPWs, do not contain the website contents and have small sizes. Therefore, we gather sufficient data of VPWs and non-VPWs for benchmarking by crawling web pages from search engine re-

sults using video piracy-related keywords. Our crawling toolkit is modified based on [28]. Fig. 1 shows examples of VPWs, legitimate video websites, and other websites found by searching keywords. We can see that VPWs have features that characterize their site content. For example, they tend to put specific keywords like "Watch free movies" at the top to attract visitors and rank higher in search engine results. Legitimate video websites like Netflix and other websites containing articles about movies and online series are also discovered by the crawler, but they are not VPWs. We aim to distinguish VPWs from others. After data processing, we transform this problem into a binary classification problem for text sequences and train a model to solve it.

We would like to highlight three challenges in VPW detection. (i) **Performance drop with single training.** If we train a model only once, the performance of the model may decrease when new datasets of VPWs arrive. We can add new keywords to the crawling tool to expand the search range of VPWs, consequently enlarging the dataset. The new data potentially have a different feature distribution from the old data. In recent works [18,28,32] for VPW detection, the data is collected once and used to train a model. We demonstrate that such a fixed model will suffer from performance drops in later experiments. (ii) **Large runtime and resource cost.** Training a new model on both new and old data together whenever new datasets are collected would be highly time-consuming and resource-inefficient. Training a separate model for each dataset is not desirable either since models cannot utilize shared knowledge across the datasets. (iii) **Lack of old data accessibility.** Some or all old training data may be lost or unavailable due to storage limits or other factors, which is an unfortunate but realistic challenge.

We incorporate continual learning methods into VPW detection to address the three challenges above. In this setting, a model is trained sequentially on multiple tasks. It enables models to learn from new tasks without explicitly retraining on previous tasks. The goal is to achieve good performance on new tasks while maintaining performance on old ones. Therefore, continual learning can address the challenges by (i) training a model continually, (ii) doing so time- and resource-efficiently by training only on new data and (iii) not training on old data. The continual learning setting reflects common real-world scenarios. Not all data is readily available for training, which is a non-ideal but realistic situation. In our dataset, VPWs are collected with different keywords and two languages. We shall give our VPWD architecture the capability to continue learning when possible VPWs with new keywords or languages are added to the dataset. Therefore, we further leverage our model to perform classification in a continual learning setting. As far as we know, our work is the first to utilize continual learning for the specific task of piracy website detection. Another related but distinct study by Ejaz et al. [6] applies continual learning in a vanilla neural network for phishing attack detection. Specifically, we employ an enhanced version [12] of a continual learning method known as Elastic Weight Consolidation (EWC) [14] on a BERT [5] model to enable continual learning.

Contributions. In summary, our contributions of this paper are as follows:

1. We propose a continual learning-based VPWD architecture and implement the system for the end-to-end process of VPW collection to detection. The architecture enables the model to learn continually and overcome performance drops of over 8%, from which we demonstrate that a fixed model will suffer when datasets of VPWs with new keywords or languages are added. To the best of our knowledge, it is the ***first*** work to utilize continual learning for piracy website detection.
2. We build a real-world dataset of features targeting VPWs by crawling the results of VPW-related keywords on search engines. The dataset contains 14,254 possible English and Chinese VPWs, with 5,759 of them labeled. As far as we are aware, it is the ***largest*** dataset for VPW detection, available at https://github.com/LucyDYu/VPWD.
3. We conduct experiments on our dataset using continual learning algorithms and achieve an accuracy of 96.4% on average, which is 0.5% higher than without continual learning algorithms.

The rest of this paper is organized as follows. Section 2 discusses the related work on website detection and continual learning. Section 3 shows the overall architecture and describes the methods used to detect VPWs. Section 4 presents the experiment results. Section 5 provides discussions and future work. Section 6 concludes the paper.

2 Related Work

This section discusses the related work on website detection and continual learning. Using computer programs and algorithms to detect certain types of websites is more efficient than manual identification by humans. Continual learning involves efficient learning methods that allow a model to learn new knowledge when data is continually gathered rather than fixed.

2.1 Website Detection

Website detection encompasses a variety of tasks depending on the intended purpose, including but not limited to piracy website detection, phishing website detection, and malicious website detection. In terms of detection approaches, there are generally three categories: non-machine learning, traditional machine learning, and deep learning approaches.

Non-machine learning. Choi and Kwak [4] analyze features of contents on piracy sites and form rules to detect them. Kim and Kwak [13] also employ sophisticated rules, including regular expressions, to detect piracy sites.

Traditional machine learning. Naïve Bayes Classifier achieves the highest accuracy in [15] for phishing website classification compared to Logistic Regression, Decision Tree, and other methods. Random Forest (RF) is used in [26] for phishing website detection.

Deep learning. Zhang et al. [32] construct a heterogeneous graph of piracy video websites and propose a graph model called HGNR for detection. Wang

et al. [28] fine-tune a pre-trained BERT model to identify Chinese video piracy websites and perform familial clustering. TEP-Net [18] is a fusion method of one-dimensional CNN and RF, using domain names and third-party services' information to identify Chinese pirated video websites. Ejaz et al. [6] propose a continual learning-based framework to detect phishing attacks using a vanilla neural network with continual learning methods, namely LwF [19] and EWC [14].

2.2 Continual Learning

In a continual learning (CL) scenario, a model is trained sequentially on various tasks. The objective is to achieve strong performance on new tasks while maintaining performance on old ones without explicitly retraining. Continual learning is challenging due to a widely acknowledged reason–**catastrophic forgetting**: this phenomenon happens when a neural network is trained on a new task and parameters are updated correspondingly, causing a significant drop in performance on previously learned tasks [22].

Continual learning algorithms can be categorized into three main groups. Regularization-based methods impose additional constraints on the parameters through regularization terms, adding extra costs associated with parameter changes. Architecture-based methods modify the model structure or use separate sets of parameters for each task, enabling the model to learn new tasks and keep knowledge of the old ones. Replay-based methods utilize some data or a generative model to generate pseudo-data of previous tasks to help maintain performance while learning new tasks.

Regularization-based. EWC [14] applies regularization terms as a penalty, utilizing the diagonal of the Fisher information matrix to restrict parameter changes elastically. EWC restricts the model from moving away from multiple optimal points as the number of tasks increases. Huszár [12] proposes to enhance EWC by considering only the previous optimal point and employing a single regularization term, regardless of the number of tasks.

Architecture-based. Progressive Network [25] continuously expands the network for new tasks and supports knowledge transfer via lateral connections. APD [31] is an order-robust method that decomposes parameters into task-shared and sparse task-adaptive parameters to prevent forgetting.

Replay-based. iCaRL [23] uses exemplar sets of the old task data. When training for a new task, it uses distillation loss to minimize the change in the output of exemplar sets. GEM [20] also uses data of old tasks as episodic memory and allows positive backward transfer of knowledge from new to old tasks by using loss in episodic memory as inequality constraints.

3 Methodology

In this section, we describe the methods we use in detail. Our VPWD architecture consists of three components as illustrated in Fig. 2: A. Website Collection, B. Dataset Construction, and C. VPW Detection with Continual Learning.

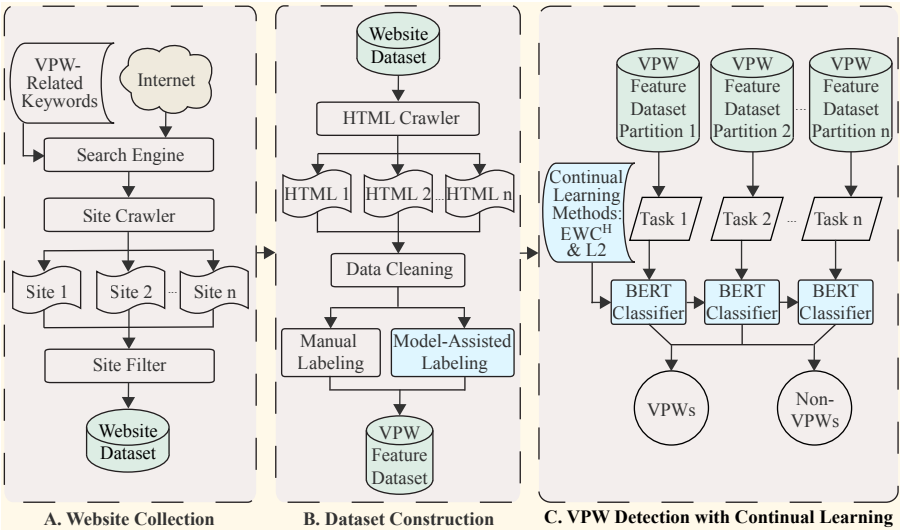


Fig. 2. Video Piracy Website Detection (VPWD) architecture.

3.1 Website Collection

The first component of our VPWD architecture involves utilizing a crawling toolkit modified based on [28]. We use a site crawler to collect URLs from search engine results using VPW-related keywords and get 103,376 URLs. These keywords include both English and Chinese phrases, such as "free movie" and "免费观看" (watch for free). We keep a record table to track and avoid revisiting existing pages to prevent redundant crawling. We modify the crawling toolkit to keep both English and Chinese websites, remove duplicate URLs based on domain names to keep one URL per domain, and filter out site index and traffic analytics websites because they only contain statistics of other sites. This process resulted in a dataset of 14,254 potential VPWs.

3.2 Dataset Construction

We use an HTML crawler to collect HTML files from the URLs. Note that we only collect publicly available data for research purposes. We then extract text from these files and combine it with the website's domain, title, keyword, and URL to create our dataset for VPW detection. Note that the data crawling steps are similar to [28] because our crawling toolkit is modified based on this work, with modifications in Section 3.1 and performing data cleaning of the extracted text.

Our system requires labeled data of VPWs and non-VPWs. Two steps are involved in labeling the collected data: manual labeling and model-assisted labeling. Using these two methods together allows us to obtain labels efficiently while maintaining accuracy. We face the challenge that manually labeling VPWs

is labor intensive since the annotators need to check the website to identify if it is a VPW. Three annotators label 567 websites from our dataset, with 254 VPWs and 313 non-VPWs. Although it is ideal to manually label all data, it is not feasible due to limitations in resources and time.

We label the remaining unlabeled data in a classical semi-supervised learning fashion according to categories from [30]. As the results in [8] show, text classification methods based on BERT models [5] outperform traditional machine learning algorithms with counting-based TF-IDF vocabulary [24]. Moreover, BERT models have been employed widely in various NLP tasks such as federated multilingual modeling [9], question answering [10] and entity typing [17], demonstrating the effectiveness and robustness of BERT models. Therefore, we adopt a pre-trained BERT model with a final layer as the base classifier, fine-tune it on the manually labeled data to learn relevant VPW features, and apply it to infer labels for the unlabeled data. We only select labels from the model with a probability over 98%.

By combining manually and model-labeled data, we obtain 5,759 labeled websites with 2,794 VPWs and 2,965 non-VPWs for our task. Training another BERT model on all the labeled data at once may appear repetitive due to many labels generated by the previously trained BERT model, but this still falls within the semi-supervised learning framework. Moreover, in our CL setting, we partition the dataset into tasks and train a CL-based model sequentially on these tasks. Therefore, the process is not simple repetition.

3.3 VPW Detection with Continual Learning

We divide our dataset into subsets and treat them as different tasks. We fine-tune a BERT model sequentially on these tasks to simulate the CL setting. This process simulates real-world scenarios where the crawler collects new VPWs and our model needs to learn new VPW features while retaining learned knowledge. Detailed experiments are in Section 4. As more keywords are added, our VPWD architecture can crawl more data, label them (manual labeling is optional for improving labeling quality), and continually learn VPW features to detect them.

The following subsections introduce preliminaries, the BERT classifier, and the CL method EWC with an improved version in detail.

Preliminaries. Notations. We use bold lowercase, bold uppercase, and calligraphy letters for vectors, matrices, and sets, respectively. We list the key notations in Table 1.

Problem Formulation. Given a sequence of tasks with datasets, continual learning aims to train the model on new task data and maintain performance on old tasks to overcome catastrophic forgetting.

BERT Classifier. Our VPWD architecture uses a pre-trained BERT model with a final layer for VPW classification. We concatenate features with spaces to form the text sequences. Then, we fine-tune the model using these input

Table 1. Notations and descriptions.

Notation	Description
$\mathcal{D}$	The entire dataset
$A, B, \cdots, T$	Task labels
$\mathcal{D}_A, \mathcal{D}_B, \cdots, \mathcal{D}_T$	Partitions of $\mathcal{D}$; $\mathcal{D} = \mathcal{D}_A \sqcup \mathcal{D}_B \sqcup \cdots \sqcup \mathcal{D}_T$
$\mathbf{F}_A$	The Fisher information matrix (FIM) of task A
$F_{A,i}$	The i-th diagonal entry of $\mathbf{F}_A$; $F_{A,i} = [\mathbf{F}_A]_{ii}$
$\boldsymbol{\theta}$	Trainable parameters

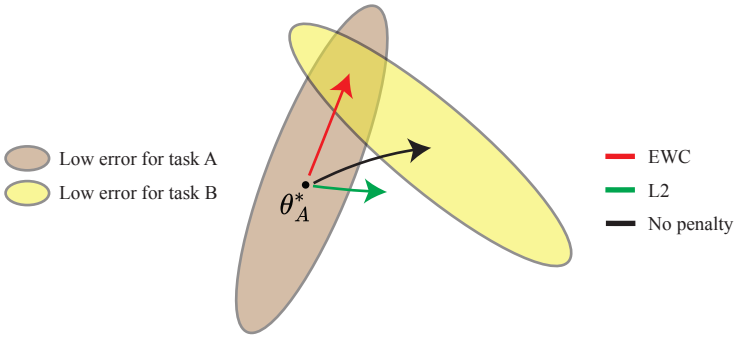


Fig. 3. Graphical illustration of EWC. The figure is adapted and redrawn based on [14]. The two ellipses represent low error regions of task A and task B. L2 penalty adds equal constraint on all parameters. Parameters may not learn the new task when the restriction is severe or move outside the low error regions of both tasks, as shown by the green arrow. EWC finds a solution in a low error region of both tasks, as shown by the red arrow, thus alleviating forgetting [14].

text sequences and labels from the training set to learn VPW features. As the BERT model can only accept 512 tokens, it is impossible to input whole text sequences at once. Methods to process long sequences with BERT include direct truncation, sliding windows [29], etc. By analyzing website text characteristics, we find that information at the beginning of the sequences of VPWs is more important, while subsequent text sequences tend to be less significant movie and online series descriptions. Therefore, we choose to truncate from the tail end of the text sequence as input to the BERT model.

We use the BERT multilingual base model (cased) [5] as our base model because it is case sensitive and capable of understanding multiple languages. Case information is important for our task since normal words in movie or series titles with uppercase have different meanings. Since the model is pre-trained on 104 languages, it provides the capability to understand numerous languages in addition to English and Chinese. This enables our VPWD architecture to continually learn features of VPWs when data in a new language arrives.

Elastic Weight Consolidation (EWC). This section provides an overview of EWC [14], a regularization-based CL method. We combine our fine-tuned BERT model and an improved version [12] of EWC to enable our VPWD architecture to continually learn features of VPWs. Kirkpatrick et al. [14] propose EWC, which uses the diagonal of the Fisher information matrix (FIM) to restrict parameter change. After training on task A and obtaining optimal parameters $\boldsymbol{\theta}^*_A$, task B is trained with the following loss function in EWC [14]:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_B(\boldsymbol{\theta}) + \sum_i \frac{\lambda}{2} F_{A,i} \left(\theta_i - \theta^*_{A,i}\right)^2, \tag{1}$$

where $\mathcal{L}_B(\boldsymbol{\theta})$ is the loss for task B; λ is a CL regularization hyperparameter reflecting importance of previous tasks; i is the label of each parameter; $\mathbf{F}_A$ is the estimated FIM via sampling from $\mathcal{D}_A$ and calculating the expectation of squared gradient of log-likelihood of each parameter. EWC approximates the posterior $\log p\left(\theta \mid \mathcal{D}_A\right)$ using a crude Laplace approximation around $\boldsymbol{\theta}^*_A$ by only using the diagonal entries of $\mathbf{F}_A$ [14].

Compared to **L2 penalty** which is considered to be a special case of EWC and imposes equal constraints on all parameters [16]:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_B(\boldsymbol{\theta}) + \sum_i \frac{\lambda}{2} \left(\theta_i - \theta^*_{A,i}\right)^2, \tag{2}$$

$F_{A,i}$ imposes finer control over parameters, as it reflects the importance of parameters to the first task A. EWC allows parameters that are less important to task A to update more freely to minimize the loss, while parameters that are more important to task A cannot change much to maintain task A performance. Fig. 3 graphically illustrates EWC and L2.

When a third task C is introduced, EWC adds another regularization term to restrict parameters from moving away from the optimal point $\boldsymbol{\theta}^*_{A,B}$ after training on tasks A and B, on top of the previous regularization term. EWC performs well to prevent catastrophic forgetting, but it has limitations. The model is restricted from moving away from multiple optimal points as the number of tasks increases, leading to suboptimal performance and sensitivity to task order since optimal parameters of early tasks are repeatedly involved in the loss [12].

Improved Elastic Weighted Consolidation. Huszár [11, 12] proposes an enhancement to EWC to address the limitation mentioned in Section 3.3. We call it **EWC$^{\mathbf{H}}$**. EWC$^{\mathrm{H}}$ considers only the previous optimal point and maintains only a single regularization term, improving calculation efficiency and demonstrating better performance. For three tasks, the new loss function in EWC$^{\mathrm{H}}$ [12] is:

$$\mathcal{L}\left(\boldsymbol{\theta}\right) = \mathcal{L}_C\left(\boldsymbol{\theta}\right) + \sum_i \frac{\lambda}{2} \left(F_{A,i} + F_{B,i}\right) \left(\theta_i - \theta^*_{A,B,i}\right)^2, \tag{3}$$

where $\mathbf{F}_B$ represents the estimated FIM via sampling from $\mathcal{D}_B$. We further leverage our fine-tuned BERT model in a CL setting using EWC$^{\mathrm{H}}$.

Table 2. Dataset Distribution - Keywords and Languages

Dimension		VPWs	Non-VPWs	Total	VPW Rate
Keywords	免费观看 (watch for free)	136	23	159	85.5%
	free movie	49	47	96	51.0%
	online series	57	76	133	42.9%
	Other keywords	2,552	2,819	5,371	47.5%
Languages	English	518	2,144	2,662	19.5%
	Chinese	2276	821	3,097	73.5%
Total		2,794	2,965	5,759	48.5%

4 Experiments

4.1 Dataset

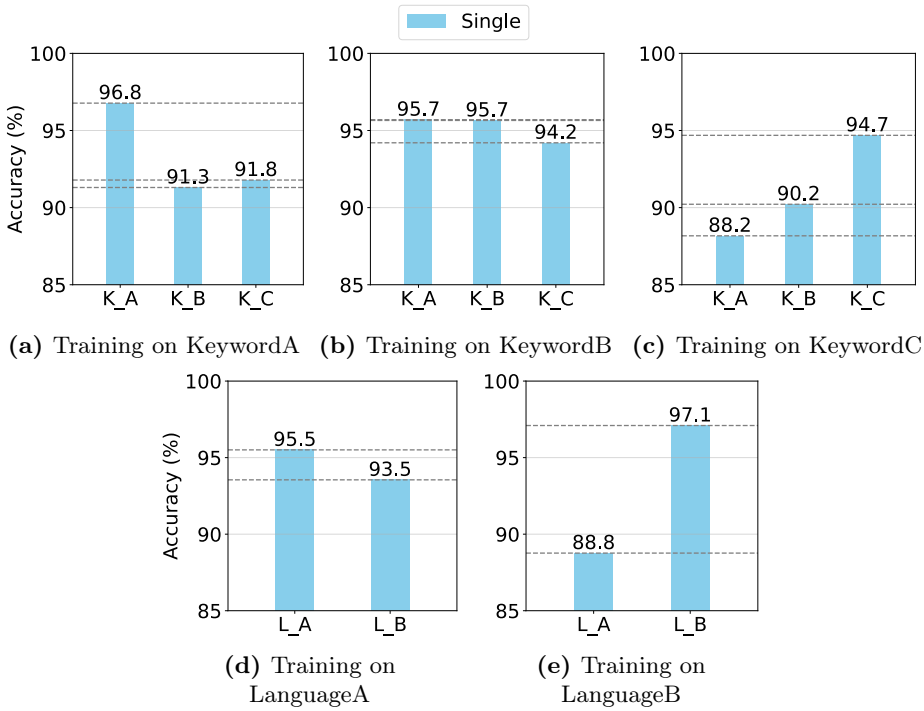
We use 52 VPW-related keywords to crawl data of English and Chinese websites. Table 2 shows sample numbers of VPWs and non-VPWs for some example keywords. To simulate the CL setting, we divide the data into partitions and train a model sequentially on these partitions. We partition the data by two dimensions: keywords and languages.

Keywords. The starting point of our VPWD architecture is searching keywords on the Internet. With more keywords, the crawler can collect more data. We simulate this by partitioning our dataset by keywords. We sort keywords and split them into three groups so that more related keywords belong to the same group, simulating the case of adding related keywords to the crawler. We get three datasets and name them KeywordA, KeywordB, and KeywordC, containing 18, 17, and 17 keywords, respectively. Each group contains both English and Chinese website data.

Languages. In this paper, we focus on English and Chinese websites, so we partition the data by language and name the datasets as LanguageA (English) and LanguageB (Chinese). Table 2 shows sample numbers of VPWs and non-VPWs for English and Chinese. We simulate the situation where the crawler collects VPW of a new language.

4.2 Experiment Setup

We employ the pre-trained BERT multilingual base model (cased) [5] with a final layer for classification. We utilize the AdamW optimizer [21] to update neural network parameters during training. Throughout the model training phase, we use a learning rate of 10^{-5} and limit the training to 10 epochs. The batch size per GPU is 16. The CL regularization λ in keyword and language partition experiments is set to 200 and 100, respectively. The training, validation, and testing ratio for each dataset partition is 81, 9, and 10 percent, respectively. All experiments are conducted with PyTorch on two NVIDIA TITAN RTX GPUs with 24G memory. The models are run three times and the average performances are reported with standard deviation in Table 3 and 4. Graphical figures illustrate one run of the models in detail in Fig. 4 and 5.



(a) Training on KeywordA **(b)** Training on KeywordB **(c)** Training on KeywordC

(d) Training on LanguageA **(e)** Training on LanguageB

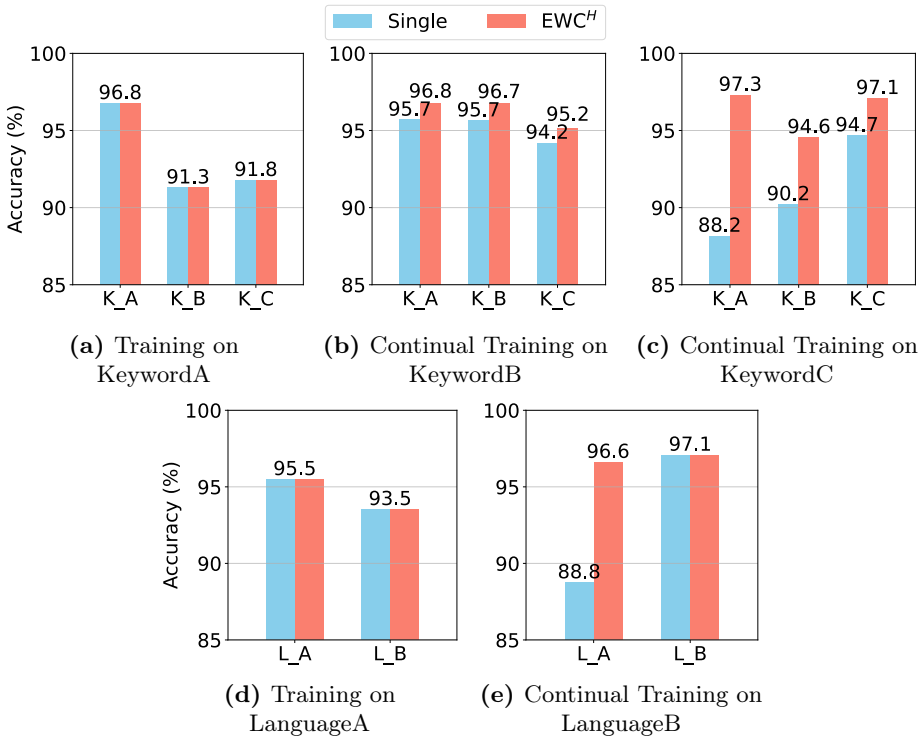
Fig. 4. Test accuracy when training on one dataset.

4.3 Research Questions Addressed

In order to evaluate our VPWD architecture for the VPW detection task, we ask the following Research Questions (RQs):

- RQ1. How does a fixed model perform on new datasets?
- RQ2. What is the effect of the EWC^{H} method when the model is continually trained on multiple datasets?
- RQ3. How does EWC^{H} as a CL algorithm compare with other CL algorithms (L2) and non-CL methods (Fine-tuning)?

Performance of a Fixed Model on New Datasets (RQ1). We partition the data according to keywords and languages as described in Section 4.1. We train the model on one of the datasets KeywordA, KeywordB and KeywordC, and report the test accuracy of the model on all datasets. Similarly, we train the model on either of the datasets LanguageA or LanguageB, and report the test accuracy on both datasets. We use the binary cross-entropy loss function. Since the model is trained on one dataset, we label it as **Single**. We empirically evaluate the performance drop in this section.



(a) Training on KeywordA **(b)** Continual Training on KeywordB **(c)** Continual Training on KeywordC

(d) Training on LanguageA **(e)** Continual Training on LanguageB

Fig. 5. Test accuracy when training with EWC$^{\mathrm{H}}$.

As we can see from Fig. 4, when the model is trained on one dataset, the performance on this dataset is high, but the performance on other datasets is much lower. This performance drop occurs in both keyword and language partition cases due to feature distribution changes. Fig. 4a shows that the model trained on KeywordA achieves 96.8% test accuracy on KeywordA, while those of KeywordB and KeywordC are 91.3% and 91.8%. Fig. 4e shows that the model trained on LanguageB achieves 97.1% test accuracy on LanguageB, while that of LanguageA is 88.8%, dropping over 8%. This result demonstrates that if we train a model only once, when data of VPWs with new keywords or languages arrive, this fixed model will have lower performance. Therefore, it is necessary to continually train the model on new datasets.

The Effect of the EWC$^{\mathbf{H}}$ Method on Continual Training (RQ2). We use the binary cross-entropy loss and modify it as mentioned in Section 3.3 Eq. (3) for **EWC$^{\mathbf{H}}$**. We train on the same model structure, keeping model parameters the same on the first task for a fair comparison in the CL setting. Therefore, Fig. 5a and 5d, as the first task in the keyword and language partition experiments respectively, show the same accuracy for **Single** and **EWC$^{\mathbf{H}}$**.

Table 3. Performance comparison of different methods on each task using keyword-based dataset partition.

Method	Test Accuracy				AUC Value				Precision	Recall	F1 Score
	KeywordA	KeywordB	KeywordC	Average	KeywordA	KeywordB	KeywordC	Average	Average	Average	Average
Fine-tuning	$97.1_{\pm.3}$	$94.7_{\pm.3}$	$96.0_{\pm.7}$	$95.9_{\pm.3}$	$96.2_{\pm1.1}$	$97.5_{\pm.2}$	$99.3_{\pm.1}$	$97.7_{\pm.4}$	$97.5_{\pm.7}$	$91.8_{\pm.1}$	$94.5_{\pm.4}$
L2	$95.7_{\pm.0}$	$92.9_{\pm1.4}$	$93.1_{\pm.3}$	$93.9_{\pm.5}$	$95.3_{\pm2.4}$	$97.3_{\pm.8}$	$97.6_{\pm.1}$	$96.7_{\pm.9}$	$93.3_{\pm.8}$	$90.9_{\pm.4}$	$92.1_{\pm.5}$
EWC$^{\mathrm{H}}$	$\mathbf{97.5}_{\pm.3}$	$\mathbf{94.9}_{\pm.6}$	$\mathbf{96.9}_{\pm.7}$	$\mathbf{96.4}_{\pm.3}$	$\mathbf{98.0}_{\pm.8}$	$\mathbf{98.2}_{\pm.1}$	$\mathbf{99.4}_{\pm.1}$	$\mathbf{98.5}_{\pm.3}$	$\mathbf{97.8}_{\pm.4}$	$\mathbf{92.7}_{\pm.4}$	$\mathbf{95.1}_{\pm.3}$

Table 4. Performance comparison of different methods on each task using language-based dataset partition.

Method	Test Accuracy			AUC Value			Precision	Recall	F1 Score
	LanguageA	LanguageB	Average	LanguageA	LanguageB	Average	Average	Average	Average
Fine-tuning	$93.8_{\pm1.6}$	$\mathbf{97.5}_{\pm.4}$	$95.6_{\pm.6}$	$96.0_{\pm1.0}$	$\mathbf{99.3}_{\pm.0}$	$97.6_{\pm.6}$	$96.0_{\pm.7}$	$87.3_{\pm4.3}$	$90.9_{\pm2.3}$
L2	$\mathbf{95.3}_{\pm.4}$	$93.3_{\pm1.2}$	$94.3_{\pm.8}$	$\mathbf{97.2}_{\pm.2}$	$95.4_{\pm.7}$	$96.3_{\pm.4}$	$93.8_{\pm.9}$	$\mathbf{90.4}_{\pm.8}$	$91.9_{\pm1.0}$
EWC$^{\mathrm{H}}$	$95.1_{\pm1.4}$	$96.8_{\pm.6}$	$\mathbf{96.0}_{\pm.9}$	$96.5_{\pm.6}$	$\mathbf{99.3}_{\pm.1}$	$\mathbf{97.9}_{\pm.3}$	$\mathbf{96.6}_{\pm1.2}$	$89.5_{\pm3.1}$	$\mathbf{92.6}_{\pm2.1}$

As we can see from Fig. 5, with **EWC$^{\mathbf{H}}$**, when the model continually trains on a new task, performance on old tasks remains high in both keyword and language partition cases. Fig. 5c shows that the model trained on KeywordC achieves 97.1% test accuracy on KeywordC, and 97.3% and 94.6% on KeywordA and KeywordB. Fig. 5e shows that the model trained on LanguageB achieves 97.1% test accuracy on LanguageB, and 96.6% on LanguageA. This result demonstrates the effectiveness of EWC$^{\mathrm{H}}$ and the necessity of continual learning.

Performance Comparison of CL Algorithms and Non-CL Methods (RQ3). We sequentially train our model on tasks using CL and non-CL methods. Again, we train on the same model structure, keeping the parameters the same on the first task for a fair comparison. For the second task and onwards, different methods are used in training as follows:

Fine-tuning. We fine-tune the model sequentially on datasets of tasks.

L2. We modify the loss function as mentioned in Section 3.3 Eq. (2).

EWC$^{\mathbf{H}}$. We modify the loss function as mentioned in Section 3.3 Eq. (3).

As shown in Table 3 and 4, **EWC$^{\mathbf{H}}$** achieves the highest average test accuracy in both keyword partition and language partition experiments. In the keyword partition experiment, the test accuracy of EWC$^{\mathrm{H}}$ is 96.4%, 0.5% higher than fine-tuning. In the language partition experiment, the test accuracy of EWC$^{\mathrm{H}}$ is 96.0%, 0.4% higher than fine-tuning. EWC$^{\mathrm{H}}$ also achieves the highest average AUC value. The results show that EWC$^{\mathrm{H}}$ is an effective choice of CL method. We also report the average precision, recall, and F1 score, where EWC$^{\mathrm{H}}$ achieves the highest metric most times.

5 Discussions and Future Work

Complexity Analysis. EWC$^{\mathrm{H}}$ uses the FIM, which is obtained by sampling from task datasets and calculating the expectation of squared gradient of log-

likelihood of each parameter. Assuming a fixed sample size and gradient calculation is bounded by a constant, the time complexity for obtaining an FIM is proportional to the number of parameters and tasks, i.e. $O(P \cdot T)$, where P represents the number of parameters and T represents the number of tasks. We only calculate the FIM between the training of two tasks and then use it to elastically constrain parameters. Since each task involves lengthy training, this computation's overhead is negligible compared to the total training time. EWC also has low computational complexity in run time [14]. Therefore, our VPWD architecture with continual learning capability is as efficient as without EWC$^{\mathrm{H}}$.

Limitations. Our VPWD architecture uses EWC$^{\mathrm{H}}$ to enable continual learning. However, EWC and EWC$^{\mathrm{H}}$ only use the diagonal of the FIM and assume each parameter is independent of others [16]. This assumption is fragile and not true in general. Moreover, EWC requires additional storage and memory to store the diagonal of the FIM and the previous optimal parameters $\boldsymbol{\theta}^*$. The limitations may impede the scalability of the framework especially when the number of tasks is large.

Future Work. Our future work will focus on proposing new CL methods, employing existing ones such as IMM [16], and conducting additional performance comparison experiments. Based on our VPWD architecture, replacing or combining EWC$^{\mathrm{H}}$ with other CL methods is convenient. Moreover, provided that the model capacity has not been reached, we can use this architecture to collect data for other website detection tasks like phishing website detection and let the model continue learning related features. In this way, our VPWD architecture can expand into a more general-purpose website detection architecture. Furthermore, we have released our dataset. In case other studies introduce CL-based VPW detection methods, we can incorporate them and conduct performance comparisons using our dataset.

6 Conclusion

In this paper, we design the ***first*** CL-based VPWD architecture and implement the system for the end-to-end process of VPW collection to detection. We build the ***largest*** real-world dataset of features targeting VPWs. We conduct empirical experiments on our dataset using CL algorithms and achieve an average accuracy of 96.4%, which is 0.5% higher than without continual learning algorithms.

Acknowledgments. The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 14222922, RGC GRF 2151185). We thank Chenlin Wang, the author of [28], for implementing the crawling toolkit, assisting with data collection in Section 3.1 and labeling in Section 3.2.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Akamai: State of the internet / security: Pirates in the outfield. https://www.akamai.com/resources/state-of-the-internet/soti-security-pirates-in-the-outfield (2022), [Online; accessed 24-Jan-2024]
2. Blackburn, D., Eisenach, J.A., Harrison, D.: Impacts of digital video piracy on the us economy. Nera Economic Consulting, Global Innovation Policy Center (2019)
3. Bridge, G.: Survey: 1 in 10 u.s. adults pirated tv, movies or live sports in 2022. https://variety.com/2023/biz/entertainment-industry/one-in-ten-us-adults-pirated-tv-movies-or-live-sports-in-2022-1235525708/ (2022), [Online; accessed 24-Jan-2024]
4. Choi, S.K., Kwak, J.: Feature Analysis and Detection Techniques for Piracy Sites. KSII Transactions on Internet & Information Systems **14**(5) (2020)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Proceedings of NAACL-HLT, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1423
6. Ejaz, A., Mian, A.N., Manzoor, S.: Life-long phishing attack detection using continual learning. Scientific Reports **13**(1), 11488 (Jul 2023). https://doi.org/10.1038/s41598-023-37552-9
7. Frick, S.J., Fletcher, D., Smith, A.C.: Pirate and chill: The effect of netflix on illegal streaming. Journal of Economic Behavior & Organization **209**, 334–347 (2023)
8. González-Carvajal, S., Garrido-Merchán, E.C.: Comparing BERT against traditional machine learning text classification. Journal of Computational and Cognitive Engineering **2**(4), 352–356 (Apr 2023)
9. Guo, Z., Zhang, Y., Zhang, Z., Xu, Z., King, I.: FedHLT: Efficient Federated Low-Rank Adaption with Hierarchical Language Tree for Multilingual Modeling. In: Companion Proceedings of the ACM Web Conference 2024. pp. 1558–1567. ACM, Singapore Singapore (May 2024). https://doi.org/10.1145/3589335.3651933
10. Hu, M., Li, M., Wang, Y., King, I.: Momentum contrastive pre-training for question answering. In: Goldberg, Y., Kozareva, Z., Zhang, Y. (eds.) Proceedings of EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022. pp. 4324–4330. Association for Computational Linguistics (2022). https://doi.org/10.18653/V1/2022.EMNLP-MAIN.291
11. Huszár, F.: On quadratic penalties in elastic weight consolidation. arXiv preprint arXiv:1712.03847 (2017), http://arxiv.org/abs/1712.03847
12. Huszár, F.: Note on the quadratic penalties in elastic weight consolidation. Proceedings of the National Academy of Sciences **115**(11) (Mar 2018). https://doi.org/10.1073/pnas.1717042115
13. Kim, E.J., Kwak, J.: Intelligent Piracy Site Detection Technique with High Accuracy. KSII Transactions on Internet & Information Systems **15**(1) (2021)
14. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., Hadsell, R.: Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences **114**(13), 3521–3526 (Mar 2017). https://doi.org/10.1073/pnas.1611835114
15. Kumar, J., Santhanavijayan, A., Janet, B., Rajendran, B., Bindhumadhava, B.: Phishing website classification and detection using machine learning. In: 2020 ICCCI. pp. 1–6 (2020). https://doi.org/10.1109/ICCCI48352.2020.9104161

16. Lee, S.W., Kim, J.H., Jun, J., Ha, J.W., Zhang, B.T.: Overcoming Catastrophic Forgetting by Incremental Moment Matching. In: Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017)
17. Li, M., Hu, M., King, I., Leung, H.F.: The Integration of Semantic and Structural Knowledge in Knowledge Graph Entity Typing. In: Duh, K., Gomez, H., Bethard, S. (eds.) Proceedings of NAACL-HLT (Volume 1: Long Papers). pp. 6625–6638. Association for Computational Linguistics, Mexico City, Mexico (Jun 2024). https://doi.org/10.18653/v1/2024.naacl-long.369
18. Li, Z., Zhang, S., Yin, J., Du, M., Zhang, Z., Liu, Q.: Fighting against piracy:an approach to detect pirated video websites enhanced by third-party services. In: 2022 IEEE Symposium on Computers and Communications (ISCC). pp. 1–7 (2022)
19. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017). https://doi.org/10.1109/TPAMI.2017.2773081
20. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. Advances in neural information processing systems **30** (2017)
21. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
22. McCloskey, M., Cohen, N.J.: Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In: Psychology of Learning and Motivation, vol. 24, pp. 109–165. Elsevier (1989)
23. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCaRL: Incremental classifier and representation learning. In: Proceedings of CVPR (Jul 2017)
24. Robertson, S.: Understanding inverse document frequency: On theoretical arguments for IDF. Journal of Documentation **60**(5), 503–520 (Oct 2004). https://doi.org/10.1108/00220410410560582
25. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. CoRR **abs/1606.04671** (2016), http://arxiv.org/abs/1606.04671
26. Shabudin, S., Samsiah, N., Akram, K., Aliff, M.: Feature Selection for Phishing Website Classification. International Journal of Advanced Computer Science and Applications **11**(4) (2020). https://doi.org/10.14569/IJACSA.2020.0110477
27. Statista: Online tv and movie revenue lost through piracy in 2016 and 2022, by country. https://www.statista.com/statistics/778342/global-online-tv-movie-revenue-loss-piracy-country/ (2023), [Online; accessed 27-Jan-2024]
28. Wang, C., Yu, Y., Pu, A., Shi, F., Huang, C.: Spotlight on Video Piracy Websites: Familial Analysis Based on Multidimensional Features. In: Knowledge Science, Engineering and Management, vol. 13370, pp. 272–288. Springer International Publishing, Cham (2022). https://doi.org/10.1007/978-3-031-10989-8_22
29. Wang, Z., Ng, P., Ma, X., Nallapati, R., Xiang, B.: Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In: Proceedings of EMNLP-IJCNLP. pp. 5878–5882. Association for Computational Linguistics, Hong Kong, China (Nov 2019), https://aclanthology.org/D19-1599
30. Yang, X., Song, Z., King, I., Xu, Z.: A Survey on Deep Semi-Supervised Learning. IEEE Transactions on Knowledge and Data Engineering **35**(9), 8934–8954 (Sep 2023). https://doi.org/10.1109/TKDE.2022.3220219
31. Yoon, J., Kim, S., Yang, E., Hwang, S.J.: Scalable and Order-robust Continual Learning with Additive Parameter Decomposition. In: ICLR (Apr 2020)
32. Zhang, S., Yin, J., Li, Z., Yang, R., Du, M., Li, R.: Node-imbalance learning on heterogeneous graph for pirated video website detection. In: 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD). pp. 834–840 (2022). https://doi.org/10.1109/CSCWD54268.2022.9776224