Stronger, Lighter, Better: Towards Life-Long Attribute Value Extraction for E-Commerce Products

Tao Zhang[1], **Chenwei Zhang**[2], **Xian Li**[2], **Jingbo Shang**[3], **Hoang H. Nguyen**[1], **Philip S. Yu**[1]
[1]University of Illinois at Chicago, Chicago, IL, USA;
[2]Amazon, Seattle, WA, USA;
[3]University of California, San Diego, CA, USA
{tzhang90,hnguy7,psyu}@uic.edu; {cwzhang,xianlee}@amazon.com; jshang@ucsd.edu

Abstract

Attribute value extraction involves identifying the value spans of predetermined attributes in product texts. This area of research has traditionally operated under a closed-world assumption, focusing on products from a static set of categories and their associated attributes. However, products in e-commerce stores are ever-increasing and evolving, calling for life-long learning. If continuously trained on the fast-increasing products and attributes, most existing solutions not only struggle for parameter efficiency but also endure foreseeable defects due to data contamination, catastrophic forgetting, etc. As a remedy, we propose and study a new task, which aims to effectively maintain a strong single model for many domains in a life-long learning fashion, without jeopardizing the model performance and parameter efficiency. We introduce factorization into the model and make it domain-aware by decoupling the modeling of product type and attribute, as a way to promote de-contamination and parameter efficiency while scaling up. Tuning the model with distillation prevents forgetting historical knowledge and enables continuous learning from emerging domains. Experiments on hundreds of domains showed that our model attains the near state-of-the-art performance with affordable parameter size, the least historical knowledge forgetting, and the greatest robustness against noises, whilst adding only a few parameters per domain when compared with competitive baselines.

1 Introduction

Attribute Value Extraction (AVE) (Nadeau and Sekine, 2007), similar to Named Entities Recognition (NER) (Chiticariu et al., 2010), aims to extract surface-form names (e.g. *red/vanilla/plastic*) of pre-defined product attributes (e.g. *Color/Scent/Material*) from product texts (e.g. product title/description). More specifically, we define each **domain** as the combination of one product category/type (PT, e.g. *Laundry_Detergent*) and one attribute (Attr, e.g. *Scent*) – extracting scent values from laundry detergent products is an example of a domain.

Training one model per domain (**One-for-Each**) means we need to maintain tens of thousands of models, which is not scalable and does not fully leverage the correlation between multiple domains jointly. Therefore, training a single model for all domains (**All-in-One**) attracts intensive studies (Zheng et al., 2018; Xu et al., 2019; Karamanolakis et al., 2020; Wang et al., 2020; Yan et al., 2021; Yang et al., 2022). These works aim to train a scalable all-in-one model that absorbs stacked data from all domains in a closed-world fashion. They achieve decent performance while assuming a fixed list of pre-defined domains is available. However, products in the real world keep evolving in an open-world fashion. New attributes, new values, and new product types necessitate effective knowledge transfer and life-long adaptation, without the need to retrain the entire model from scratch.

Effective parameter sharing and parameter efficiency serve critical roles in such knowledge transfer and scalable adaptation across many domains. Intuitively, similar PTs can share the same attributes and values. For example, *Laundry_Detergent* and *Hand_Sanitizer* may both have *cedarwood* as the *Scent* value. The approach of fully shared parameters does not always result in the desired knowledge transfer and can sometimes negatively impact performance, especially when dealing with complex and evolving data. For example, different PTs possess drastically different values on the same attribute: *Size* for *Shoes* can be *7/7.5/8* and *Size* for *Shirt* takes values such as *XS/S/M/L/XL*. Also, the same value can be adopted by different domains: *Orange* can be a typical *Flavor* value for *Drink* products, while the same value is usually mentioned in *Umbrella* products as a *Color*. Although existing works (Xu et al., 2019;

Findings of the Association for Computational Linguistics: ACL 2024, pages 8631–8643

Yan et al., 2021; Yang et al., 2022) started to adopt various techniques to add additional inputs to make the model better aware of the domain it is handling and therefore alleviate domain contamination and facilitate knowledge transfer, we argue that because model parameters are essentially accessible to all domains, such fine-grained, domain-specific knowledge may find it hard to be stored organically in the form of loosely organized model parameters, exactly according to how the knowledge distributes and evolves.

For the first time, we study the open-world life-long AVE task that requires model capability on both parameter efficiency for scale-up extraction and life-long extraction of evolving new data. We developed a unified solution, **DALLA** - Domain-Aware Life-Long Adapter. To alleviate the domain contamination without sacrificing parameter efficiency, we propose a factorized adapter to decouple the product type and the attribute by assigning them with smaller sub-matrices after performing low-rank decomposition on model weight matrices. Factorization not only facilitates parameter savings but, more importantly, decouples the PT/Attr aspects, allowing them to operate independently while maintaining the model's awareness of the entire domain. Weight reconstruction only utilizes specific product type and attribute sub-matrices. This enables parameter sharing on a fine-grained (PT/Attr) level instead of shared across all domains, and strengthens domain-relevant knowledge transfers. Furthermore, we introduce a life-long distillation scheme based on the factorized adapter, to effectively boost underperforming or totally new domains while avoiding catastrophic forgetting of existing well-performing domains.

Our main contributions are summarized as:

- To our knowledge, **DALLA** is the first work to study the open-world life-long AVE task and the first time to introduce model parameter saving for scale-up and life-long attribute value extraction.
- We propose the domain-aware factorized adapter module, along with its distillation strategy, to enable an Attribute Value Extraction (AVE) model that is scalable on vast and expanding data. The proposed model is robust against catastrophic forgetting, domain contamination, and noisy data, while also achieving performance on continually unseen data.
- We conduct extensive experiments to show our superiority in parameter efficiency and state-of-the-art robustness performance on the open-world AVE task compared with prevalent and latest baselines.

2 Domain-Aware Life-long Adapter

Problem formulation: We address the attribute value extraction (AVE) problem by regarding it as a sequential tagging problem, as a popular choice. Based on the BIO schema , AVE assigns a sequence of labels $Y = [y_1, ..., y_{N_{token}}]$ to a given product profile sentence $X = [x_1, ..., x_{N_{token}}]$ with N_{token} tokens, given a product type (PT) and an attribute (Attr). We take the combination of a product type and an attribute as a domain $(PT, Attr)$. Given the domain, an AVE model should mark out all the entities in the product profile as attribute values. An entity is a span of tokens $\mathbf{e} = [x_i, ...x_j](1 \leq i \leq j \leq N_{token})$. Specifically, the first token of an entity mentioned in the sentence is labeled as B, the other tokens inside that entity mention are labeled as I, and the non-entity tokens are labeled as O.

Open-World Life-Long AVE Task: Previous works (Wang et al., 2020; Yan et al., 2021; Yang et al., 2022) show remarkable performances on closed-world data but fail to address the concern of life-long learning capabilities on the open-world data, which keeps evolving on either attribute value or domain set. Similar to most previous works, directly retraining with open-world data leads to catastrophic forgetting.

Unlike closed-world settings, for open-world life-long AVE task, we are inspired by (Ke and Liu, 2022) and seek to incrementally learn a model f with T sequential domains $\mathbb{D} = \{\mathcal{D}_i\}_{i=1}^{T}$: $f : X \times \mathbb{D} \rightarrow Y$, namely a sequential learning process $(f(X, \mathcal{D}_1) \rightarrow Y^{\mathcal{D}_1}, ..., f(X, \mathcal{D}_T) \rightarrow Y^{\mathcal{D}_T})$. Each domain is regarded as a unit dataset absorbed by f for training and testing.

An ideal solution for the open-world, life-long AVE task requires: (1) Scalability: the model can provide cost-effective training on large-sized and number-increasing domains. We seek parameter efficiency by a domain-aware factorized adapter, depicted in Section 2.1. (2) Robustness: the model can overcome catastrophic forgetting and domain contamination during evolving training. We decouple sub-matrices for PTs and Attr for conditional parameter sharing and allow each PT/Attr sub-matrix to update independently to avoid contamination and propagation, detailed in Section 2.1. Furthermore, we distill knowledge through

the factorized sub-matrices for knowledge forgetting prevention and similar PTs/Attrs association, presented in Section 2.2. (3) Generalizability: the model is applicable to unseen data. We will use domain-aware sub-matrices to enable unseen domain extraction in Section 2.2.

2.1 Parameter Efficiency via Domain-Aware Factorized Adapter

ALBERT (Lan et al., 2019) is a compressed BERT (Devlin et al., 2018). ALBERT employs compression techniques, including (1) vocabulary embedding layer factorization; and (2) training one transformer encoder layer, and subsequently sharing the encoder parameters among the remaining layers. To compensate for the performance sacrifice of ALBERT after compression, we choose ALBERT$_{xlarge}$ as the basic model, wedged the proposed domain-aware factorized adapter, shown in the right part of Figure 2. Therefore, DALLA can retain ALBERT's compression strategy as well as enjoy expressive contextual embedding compatible with BERT. Furthermore, to alleviate the model training burden, we follow the adapter tuning (Houlsby et al., 2019; Mahabadi et al., 2021) by freezing the original model while only tuning the adapter module.

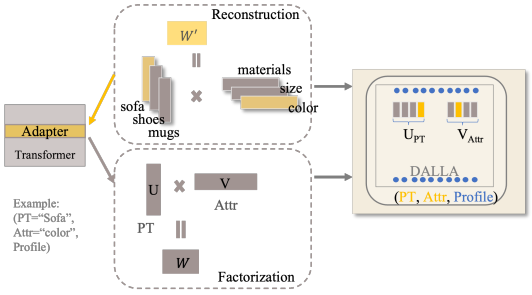

Figure 1: Adapter layer factorization and reconstruction.

Factorization for Parameter Saving: Weight factorization plays an important role in model compression (Panahi et al., 2021; Chen et al., 2021), which uses singular value decomposition (SVD) to compress the learned model weights. With SVD, the learned matrix is factorized into three matrices (U, S, V), where S is usually a non-trainable identity matrix and absent during training. The multiplication of these smaller matrices will approximate the original one with fewer total parameters to achieve the model parameter saving. Given a matrix $W \in \mathbb{R}^{H_i \times H_o}$, we use the low-rank approximation via SVD: $W \approx UV^\top$, where $U \in \mathbb{R}^{H_i \times r}$, $V \in \mathbb{R}^{r \times H_o}$, and r is the rank of matrix W, H_i and H_o are the input hidden embedding size and output hidden embedding size. The reduced size will be $H_i H_o - (H_i r + H_o r)$. For all domains included in the model training dataset, we construct sub-matrices for all mentioned PT and $Attr$ as $\mathcal{U}_{PT} = \{U_{PT_i}\}_{i=1}^{N}$ and $\mathcal{V}_{Attr} = \{V_{Attr_j}\}_{j=1}^{M}$, where N and M are the numbers of PTs and Attrs. For a single domain $(PT_i, Attr_j)$, the reconstruction of W (shown in Figure 1), denoted as $W' = U_{PT_i} V_{Attr_j}^\top$, plays the same role with W in the original network.

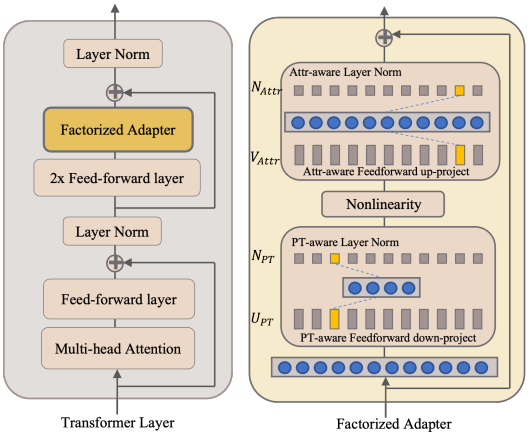

Figure 2: Factorized adapter layer integration.

If the model is facing scale-up domains, we can maintain existing sub-matrices for well-performing domains or add new sub-matrices corresponding to the new domains. The creation of multiple PTs and Attrs is shown in Figure 1. We first construct and randomly initialize sub-matrices for all PTs (e.g. *sofa*, *shoes*, *mugs*) and Attrs (e.g. *materials*, *size*, *color*). When taking in the sample $\{$"*sofa*", "*color*", *profile*$\}$, the model only learns sub-matrix U_{sofa} and V_{color} rather than the original parameter layer W. Moreover, the sub-matrices of the other PTs and Attrs are kept untouched until the model deals with samples associated with them.

Factorized Adapter Integration: To strike a balance between the efforts invested in model training and the need for rapid adaptation to various tasks, researchers (Houlsby et al., 2019; Mahabadi et al., 2021) have proposed the integration of small modules, known as adapter layers, into a pre-trained model. These adapter layers serve as a middle ground between full fine-tuning and keeping the pre-trained weights fixed. The weights of the original network are untouched, whilst the new adapter layers are initialized at random. In adapter-tuning,

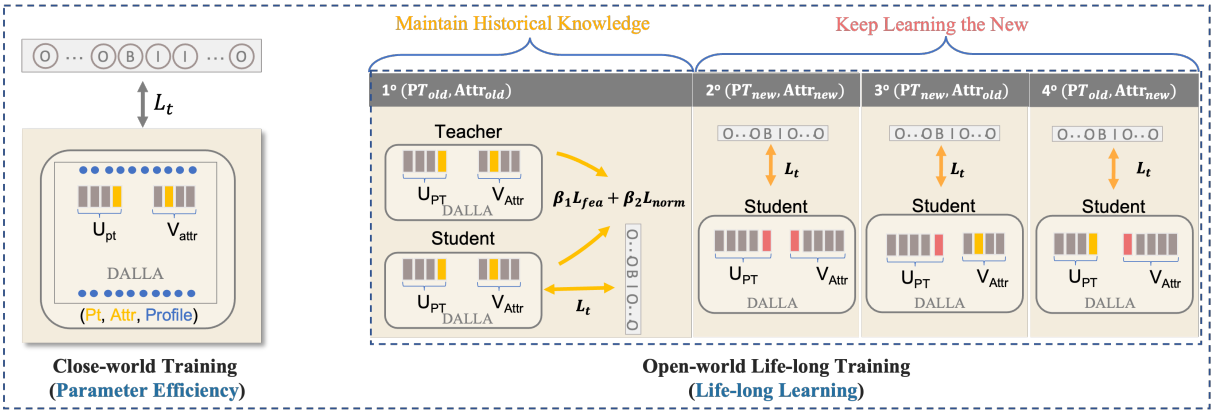

Figure 3: Illustration of the two-stage pipeline training. In the closed-world training, the model is trained on seen-domain samples and only updates the domain-relevant sub-matrices (highlighted in the yellow box), along with their corresponding layer norm. The remaining sub-matrices are kept unchanged. In the open-world life-long training, the model deals with new data. A sub-matrix update scheme for four types of new domain cases is employed to demonstrate how the model maintains historical knowledge while continuously learning from the new data. Newly created sub-matrices are marked in red, while the existing sub-matrices from seen domains are indicated in yellow. The grayed sub-matrices are irrelevant PTs and Attrs, so they are not activated under each setting.

the parameters of the original network are frozen and therefore shared. This means that the total model size grows relatively slowly when more domains are added. As shown in the left part of Figure 2, the trainable parameters are illustrated in yellow. When faced with a given sample, only the factorized adapter consisting of the corresponding sub-matrix and layer norm of its PT and Attr is trainable.

As shown in Figure 2, we conduct SVD on the single adapter layer, W, and replace it with domain-aware sub-matrices, $(\mathcal{U}_{PT}, \mathcal{V}_{Attr})$ for domain-aware W' restoration. $\mathcal{U}_{PT}$, and $\mathcal{V}_{Attr}$ are a set of weights to squeeze domain-aware information through a small-size intermediate layer. As discussed in the previous works (Choi et al., 2020; Liu et al., 2021), layer norms in the pre-trained model store the mean and variance of the layer input, which can be utilized as a proxy to associate historical knowledge for life-long learning. To capture fine-grained association, we introduce the factorized norm layer sets $\mathcal{N}_{PT} = \{N_{PT_i}\}_{i=1}^{N}$ and $\mathcal{N}_{Attr} = \{N_{Attr_j}\}_{j=1}^{M}$ for $\mathcal{U}_{PT}$, and $\mathcal{V}_{Attr}$, respectively. N_{PT_i} is stacked right after U_{PT_i} to record the PT_i's input statistics, so as to N_{Attr_j} and V_{Attr_j}.

2.2 Life-Long Learning via Adaptive Distillation

We cast DALLA's training strategy on open-world life-long AVE into a two-stage process, as shown in Figure 3: (1) closed-world training stage, where we first feed the domain-aware factorized adapter with a number of seen domains that are pre-defined in a closed-world fashion; (2) open-world life-long training stage, where we conduct the adaptive distillation on unseen domains in an open-world fashion to mimic the evolving domains. To avoid catastrophic forgetting and benefit from similar PTs/Attrs associations, we adopt knowledge distillation based on the corresponding PTs/Attrs sub-matrix, which can further alleviate the cold start issue faced with new domains. The distillation allows parameter sharing conditioned on specific PT/Attr which prevents the domain contamination by only allowing relevant sub-matrix updates while leaving the remaining sub-matrices untouched.

We denote $f_{\hat{\theta}}$ as the model learned in closed-world training stage by

$$\mathcal{L}_{\text{t}}(\mathcal{D}, X, Y) = \text{Entropy}(Y, f_{\hat{\theta}(U_{PT}, V_{Attr})}(X)), \quad (1)$$

where $\mathcal{D} = (PT, Attr)$. $\hat{\theta}(U_{PT}, V_{Attr})$ means parameter updates on PT and Attr sub-matrices related to the domain $(PT, Attr)$. For the open-world life-long training stage, we initialize the teacher and the student model as $f_{\theta_{tea}} = f_{\theta_{stu}} = f_{\hat{\theta}} = \arg\min_{\hat{\theta}} \mathcal{L}_{\text{t}}$.

To better understand the association and distinction on both coarse-grained (domain) level and fine-grained (PT/Attr) level, the distillation loss

contains two parts: (1) domain feature transfer:

$$\mathcal{L}_{\text{fea}}(X) = \frac{1}{N}\sum_{n=1}^{N} -f_{\theta_{tea},n}(X)\log f_{\theta_{stu},n}(X), \tag{2}$$

where $X \in \mathcal{X}$, containing N entities. $f_{\cdot,n}(X) = f_{\cdot_{(U_{PT},V_{Attr})},n}(X)$ means the output of entity n given a domain $(PT, Attr)$; (2) factorized norm layer transfer:

$$\mathcal{L}_{\text{norm}}(\Sigma_x, \Sigma'_x) = \frac{(\mu_x - \mu'_x)^2 + \sigma_x^2}{2\sigma'^2_x} - \log\frac{\sigma_x}{\sigma'_x} - \frac{1}{2}, \tag{3}$$

which is the Kullback-Leibler (KL) divergence of two Gaussian distributions to match the statistics $\Sigma' = (\mu', \sigma')$ stored in the norm layers of the teacher and the statistics $\Sigma = (\mu, \sigma)$ computed for the student output at the same norm layers of the teacher, where $x \in \{\mathcal{N}_{PT} \bigcup \mathcal{N}_{Attr}\}$. The updated student model is $\hat{\theta}_{stu} = \arg\min_{\theta_{stu}} \alpha\mathcal{L}_{\text{t}} + \beta_1\mathcal{L}_{\text{fea}} + \beta_2\mathcal{L}_{\text{norm}}$, where α, β_1, and β_2 are coefficients.

When considering the new data for life-long learning, we first distinguish four different cases of unseen domains, as shown in the right part of Figure 3. We adopt different distillation schemes adaptively for each case:

(1) $(PT_{old}, Attr_{old})$, a seen domain with new values of existing domains or a new combination of existing PT/Attr. Distillation associates the historical knowledge with new values, or help the model adapt to this new combination, through losses denoted in Eq. 2 and 3.

(2) $(PT_{new}, Attr_{new})$, an unseen domain with a new PT and a new Attr. New PT and Attr sub-matrices are created and updated without overriding any already learned knowledge using the objective in Eq. 1.

(3) $(PT_{new}, Attr_{old})$, an unseen domain with a new PT and an existing Attr. Since we should create a new sub-matrix for the PT, we cannot directly distill the knowledge from the teacher but train the student with the objective denoted in Eq. 1. However, sharing the Attr sub-matrix can alleviate cold start problem even though we randomly initialize the new PT sub-matrix.

(4) $(PT_{old}, Attr_{new})$, an unseen domain with a new Attr and an existing PT. Similar to the case above, we employ parameter sharing conditioned on PT to expedite model optimization. The model only needs to put effort into learning the new Attr plus subtle updates on the PT sub-matrix.

Models	%Para	Macro-F1 (P|R)	Micro-F1 (P|R)
All-in-One			
ALBERT	**0.10**×	64.1(66.0|62.2)	64.3(65.7|62.9)
ALBERT$_{xlarge}$	<u>0.55</u>×	70.0(72.1|68.0)	71.2(73.8|68.7)
BERT	1.00×	71.0(73.0|69.1)	71.0(73.5|68.8)
ALBERT$_{ft\sharp}$	4.25×	61.8(65.4|58.4)	61.7(64.5|59.2)
BERT$_{ft}$	0.68×	<u>72.4</u>(73.9|<u>71.0</u>)	<u>73.8</u>(75.1|<u>72.7</u>)
SUOpenTag	2.01×	71.1(<u>74.2</u>|68.1)	72.1(<u>75.2</u>|69.2)
DALLA	0.74×	**74.0**(**75.6**|**72.5**)	**75.1**(**76.4**|**73.8**)
One-for-Each			
BERT$_{Ada}$	42.72×	76.5(78.9|74.2)	76.9(78.9|75.0)
ALBERT$_{Ada}$	19.09×	75.4(77.2|73.6)	77.5(79.1|76.0)

Table 1: Performance of all models. %Para indicates the percentage of trainable parameters. We take BERT (110M) as 1.0. The bolded text indicates the best score, while the underline is used by the second-best model.

3 Experimental Settings

Dataset and Baselines Details about dataset are interpreted in Appendix A. We compare our model with two groups of baselines, including the All-in-One and the One-for-Each. All baselines and the implementation details are presented in Appendix C and Appendix B. We additionally discuss the parameter saving estimation after implementing SVD on the different model components in Appendix D.

4 Evaluation of Parameter Efficiency

Model Effectiveness and Efficiency We evaluate the model parameter efficiency in Table 1 via discussion on three comparisons: (1) Different backbone model (ALBERT, BERT, ALBERT$_{xlarge}$) comparison necessitates the trade-off between parameter saving and decent performance; (2) Whether and where to wedge the factorized layer (BERT vs. BERT$_{ft}$ and ALBERT vs. ALBERT$_{ft\sharp}$, BERT$_{ft}$ vs. ALBERT$_{ft\sharp}$) raise the concern that the position for SVD indeed affects the parameter efficiency because single factorized adapter layer in BERT$_{ft}$ surpasses BERT, ALBERT and ALBERT$_{ft\sharp}$; (3) Scale-up techniques comparison (Adapter in BERT$_{ft}$ vs. Attention in SUOpenTag) reveals the superiority of factorized adapter. From all of the above comparisons, we observe that techniques employed in DALLA(ALBERT$_{xlarge}$ backbone model with single factorized adapter layer) address both the trade-off concern and scale-up advantage effectively.

From Table 1, DALLA attains almost the best on most of the evaluation metrics, especially for the Marcro-F1 and Micro-F1 with great margin against

the second best to be 1.6% and 1.3%. Apart from showing superiority among the All-in-One group, our model exhibits competitive performances to the two methods from the One-for-Each group, which are regarded as the upper bounds. Compared with BERT and ALBERT$_{xlarge}$, better achievements in DALLA and the $BERT_{ft}$ indicate the benefits from factorization to decouple PT and Attr allowing conditional parameter sharing. Therefore, the association and distinction among domains can be well captured and uncovered through the update or distillation of the PT-specific/Attr-specific submatrices. The observed increase in parameters of ALBERT$_{ft\sharp}$ is an intentional outcome, as excessive use of factorization can indeed lead to issues. As depicted in Table 6, when factorization is applied to Query, Key, Value, Dense, and FFN components in the transformer encoder, the parameter size actually expands rather than contracts.

Figure 4 reflects the trade-off between parameter efficiency (performance over per trainable parameter) and performance achievements across different model variations. All models in One-for-Each are located in the bottom-right corner, which implies they are good at achieving decent performance since model training is tailored for each domain. In the group of All-in-One, ALBERT seems to have a great advantage over the others in parameter efficiency due to its model compression strategy. However, it sacrifices too much performance (the worst F1 in Table 1). DALLA and BERT$_{ft}$ are the first and the second best models obtaining the trade-off between performance and parameter efficiency in Figure 4. They all are qualified to be scale-up models. But under the setting of open-world life-long AVE, evolving data challenges the stability of most models in the All-in-One, because of a series of issues including weight overriding, catastrophic forgetting, and domain contamination when a single model is shared by all domains.

5 Evaluation of Life-Long Adaptation

Alleviating Catastrophic Forgetting We first study DALLA's capability of overcoming catastrophic forgetting and domain contamination. To evaluate catastrophic forgetting alleviation, we split the domains into seen and unseen domains. Initially, we train the model on the seen domains during the closed-world training stage. Subsequently, we proceed with open-world training by conducting n=3 consecutive training sessions on expanding sets of disjoint unseen domains. We then record the performance drops on **seen domains** after each set of open-world training sessions. There are two settings in the open-world training, one is when no additional seen domain data is available ($N_R = 0\%$, no additional seen samples; unseen data only). We aim to use this setting to test the direct knowledge transfer capability to unseen domains. The other setting also includes new values from seen domains ($N_R = 10\%$, 10% additional seen samples). We aim to use this setting to test if the model can be assertive in learning new knowledge while not forgetting the knowledge on the existing seen domains.

Figure 4: Model parameter efficiency. The top-right corner is the ideal area. The scale/color of each circle is calculated using the multiplication of each model's performance rank (x-axis) and parameter efficiency rank (y-axis).

From the left part of Table 2, DALLA shows its superiority in controlling the forgetting score after several runs on unseen domains with the lowest scores in average forgetting. Although BERT$_{ft}$ shows comparable performance with DALLA in terms of parameter efficiency, BERT$_{ft}$ along with other baselines suffer from severe catastrophic forgetting because a great portion of parameters is shared. When we increasingly add new domains, DALLA still maintains decent performance on seen domains, which shows its advantage in overcoming domain contamination, compared with the others. When comparing the performance of models with $N_R = 0\%$ and $N_R = 10\%$, it is important to note that for the latter, all models tackle the forgetting issue by retrieving their knowledge from seen domain samples. However, DALLA surpasses the other models and achieves the best performance in terms of preserving performance.

Handling Various Unseen Cases We evaluate the model's life-long adaptation ability on unseen domains, as shown in the right part of Table 2. Based on the four types of unseen domains, discussed

	N_R	Macro-F1 on all seen domains & average forgetting ↓ Δ				Macro-F1 on four types of unseen domains			
Stage		Closed-World	Open-World-1	Open-World-2	Open-World-3	$(PT_{old}, Attr_{old})$	$(PT_{new}, Attr_{new})$	$(PT_{new}, Attr_{old})$	$(PT_{old}, Attr_{new})$
# domains		(400)	(100 unseen)	(200 unseen)	(300 unseen)	(10 unseen)	(10 unseen)	(10 unseen)	(10 unseen)
BERT	0%	77.21	74.38 ↓ Δ2.83	71.19 ↓ Δ6.02	68.54 ↓ Δ8.67	76.83	82.69	70.74	76.02
BERT$_{ft}$	0%	79.06	77.09 ↓ Δ1.97	75.07 ↓ Δ3.99	72.86 ↓ Δ6.20	80.93	84.44	75.22	78.03
SUOpenTag	0%	78.32	76.15 ↓ Δ2.17	74.31 ↓ Δ4.01	71.28 ↓ Δ7.04	78.57	83.94	73.84	77.11
MAVEQA	0%	**92.36**	90.83 ↓ **Δ1.53**	88.27 ↓ Δ4.09	86.00 ↓ Δ6.36	**97.24**	**91.20**	**80.49**	**88.71**
DALLA	0%	79.15	77.48 ↓ Δ1.67	75.85 ↓ **Δ3.30**	74.35 ↓ **Δ4.80**	81.36	85.07	76.49	79.27
BERT	10%	77.21	75.43 ↓ Δ1.87	72.14 ↓ Δ5.07	70.99 ↓ Δ6.22	75.44 ↓ Δ1.39	82.14 ↓ Δ0.55	69.36 ↓ Δ1.38	76.05 ↑ Δ0.03
BERT$_{ft}$	10%	79.06	78.39 ↓ **Δ0.67**	76.72 ↓ Δ2.34	75.51 ↓ Δ3.55	80.75 ↓ Δ0.18	85.02 ↑ Δ0.58	75.39 ↑ Δ0.17	78.76 ↑ Δ0.73
SUOpenTag	10%	78.32	77.02 ↓ Δ1.30	75.03 ↓ Δ3.29	73.47 ↓ Δ4.85	77.02 ↓ Δ1.55	84.62 ↑ Δ0.68	73.65 ↓ Δ0.19	77.35 ↑ Δ0.24
MAVEQA	10%	**92.36**	90.99 ↓ Δ1.37	89.33 ↓ Δ3.03	86.85 ↓ Δ5.51	97.33 ↑ Δ0.09	91.07 ↓ Δ0.13	81.14 ↑ **Δ0.65**	87.62 ↓ Δ1.09
DALLA	10%	79.15	78.27 ↓ Δ0.88	77.28 ↓ **Δ1.87**	76.57 ↓ **Δ2.58**	82.84 ↑ **Δ1.48**	85.93 ↑ **Δ0.86**	76.88 ↑ Δ0.39	80.25 ↑ **Δ0.98**

Table 2: Average macro F1 and average forgetting on all seen domains (left) and life-long learning on four different types of unseen domains (right). Left: for each model, we conduct closed-world training on 400 seen domains and we continue the model training by progressively introducing accumulating unseen domains. This process is repeated three times, with each iteration adding 100 additional distinct domains. N_R is the portion of reserved samples in each seen domain. ↓ Δ denotes the performance drop on seen domains. Right: we train the models on 100 seen domains and retrain on 10 unseen domains for each case as described in 2.2. ↓ means performance drop from $N_R = 0\%$ to 10% on each model, while ↑ indicates the performance gain.

in Figure 3, we present two groups of the results: direct transfer to the unseen ($N_R = 0\%$) and transfer on both unseen domains and new values of seen domains ($N_R = 10\%$). For the first group results, DALLA shows the second-best performance among other baselines, because MAVEQA, originated from where the dataset comes, achieves nearly 99% on all attributes as reported in the paper (Yang et al., 2022). We discussed more details about MAVEQA in the Appendix C.

The second group exhibits not only the macro-f1 scores but also the performance changes considering the effect of adding in a small portion of seen domain examples. DALLA improves the performance while the other baselines suffer from decreased performance in the case of $(PT_{old}, Attr_{old})$. This suggests that the other baselines suffer from contamination from the seen domain samples, attempting to maintain performance, let alone strengthen desired knowledge transfer to enhance the performance of the domains that consist of existing PTs and Attrs. We see BERT and MAVEQA loss score in the case of $(PT_{new}, Attr_{new})$. The others use their proposed domain-aware component to build specific knowledge for the totally new domains, while they directly concatenate PT and Attr with the model input is insufficient to capture the current domain well. Compared with BERT$_{ft}$ and SUOpenTag, DALLA still shows the superiority of domain-aware ability due to independent updates on PT/Attr sub-matrices. In the remaining two cases, $(PT_{new}, Attr_{old})$, and $(PT_{old}, Attr_{new})$, we find learning on new PTs with existing Attrs is harder than new Attrs with existing PTs according to smaller gains and performance drop shown in the case of $(PT_{new}, Attr_{old})$. In fact, the value form for the same Attr among different PTs can be dramatically different. e.g. *Size* for *Shoes* should be numerical like *7/7.5/8*, while *Size* for *Shirt* takes values in characters such as *XS/S/M/L/XL*. Moreover, we observe from the data that many attributes can be shared by tens to hundreds of product types but much sparser for product types to attributes, which implies learning well on Attr sub-matrices is harder than that on PT sub-matrices. If the Attr sub-matrix is underfitting, directly adopting it as initialization still requires huge efforts in model training instead of providing a beneficial warm-up.

6 Analysis

Robustness to Noises We further study a problem on model robustness to noisy training values in Table 3. We conduct noise manipulation on the dataset, including randomly removing values and sliding BIOE tags (with a window size of 1) on both 10% seen and 10% unseen data. Compared with SUOpenTag on both seen, unseen, and overall domains, DALLA shows 0.9% - 1.91% lower performance drop, even double efforts in minimizing the error propagation.

Sub-Matrix Visualization We visualize PT/Attr sub-matrices to uncover the associations among PTs/Attrs. Note that those sub-matrices start with randomly initialized parameters and are learned in a purely data-driven fashion, rather than being assigned manually or under any assistance of their PT/Attr name semantics. From Figure 5(top), the

Models	Overall	Seen	Unseen
SUOpenTag	66.88|70.04	71.29|73.04	63.94|67.99
w. Noise	-4.04|-4.41	-6.03|-6.19	-2.19|-3.44
DALLA	69.03|75.38	74.38|77.39	66.48|72.51
w. Noise	**-2.13|-3.20**	**-4.29|-4.89**	**-2.10|-3.34**

Table 3: Robustness to the noise data. Results are performance drops after adding in noise data under seen, unseen, and overall domains with their Macro-F1|Micro-F1 scores. The less is the better for error robustness.

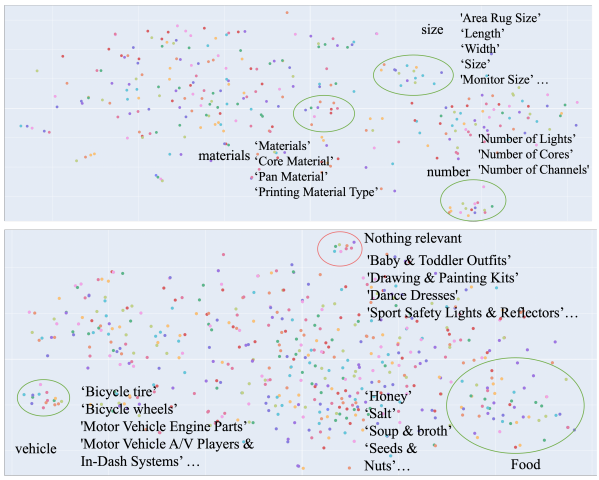

Figure 5: Visualization of Attribute (top) and Product Type (bottom) sub-matrices. Desired clusters are circled in green, while the red one is an abnormal cluster.

clusters marked in the green circle indicate well-learned attribute type semantics (e.g. "Materials", "Pan Materials", and "Core Materials"), via learning to extract their values. In Figure 5(bottom), we find that similar PTs are in proximity to each other (e.g. a cluster of food PTs in the lower right corner), which is very encouraging given no explicit prior knowledge of PT name semantics. We also observe that some irrelevant PTs in the red circle are close to each other (e.g. "Baby and Toddler Outfits" and "Drawing and Painting Kits"), mostly due to random initialization and lack of proper training on those PTs when very few data points are available. We expect more discriminative PT embeddings could be obtained with more incoming data, given our life-long learning setting.

6.1 Ablation study on adaptive distillation

In this section, we discuss the importance of distillation and what should be distilled by ablation study. As shown in Table 4, we test the distillation ability by measuring forgetting that occurs on average in the seen domains after we distill different parts (norm layer, sentence feature embedding, encoder logits, and the weight sub-matrices) to distill. The impact of distilled knowledge from feature embedding and norm layer on the performance is outstanding among all alternatives, which indicates their importance in incremental open-world AVE. Therefore, we test the effectiveness of their combination and report the best score in the last row of Table 4.

	Average forgetting (%)		
Stage # domains	Open-World-1 (100 unseen)	Open-World-2 (200 unseen)	Open-World-3 (300 unseen)
w.o. L_d	-3.02	-5.31	-6.55
w. L_{norm}	-2.25	-3.87	-5.34
w. L_{fea}	-1.83	-3.36	-4.93
w. L_{logit}	-2.66	-4.08	-5.78
w. L_w	-3.00	-4.91	-6.93
w. $L_{norm+fea}$	**-1.67**	**-3.30**	**-4.80**

Table 4: Average forgetting under different distillation methods. w.o. L_d is without the usage of any distillation loss. w. L_* means employing different parts (norm layer, sentence feature embedding, encoder logits, and the weight sub-matrices) in the model to calculate the distillation loss.

6.2 Error Analysis

After ranking all domains' performance, we observe two major error types. First, since training samples are imbalanced among domains, some domains consist of a limited number of samples, resulting in their relevant PT and Attr sub-matrices being inadequately learned. These domains usually suffer overwriting by other predominant domains. Second, attribute complexity affects the performance, because some attributes participate in multiple domains, such as "size" and "type". Since we maintain a uniform sub-matrix size for each attribute, those attributes with diverse surface names across different PTs present challenges. Learning a versatile Attr sub-matrix capable of accommodating all variations of surface names proves to be difficult, particularly for attributes like "size" and "type".

We conduct experiments to show even suffering errors DALLA still can control them compared with other baselines, as shown in Table 5 and Table 6, DALLA exhibit its superiority with lowest performance drop when handling rare or complex domains on open-world scenarios.

	Average forgetting (%)			
Stage # domains	Close-world (rare domains)	Open-World-1 (20 unseen)	Open-World-2 (30 unseen)	Open-World-3 (40 unseen)
DALLA	57.39	**-3.28**	**-4.22**	**-6.07**
MAVEQA	64.39	-3.92	-5.44	-8.25
SUOpenTag	53.29	-4.02	-5.97	-9.01

Table 5: Error Control on rare domains. Rare domains are those with $\#sample \in [100, 150]$

	Average forgetting (%)			
Stage # domains	Close-world (complex domains)	Open-World-1 (20 unseen)	Open-World-2 (30 unseen)	Open-World-3 (40 unseen)
DALLA	64.39	**-2.19**	**-3.48**	**-5.24**
MAVEQA	67.85	-3.11	-4.27	-6.66
SUOpenTag	59.36	-3.92	-6.08	-8.95

Table 6: Error control on complex domains. Complex domains comprise attributes with a high diversity of distinct values, typically ranking within the top 10 in terms of variety.

7 Related Work

Attribute Value Extraction Sequence tagging is one of the popular choices for attribute value extraction, where existing works are mostly improving scalability and effectiveness when faced with large-volume data. SUOpenTag (Xu et al., 2019) employs BERT as the encoder to capture profile embedding and attributes. TXtract (Karamanolakis et al., 2020) encodes the hierarchical embedding of product type for pt-specific extraction. AdaTag (Yan et al., 2021) proposes a mixture of expert and hypernetwork to capture the attribute-aware features. AVEQA (Wang et al., 2020) regards AVE as a question-answering task extracting the answer span. MAVE (Yang et al., 2022) devises the global-local connection to enable the model to be aware of different parts of the profile. Shinzato (Shinzato et al., 2022) designs to expand queries for effectively improving rare attributes' performance. Even though these models can handle huge data, it is still underneath the iceberg to maintain model stability and avoid catastrophic forgetting when faced with incremental data in an open-world fashion.

Adapter for Parameter Efficiency Incorporating adapter layers into neural networks (Houlsby et al., 2019; Mahabadi et al., 2021; Hu et al., 2021) has garnered significant attention, particularly in the era of large language models (Touvron et al., 2023; Chowdhery et al., 2023; Ye et al., 2023). As the size of model parameters experiences exponential growth with the evolution of language models, employing an over-parameterized neural network for downstream tasks necessitates leveraging advantageous low-rank properties during the training process to facilitate rapid adaptation. The low-rank factorized adapter can save trainable parameter size to speed up model training. We further adjust the usage of the low-rank matrix by implementing task-specific semantic associations (PTs and Attrs) to scale up model capacity and foster life-long adaptability.

8 Conclusion

In this paper, we study the open-world life-long attribute value extraction task and devise a factorized life-long adapter module to enable model scalability and robustness. Factorized adapter decouples product type and attribute, allowing domain interaction on either PT-level or Attr-level. Therefore, the extraction model is domain-aware but updated independently for de-contamination. Moreover, we endow the model with life-long adaptation ability by distilling the decoupled information to avoid catastrophic forgetting. DALLA can achieve decent performance on the benchmark with the ability to control forgetting and error propagation and only consume affordable parameters.

9 Limitations

After decoupling PT and Attr, we can obtain domain associations through parameter sharing conditioned on PT/Attr to warm up model training on unseen domains that have associated PTs/Attrs. To learn solid associations, we expect each sub-matrix to be well learned by sufficient data points. Otherwise, the knowledge transfer and performance on unseen domains with associated PTs/Attrs towards worse.

Furthermore, even if the model is unaware of the specific names of PTs (Product Types) and Attrs (Attributes), DALLA is capable of capturing the relationships inside PTs and Attrs. To alleviate the learning burden on the model, in future work, we investigate a hypernetwork approach consistent with lifelong learning. This approach encodes the semantic meaning of PTs and Attrs into the creation of sub-matrices, enabling the model to leverage this encoded information during open-world life-long training.

10 Acknowledgement

We thank the anonymous reviewers for their constructive feedback which we incorporated in the final version of this manuscript. This work is supported in part by NSF under grant III-2106758.

References

Patrick Chen, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. 2021. Drone: Data-aware low-rank compression for large nlp models. *Advances in neural information processing systems*, 34:29321–29334.

Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. Domain adaptation of rule-based annotators for named-entity recognition tasks. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1002–1012.

Yoojin Choi, Jihwan Choi, Mostafa El-Khamy, and Jungwon Lee. 2020. Data-free network quantization with adversarial knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 710–711.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Giannis Karamanolakis, Jun Ma, and Xin Luna Dong. 2020. Txtract: Taxonomy-aware knowledge extraction for thousands of product categories. *arXiv preprint arXiv:2004.13852*.

Zixuan Ke and Bing Liu. 2022. Continual learning of natural language processing tasks: A survey. *arXiv preprint arXiv:2211.12701*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Yuang Liu, Wei Zhang, Jun Wang, and Jianyong Wang. 2021. Data-free knowledge transfer: A survey. *arXiv preprint arXiv:2112.15278*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. 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 565–576.

Alireza Makhzani and Brendan Frey. 2013. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197.

Aliakbar Panahi, Seyran Saeedi, and Tom Arodz. 2021. Shapeshifter: a parameter-efficient transformer using factorized reshaped matrices. *Advances in Neural Information Processing Systems*, 34:1337–1350.

Keiji Shinzato, Naoki Yoshinaga, Yandi Xia, and Wei-Te Chen. 2022. Simple and effective knowledge-driven query expansion for QA-based product attribute extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 227–234, Dublin, Ireland. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020. Learning to extract attribute value from product via question answering: A multi-task approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 47–55.

Huimin Xu, Wenting Wang, Xinnian Mao, Xinyu Jiang, and Man Lan. 2019. Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5214–5223.

Jun Yan, Nasser Zalmout, Yan Liang, Christan Grant, Xiang Ren, and Xin Luna Dong. 2021. Adatag: Multi-attribute value extraction from product profiles with adaptive decoding. *arXiv preprint arXiv:2106.02318*.

Li Yang, Qifan Wang, Zac Yu, Anand Kulkarni, Sumit Sanghai, Bin Shu, Jon Elsas, and Bhargav Kanagal. 2022. Mave: A product dataset for multi-source attribute value extraction. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1256–1265.

Junjie Ye, Xuanting Chen, Nuo Xu, Can Zu, Zekai Shao, Shichun Liu, Yuhan Cui, Zeyang Zhou, Chao Gong, Yang Shen, et al. 2023. A comprehensive capability analysis of gpt-3 and gpt-3.5 series models. *arXiv preprint arXiv:2303.10420*.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Opentag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1049–1058.

A Datasets

We evaluate our model on the public Multi-source Attribute Value Extraction (MAVE) dataset (Yang et al., 2022), which is created from raw values from Amazon Review pages (Ni et al., 2019). MAVE contains over 2.2 million products distributed into 1257 different product categories, with 3 million attribute-value annotations. We further filter domains with a frequency above 100, resulting in a dataset of 997 domains encompassing 448 PTs and 365 Attrs.

B Implementation Details

We split the dataset by separating samples under each domain into train:eval:test = 8:1:1,(The ratio is referred from MAVE (Yang et al., 2022)) We conduct experiments with two-stage trainers: closed-world training on only seen domains and open-world training on either only unseen domains or unseen+seen domains. For Table 1, we split the data into 500/400 seen/unseen domains. We make a split on 500 seen domain data with a ratio 9:1. 90% samples are used for closed-world training while 10% samples are used in open-world training as the new value cases of the domain type, $(PT_{old}, Attr_{old})$. 400 unseen domains comprise the training samples of the other three new domain types. The evaluation is performaned on all test samples of each domain.

Slightly different in Table 2, we only have 400 seen domains to ensure sufficient unseen domains participating in the three incremental trainings (from 100 to 300 unseen domains) at the open-world stage. And we also set apart 10% of seen domains for $N_R = 10\%$. Another difference is the testing set, containing only the seen domain testing samples to evaluate forgetting scores.

We experimentally use the ALBERT$_{xlarge}$ as the backbone. We optimize using SparseAdam (Makhzani and Frey, 2013) to enable a single sub-matrix update whose learning rate is increased linearly over the first 10% of the steps and then decayed linearly to zero. All runs are conducted on 4 V100s with a batch size of 32. For each dataset and algorithm, we set the hyperparameters $(\alpha = 0.6, \beta_1 = 0.4, \beta_2 = 0.3)$ according to the best model according to Macro-F1 on the validation set. The sub-matrix hidden size is $r = 32$. Because, after evaluating a list value of r, 32 is the one that balances between efficient training and SOTA performance achievements. All the results are the average score after 5 runs.

C Baselines

We compare our model with two groups of baselines. If the model trains multiple times each time targeting a single domain, we group these methods into the One-for-Each group. Otherwise, if trained on all domains at once, the models fall into the All-in-One group.

The One-for-Each training strategy is applied to the following: (1) BERT$_{Ada}$, an implementation of the work in Houlsby et al. (2019). (2) ALBERT$_{Ada}$, similar implementation of BERT$_{Ada}$ with the modification that we wedge the adapter into the single encoder for replication in ALBERT (Lan et al., 2019).

The All-in-One group includes: (1) ALBERT: a compressed BERT (Devlin et al., 2018) taking in the concatenation of domain names and the product text profile. (2) BERT: vanilla baselines varying in model size, fed with the concatenation of domain and profile. Empirically, we concatenate the last two layers as the final features for classification. (3) ALBERT$_{ft^\sharp}$: we apply the weight factorization on all key components (Query, Key, Value, Dense, Feed-forward layers) forming the transformer encoder. During the life-long training, we only employ $\mathcal{L}_t$ and $\mathcal{L}_{fea}$ and remove $\mathcal{L}_{norm}$ considering the model training expense. (4) BERT$_{ft}$: we conduct the factorization on the linear layer above BERT and before the label classification layer. The input to this linear layer is the concatenation of the last two feature layers in BERT. This ensures the same backbone setting with the baseline BERT and provides a comparable size feature representa-

	Linear	Transformer Encoder	Factorized Adapter
Setting: Hidden State Size, SVD Sub-matrix Size	768, 50	768, 50	768, 50
Components for applying SVD	Linear Layer (in=768, out=768)	Query/Key/Value/Dense/FFN	Linear Layer (in=768, out=768)
# trainable para. for three cases: All-in-One, One-for-Each, Domain-aware	All-in-one: 768*768 = **589K** One-for-Each: 10 domains: 768*768*10 =**5.89M** 100 domains: 768*768*100 =**58.9M** Domain-aware: 768*50*10 +10*50*768= **0.77M**	- Query/Key/Value/Dense All-in-one:768*768 = **0.59M** One-for-Each: 10 domains: 768*768*10 =**5.9M** 100 domains: 768*768*100 =**59M** Domain-aware: 768*50*10 +10*50*768 = **0.77M** - Feed forward Network All-in-one:768*4*768 =**2.36M** One-for-Each: 10 domains:768*4*768*10 = **23.6M** 100 domains: 768*4*768*100 = **236M** Domain-aware: 768*4*50*10+10*50*768 = **1.92M** **- Total: Q+K+V+D + 2*FFN** All-in-one:0.59*4+2.36*2=**7.08M** One-for-Each: 10 domains:7.08*10 = **70.8M** 100 domains: 7.08*100 = **708M** Domain-aware: 0.77*4+1.93*2 = **6.94M**	All-in-one: None (Adapter should be trained for each task) One-for-Each: 10 domains: (768*50+2*50+50*768+2*768)*10 =**0.78M** 100 domains: 0.078*100 =**7.8M** Domain-aware: (768*50+2*50)*10 +10*(50*768+2*768) = **0.78M**

Figure 6: An example to show parameter saving under three model training strategies: All-in-One, One-for-Each, Domain-aware.

tion with DALLA, which utilizes ALBERT$_{xlarge}$ with a hidden size of 2048. (5) SUOpenTag (Xu et al., 2019): enable the model to be aware of the attention of attributes to the profile. The model shows its scalability on the large-volume dataset with well-learned attention between attributes and values' context. (6) MAVE (Yang et al., 2022): a multi-source attribute value extraction model via question answering, modeling the products with structure and long profiles. Compared with its preliminary work AVEQA (Wang et al., 2020), it addresses mutual attention of the local separate parts (product type, attribute, title, profile) in the long text. However, MAVE dataset is constructed by AVEQA and annotation rules. That is why the evaluations of AVEQA and MAVEQA on MAVE are in high scores approaching 100% as reported, and we only adopt it as a baseline when evaluating catastrophic forgetting rather than performance on the extraction task.

Our model belongs to the All-in-One group. The comparison with these two groups of baselines is to verify the effectiveness of model de-contamination when performances are approaching the One-for-Each model, and model life-long learning ability with high parameter efficiency when superiority is exhibited among All-in-One models.

D Parameter Saving

We describe the parameter-saving computation details. Each adapter layer has two sets of projection matrices. One is $\mathcal{U}_{PT} \in \mathbb{R}^{N \times H_i \times r}$, the other is $\mathcal{V}_{Attr} \in \mathbb{R}^{M \times r \times H_o}$. N and M are the number of PTs and Attrs included for training. $2r + 2H_o$ parameters are used for factorized norm layer $\mathcal{N}_{PT}$ and $\mathcal{N}_{Attr}$. The total number of parameters for factorized Adapters across training domains is $N \cdot (H_i \cdot r + 2r) + (r \cdot H_o + 2H_o) \cdot M$, which scales linearly with the hidden layer size r times the number of PTs or Attrs. However during each run for a specific domain, the trainable parameter of the factorized adapter is $1 \cdot (H_i \cdot r + 2r) + (r \cdot H_o + 2H_o) \cdot 1$. The rest are unavailable for gradient modification, which contributes to the training efficiency of DALLA.

We provide an example of how to save model size through factorization in Figure 6. We select 10 PTs and 10 Attrs, which can be combined to form a minimum of 10 and a maximum of 100 domains. We conduct SVD on different model components, including the linear layer, transformer encoder, and adapter. All of these components are integrated into a backbone model based on cased BERT, which consists of 12 layers and a hidden size of 768. After integrating these components, we conduct model

training using three different strategies: All-in-One, One-for-Each, and Domain-aware. If we apply SVD on a linear layer or a transformer encoder, the domain-aware model, which implements the parameter factorization, results in significant parameter reduction. This reduction in parameters is considerably less than that observed in the One-for-Each model and approaches the parameter count of the All-in-One model. Excessive factorization can have a detrimental impact on performance, as evidenced by the degradation of parameter savings when applying SVD to all key components in an encoder, compared to applying it solely to a linear layer. Table 1 reveals that ALBERT$_{ft}$ exhibits the poorest performance, whereas BERT$_{ft}$ ranks as the second-best performer. Factorized Adapter demonstrates a nearly equivalent parameter size to that of the linear layer after implementing SVD. However, it excels in overcoming the challenge of catastrophic forgetting, as evidenced by the results presented in Table 2.