Incremental Prompting: Episodic Memory Prompt for Lifelong Event Detection

Minqian Liu[♣], **Shiyu Chang**[♠], **Lifu Huang**[♣]
[♣]Virginia Tech, [♠]University of California Santa Barbara
[♣]{minqianliu,lifuh}@vt.edu, [♠]chang87@ucsb.edu

Abstract

Lifelong event detection aims to incrementally update a model with new event types and data while retaining the capability on previously learned old types. One critical challenge is that the model would catastrophically forget old types when continually trained on new data. In this paper, we introduce **E**pisodic **M**emory **P**rompts (**EMP**) to explicitly retain the learned task-specific knowledge. Our method adopts continuous prompt for each task and they are optimized to instruct the model prediction and learn event-specific representation. The EMPs learned in previous tasks are carried along with the model in subsequent tasks, and can serve as a memory module that keeps the old knowledge and transferring to new tasks. Experiment results demonstrate the effectiveness of our method. Furthermore, we also conduct a comprehensive analysis of the new and old event types in lifelong learning.[1]

1 Introduction

Class-incremental event detection (Cao et al., 2020; Yu et al., 2021) is a challenging setting in lifelong learning, where the model is incrementally updated on a continual stream of data for new event types while retaining the event detection capability for all the previously learned types. The main challenge of class-incremental event detection lies in the *catastrophic forgetting* problem, where the model's performance on previously learned types significantly drops after it is trained on new data. Recent studies (Lopez-Paz and Ranzato, 2017; Wang et al., 2019) have revealed that replaying stored samples of old classes can effectively alleviate the catastrophic forgetting issue. However, simply fine-tuning the entire model on the limited stored samples may result in overfitting, especially when the model has a huge set of parameters. How to effectively leverage the limited stored examples still remains an important question.

Prompt learning, which is to simply tune a template-based or continuous prompt appended to the input text while keeping all the other parameters freezed, has recently shown comparable or even better performance than fine-tuning the entire model in many NLP tasks (Brown et al., 2020; Li and Liang, 2021; Lester et al., 2021). It is especially flavored by lifelong learning since it only tunes a small amount of parameters. However, it is still non-trivial to equip prompts with the capability of retaining acquired knowledge and transferring to new tasks in the class-incremental setting.

In this work, we propose an incremental prompting framework that introduces **E**psodic **M**emory **P**rompts (**EMP**) to store and transfer the learned type-specific knowledge. At each training stage, we adopt a learnable prompt for each new event type added from the current task. The prompts are initialized with event type names and fine-tuned with the annotations from each task. To encourage the prompts to always carry and reflect type-specific information, we entangle the feature representation of each event mention with the type-specific prompts by optimizing its type distribution over them. After each training stage, we keep the learned prompts in the model and incorporate new prompts for next task. In this way, the acquired task-specific knowledge can be carried into subsequent tasks. Therefore, our EMP can be considered as a soft episodic memory that preserves the old knowledge and transfers it to new tasks. Our method does not require task identifiers at test time, which enables it to handle the challenging class-incremental setting. Our contributions can be summarized as follows:

• We propose **E**psodic **M**emory **P**rompts (**EMP**) which can explicitly carry previously learned knowledge to subsequent tasks for class-incremental event detection. Extensive experi-

[1]The source code is publicly available at https://github.com/VT-NLP/Incremental_Prompting.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2157–2165
October 12–17, 2022.

ments validate the effectiveness of our method.

• To the best of our knowledge, we are the first to adopt prompting methods for class-incremental event detection. Our framework has the potential to be applied to other incremental learning tasks.

2 Problem Formulation

Given an input text $x_{1:L}$ and a set of target spans $\{(x_i, x_j)\}$ from it, an event detection model needs to assign each target span with an event type in the ontology or label it as *Other* if the span is not an event trigger. For class-incremental event detection, we aim to train a single model f_θ on a sequence of T tasks $\{\mathcal{D}_1, ..., \mathcal{D}_T\}$ that consist of non-overlapping event type sets $\{\mathcal{C}_1, ..., \mathcal{C}_T\}$[2]. In each t-th task, the model needs to classify each mention to any of the types that have seen so far $\mathcal{O}_t = \mathcal{C}_1 \bigcup ... \bigcup \mathcal{C}_t$. The training instances in each task $\mathcal{D}_t$ consist of tuples of an input text $x^t_{1:L}$, a target span $\bar{x}^t$, and its corresponding label $\mathbf{y}^t$ where $\mathbf{y}^t \in \mathcal{C}_t$. For convenience, the notations are for the t-th training stage by default unless denoted explicitly in the following parts of the paper.

3 Approach

3.1 Span-based Event Detection

Given an input sentence $x^t_{1:L}$ from task $\mathcal{D}_t$, we first encode it with BERT (Devlin et al., 2019) to obtain the contextual representations $\mathbf{x}^t_{1:L} = \text{BERT}(x^t_{1:L})$. Note that we freeze BERT's parameters in our method and all baselines. For each span $\bar{x}^t$, we concatenate its starting and ending token representations and feed them into a multilayer perceptron (MLP) to get the span representation $\mathbf{h}^t_{span}$. Then, we apply a linear layer on $\mathbf{h}^t_{span}$ to predict the type distribution of the span $p^t = linear(\mathbf{h}^t_{span})$. We use cross-entropy loss to train the model on $\mathcal{D}_t$:

$$\mathcal{L}_C = -\sum_{(\bar{x}^t, y^t) \in \mathcal{D}_t} \log p^t. \quad (1)$$

3.2 Episodic Memory Prompting

To overcome the catastrophic forgetting and exemplar memory overfitting issues, we design an incremental prompting approach with Episodic Memory Prompts (EMPs) to preserve the knowledge learned from each task and transfer to new tasks.

Given an incoming task $\mathcal{D}_t$ and its corresponding new event type set $\mathcal{C}_t = \{c^t_1, ..., c^t_{n_t}\}$, we first initialize a sequence of new *prompts* $\mathbf{C}^t = [\mathbf{c}^t_1, ..., \mathbf{c}^t_{n_t}]$ where $\mathbf{c}^t_i \in \mathbb{R}^{1\times e}$ is a type-specific prompt for type c^t_i, n_t is the number of event types in the t-th task. e is the embedding dimension size. In our experiments, we use the event type name to initialize each event type prompt $\mathbf{c}^t_i$ (see Appendix A for details). Note that we always preserve the prompts learned from previous tasks, thus the accumulated prompts until the t-th task are represented as $\mathbf{I}^t = [\mathbf{C}^1, ..., \mathbf{C}^t]$. Given a particular sentence $x^t_{1:L}$ from $\mathcal{D}_t$, we concatenate it with the accumulated prompts $\mathbf{I}^t$, encode the whole sequence with BERT, and obtain the sequence of contextual representations $[\tilde{\mathbf{x}}^t_{1:L}; \tilde{\mathbf{I}}^t]$, where $\tilde{\mathbf{x}}^t_{1:L}$ and $\tilde{\mathbf{I}}^t$ denote the sequence of contextual embeddings of $x^t_{1:L}$ and $\mathbf{I}^t$ respectively. $[;]$ is concatenation operation. Then, similar as Section 3.1, we obtain a representation $\tilde{\mathbf{h}}^t_{span}$ for each span based on $\tilde{\mathbf{x}}^t_i$, and predict the logits over all target event types $\tilde{p}^t = linear(\tilde{\mathbf{h}}^t_{span})$.

We expect the EMPs to be specific to the corresponding event types and preserve the knowledge of each event type from previous tasks. So we design an entangled prompt optimization strategy to entangle the feature representation of each span with the event type-specific prompts by computing an event type probability distribution over them. Specifically, given a span representation $\tilde{\mathbf{h}}^t_{span}$ and EMP representations $\tilde{\mathbf{I}}^t$, we compute the probability distribution over all prompts as $\tilde{p}^t_c = \text{MLP}(\tilde{\mathbf{I}}^t) \cdot \tilde{\mathbf{h}}^t_{span}$, where $\cdot$ is the dot product. Finally, we combine the original logits $\tilde{p}^t$ and $\tilde{p}^t_c$ to predict the event type label for each span:

$$\tilde{\mathcal{L}}_C = -\sum_{(\bar{x}^t, y^t) \in \mathcal{D}_t} \log (\tilde{p}^t + \tilde{p}^t_c). \quad (2)$$

At the end of each training stage, we keep the learned prompts from the current task $\mathbf{C}^t$ in the model, and then initialize a new prompt $\mathbf{C}^{t+1}$ for the next task and concatenate it with the previous accumulated prompts $\mathbf{I}^t$ incrementally: $\mathbf{I}^{t+1} = [\mathbf{I}^t; \mathbf{C}^{t+1}]$.

3.3 Lifelong Learning with Experience Replay and Knowledge Distillation

To alleviate the catastrophic forgetting issue, two strategies have been widely applied in many lifelong learning works (Rebuffi et al., 2017; Sun et al., 2020; Cao et al., 2020; Yu et al., 2021): (1) Experience Replay which is to repeatedly optimize the model on the stored previous data in subsequent tasks; and (2) Knowledge Distillation (KD)

[2]Though the type sets from all tasks contain *Other*, they have distinct meanings given different seen types.

that is to ensure the output probabilities and/or features from the current and previous models to be matched, respectively. We also adopt these two baselines to validate the compatibility of our method with other lifelong learning techniques.

Specifically, after training on $\mathcal{D}_t$, we apply the herding algorithm (Welling, 2009) to select 20 training samples for each type into the memory buffer, denoted as $\mathcal{M}$. Similar as Equation 2, the objective for experience replay is:

$$\mathcal{L}_{ER} = -\sum_{(\bar{x}^r, y^r) \in \mathcal{M}} \log\,(\tilde{p}^t + \tilde{p}^t_c). \quad (3)$$

For knowledge distillation, following (Cao et al., 2020), we apply both *prediction-level* and *feature-level* distillation. The objectives for prediction-level KD and feature-level KD are computed as:

$$\mathcal{L}_{PD} = -\sum_{(\bar{x}^r, y^r) \in \mathcal{M}} (\tilde{p}^{t-1} + \tilde{p}^{t-1}_c) \log\,((\tilde{p}^t + \tilde{p}^t_c)),$$

$$\mathcal{L}_{FD} = \sum_{(x^r, (x^r_i, x^r_j), y^r) \in \mathcal{M}} 1 - g(\bar{\mathbf{h}}^{t-1}_{span}, \bar{\mathbf{h}}^{t}_{span}),$$

where g is the cosine similarity function. $\bar{\mathbf{h}}^{t-1}_{span}$ and $\bar{\mathbf{h}}^{t}_{span}$ are l_2-normalized features from the model at $t-1$ and t stages, respectively.

Optimization We combine the multiple objectives with weighting factors α and β as follows:

$$\mathcal{L} = \tilde{\mathcal{L}}_C + \alpha \mathcal{L}_{ER} + \beta(\mathcal{L}_{PD} + \mathcal{L}_{FD}).$$

4 Experiments and Discussions

Experiment Settings We conduct experiments on two benchmark datasets: ACE05-EN (Doddington et al., 2004) and MAVEN (Wang et al., 2020), and construct the class-incremental datasets following the *oracle negative* setting in (Yu et al., 2021). We divided the ontology into 5 subsets with distinct event types, and then use them to constitute a sequence of 5 tasks denoted as $\mathcal{D}_{1:5}$. We use the same partition and task order permutations in (Yu et al., 2021). During the learning process from $\mathcal{D}_1$ to $\mathcal{D}_5$, we constantly test the model on the entire test set (which contains the whole ontology) and take the mentions of unseen event types as negative instances. More implementation details, including parameters, initialization of prompts as well as baselines are shown in Appendix A.

Baselines We consider the following baselines for comparison: (1) **BERT-ED**: simply trains the BERT based event detection model on new tasks without prompts, experience replay or knowledge distillation. It's the same as the span-based event detection baseline in Section 3.1. (2) **KCN** (Cao et al., 2020): use a prototype-based example sampling strategy and hierarchical distillation. As the original approach studied a different setting, we adapt their prediction-level and feature-level distillation as the baseline. (3) **KT** (Yu et al., 2021): transfer knowledge between old types and new types in two directions. (4) **iCaRL*** (Rebuffi et al., 2017): use nearest-mean-of-exemplars rules to perform classification combined with knowledge distillation. iCaRL adopts different strategies for classification, experience replay, and distillation. We thus directly report the result in (Yu et al., 2021) for reference. (5) **EEIL** (Castro et al., 2018): use an additional finetuning stage on the balanced dataset. (6) **BIC** (Wu et al., 2019): use a bias correction layer after the classification layer. (7) **Upperbound**: trains the same model on all types in the datasets jointly. For **iCaRL**, **EEIL**, and **BIC**, we use the same implementation in (Yu et al., 2021). For fair comparison, our approach and all baselines (except for the Upperbound baseline) are built upon **KCN** and use the same experience replay and knowledge distillation strategies described in Section 3.2. We set the exemplar buffer size as 20, and allow one exemplar instance to be used in each training batch instead of the whole memory set. Note that this replay setting is different from the one in (Yu et al., 2021), where we allow much less frequent exemplar replay, and thus our setting is more efficient, challenging, and realistic.

Results We present the main results in Table 1. We have following observations: (1) by comparing the performance of various approaches on Task 1 which are not affected by any catastrophic forgetting, our approach improves 4.1% F-score on MAVEN and 1.3% F-score on ACE05, demonstrating that by incorporating task-specific prompts, event detection itself can be significantly improved. EMPs even provide more improvement on MAVEN which contains a lot more event types than ACE05, suggesting the potential of incorporating EMPs for fine-grained event detection; (2) **KCN** can be viewed as an ablated version of our approach without EMPs. Our approach consistently outperforms **KCN** on almost all tasks on both datasets, demon-

	MAVEN					ACE05-EN				
Task	1	2	3	4	5	1	2	3	4	5
BERT-ED	63.51	39.99	33.36	23.83	22.69	58.30	43.96	38.02	21.53	25.71
iCaRL* (Rebuffi et al., 2017)	18.08	27.03	30.78	31.26	29.77	4.05	5.41	7.25	6.94	8.94
EEIL (Castro et al., 2018)	63.51	50.62	45.16	41.39	38.34	58.30	54.93	52.72	45.18	41.95
BIC (Wu et al., 2019)	63.51	46.69	39.15	31.69	30.47	58.30	45.73	43.28	35.70	30.80
KCN (Cao et al., 2020)	63.51	51.17	46.80	38.72	38.58	58.30	54.71	52.88	44.93	41.10
KT (Yu et al., 2021)	63.51	52.36	47.24	39.51	39.34	58.30	**55.41**	53.95	45.00	42.62
EMP (Ours)	**67.86**	**60.26**	**58.61**	**54.81**	**50.12**	**59.60**	53.19	**55.20**	**45.64**	**43.28**
Upperbound (Ours)	/	/	/	/	68.42	/	/	/	/	67.22

Table 1: Comparison between our approach and baselines in terms of micro F-1 (%) on 5 class-incremental tasks. We report the *averaged* results on 5 permutations of tasks so that the results are independent of randomness.

strating the effectiveness of EMPs on improving class-incremental event detection; (3) Comparing with **BERT-ED**, **KCN** adopts experience replay and knowledge distillation. Their performance gap verifies that these two strategies can dramatically alleviate catastrophic forgetting; (4) There is still a large gap between the current approaches and the upperbound, indicating that catastrophic forgetting still remains a very challenging problem. Note that the only difference in **EEIL**, **BIC**, **KCN**, and **KT** is the lifelong learning techniques they applied, thus these models have identical F-score on Task 1. We also analyze failed examples in Appendix B.

Analysis of New and Old Types in Lifelong Learning Figure 1 shows the F-score on old and new event types in each training stage for our approach and **KT** (Yu et al., 2021) on MAVEN. Our approach consistently outperforms **KT** by a large margin on both old types and new types, demonstrating that our EMPs effectively preserve learned knowledge from old event types and improve event detection when annotations are sufficient. Interestingly, comparing the F-score on new types in Task 1 and old types in Task 2, both methods improve the performance on the types of Task 1, indicating that both methods have the potential of leveraging indirect supervision to improve event detection.

Ablation Study We consider four ablated models based on our EMPs: (1) change the prompt initialization[3] from using event type name representations to using random distribution; (2) remove the entangled prompt optimization but still append the event type prompts to the end of each input sentence and apply Equation 1 only to detect the events; (3) remove the knowledge distillation loss $\mathcal{L}_{PD}$ and $\mathcal{L}_{FD}$; (4) use completely fixed prompts

[3]Appendix A shows the details of prompt initialization. We use the same initialization for the discrete prompt ablation.

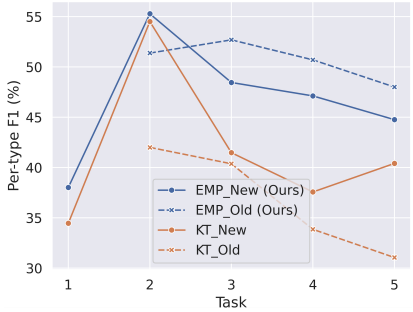

Figure 1: Per-type F1 on old types and new types in each lifelong task on one *randomly selected* permutation of the MAVEN dataset. The F-scores on old and new types reflect the ability to retain acquired knowledge and to learn new types, respectively. Best viewed in color.

to replace the trainable soft prompts. From Table 2, we observes that: (1) using event type names to initialize the prompts is helpful in most tasks; (2) both entangled prompt optimization and knowledge distillation can help alleviate catastrophic forgetting; (3) switching the continuous prompts to discrete prompts degrades the performance significantly, suggesting that the continuous prompts are generally more promising than discrete prompts.

Task	1	2	3	4	5
EMP (Ours)	**67.86**	**60.26**	**58.61**	**54.81**	**50.12**
- w/o EInit	66.73	58.99	57.63	53.98	49.33
- w/o EPO	67.04	59.02	57.79	53.72	49.05
- w/o KD	67.86	57.57	55.83	53.02	48.65
- Discrete	60.13	51.98	50.60	48.97	43.68

Table 2: Ablation study on event-specific prompt initialization (EInit), entangled prompt optimization (EPO), knowledge distillation (KD), and trainable soft prompts (Discrete) on MAVEN. We report the *averaged* results on 5 permutations of tasks.

Effect of Exemplar Buffer Size We conduct an analysis on the effect of exemplar buffer size. We explore the buffer size for each type in {0, 10, 20}. We use **KT** as the baseline when buffer size is 20

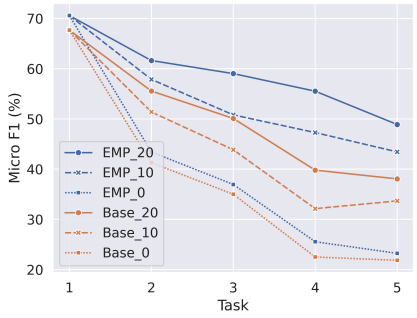

Figure 2: Performance with different buffer size in each task on one *randomly selected* permutation of MAVEN. Best viewed in color.

and 10. Note that when buffer size is 0, we do not adopt either experience replay or knowledge distillation and thus use **BERT-ED** as the baseline. We plot the results on Figure 2. We observed that: (1) Decreasing the buffer size for each type from 20 to 10 degrades the performance of both models. This indicates that reducing data diversity may result in the overfitting on example data, and thus deteriorates the performance; (2) Our method still outperforms the **KT** baseline when storing only half of history examples, which indicates our method is able to utilize the stored examples more effectively. (3) When the buffer size decreases to 0, the performance of both methods drops significantly. This shows that both approaches highly rely on the stored data to overcome the catastrophic forgetting problem. This calls for developing more advance techniques to reduce the dependence on stored examples, as storing past data could result in data leakage in real-world applications.

5 Related Work

Lifelong Event Detection Deep neural networks have shown state-of-the-art performance on supervised event detection (Nguyen et al., 2016; Feng et al., 2016; Zhang et al., 2017; Huang and Ji, 2020; Wang et al., 2021b). However, when moving to lifelong learning setting, their performance significantly drops (Kirkpatrick et al., 2017; Aljundi et al., 2019; Biesialska et al., 2020; Cui et al., 2021; Ke et al., 2021b; Madotto et al., 2021; Ke et al., 2021a; Feng et al., 2022). Though experience replay (Lopez-Paz and Ranzato, 2017; de Masson d'Autume et al., 2019; Guo et al., 2020; Han et al., 2020; Zhao et al., 2022) and knowledge distillation (Chuang et al., 2020; Cao et al., 2020) have shown to be effective in overcoming catastrophic forgetting, they highly rely on the stored data from old tasks, which is not the most realistic setting for lifelong learning.

Prompt Learning Conditioning on large-scale pre-trained language models, prompt learning (Brown et al., 2020; Lester et al., 2021; Liu et al., 2021; Wang et al., 2021a,c, 2022) has shown comparable performance as language model fine-tuning. Specific to lifelong learning, Qin and Joty (2021) use prompt tuning to train the model as a task solver and data generator for lifelong few-shot problem. Zhu et al. (2022) propose continual prompt tuning for dialogue state tracking. To the best of our knowledge, we are the first work to adopt prompt learning for class-incremental event detection.

6 Conclusion

We propose a novel Episodic Memory Prompting (EMP) framework for class-incremental event detection. During each training stage, EMP learns type-specific knowledge via a continuous prompt for each event type. The EMPs trained in previous tasks are kept in the model, such that the acquired task-specific knowledge can be transferred into the following new tasks. Experimental results validate the effectiveness of our method comparing with competitive baselines. Our extensive analysis shows that by employing EMPs, both event detection itself and the incremental learning capability of our approach are significantly improved.

Acknowledgements

We thank the anonymous reviewers and area chair for their valuable time and constructive comments, and the helpful discussions with Zhiyang Xu. We also thank the support from the Amazon Research Awards.

References

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. 2019. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11816–11825.

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

Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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

Francisco M. Castro, Manuel J. Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. 2018. End-to-end incremental learning. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, volume 11216, pages 241–257. Springer.

Yung-Sung Chuang, Shang-Yu Su, and Yun-Nung Chen. 2020. Lifelong language knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2914–2924, Online. Association for Computational Linguistics.

Li Cui, Deqing Yang, Jiaxin Yu, Chengwei Hu, Jiayang Cheng, Jingjie Yi, and Yanghua Xiao. 2021. Refining sample embeddings with relation prototypes to enhance continual relation extraction. 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 232–243, Online. Association for Computational Linguistics.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems*, pages 13122–13131.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Shaoxiong Feng, Xuancheng Ren, Kan Li, and Xu Sun. 2022. Hierarchical inductive transfer for continual dialogue learning. *CoRR*, abs/2203.10484.

Xiaocheng Feng, Lifu Huang, Duyu Tang, Heng Ji, Bing Qin, and Ting Liu. 2016. A language-independent neural network for event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 66–71, Berlin, Germany. Association for Computational Linguistics.

Yunhui Guo, Mingrui Liu, Tianbao Yang, and Tajana Rosing. 2020. Improved schemes for episodic memory-based lifelong learning. In *Advances in Neural Information Processing Systems*.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6440, Online. Association for Computational Linguistics.

Lifu Huang and Heng Ji. 2020. Semi-supervised new event type induction and event detection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 718–724.

Zixuan Ke, Bing Liu, Hu Xu, and Lei Shu. 2021a. CLASSIC: Continual and contrastive learning of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6871–6883, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Zixuan Ke, Hu Xu, and Bing Liu. 2021b. Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755, Online. Association for Computational Linguistics.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*,

pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *CoRR*, abs/2110.07602.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7452–7467, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Chengwei Qin and Shafiq Joty. 2021. LFPT5: A unified framework for lifelong few-shot language learning based on prompt tuning of T5. *CoRR*, abs/2110.07298.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5533–5542. IEEE Computer Society.

Jingyuan Sun, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2020. Distill and replay for continual language learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3569–3579, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Chengyu Wang, Jianing Wang, Minghui Qiu, Jun Huang, and Ming Gao. 2021a. TransPrompt: Towards an automatic transferable prompting framework for few-shot text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2792–2802, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 796–806, Minneapolis, Minnesota. Association for Computational Linguistics.

Sijia Wang, Mo Yu, Shiyu Chang, Lichao Sun, and Lifu Huang. 2021b. Query and extract: Refining event extraction as type-oriented binary decoding. *arXiv preprint arXiv:2110.07476*.

Sijia Wang, Mo Yu, and Lifu Huang. 2022. The art of prompting: Event detection based on type specific prompts. *arXiv preprint arXiv:2204.07241*.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online. Association for Computational Linguistics.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. 2021c. Learning to prompt for continual learning. *CoRR*, abs/2112.08654.

Max Welling. 2009. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1121–1128, New York, NY, USA. Association for Computing Machinery.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. 2019. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 374–382. Computer Vision Foundation / IEEE.

Pengfei Yu, Heng Ji, and Prem Natarajan. 2021. Lifelong event detection with knowledge transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5278–5290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tongtao Zhang, Spencer Whitehead, Hanwang Zhang, Hongzhi Li, Joseph Ellis, Lifu Huang, Wei Liu, Heng

Ji, and Shih-Fu Chang. 2017. Improving event extraction via multimodal integration. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 270–278.

Kang Zhao, Hua Xu, Jiangong Yang, and Kai Gao. 2022. Consistent representation learning for continual relation extraction. *CoRR*, abs/2203.02721.

Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. 2022. Continual prompt tuning for dialog state tracking. *CoRR*, abs/2203.06654.

A Experimental Details

Implementation Details During training, we use AdamW (Loshchilov and Hutter, 2019) optimizer with the learning rate set to $1e-4$ and weight decay set to $1e-2$. Different from previous work (Yu et al., 2021), we set the batch size to 1 as we encode each sentence once and consider all target spans in the sentence at the same time. We adopt gradient accumulation with the step set to 8. As the number of batches is large, we apply a periodic replay and distillation strategy with the interval set to 10 to reduce computational cost. For each lifelong task $\mathcal{D}_t$, we set the maximum number of training epochs to 20. We adopt the early stopping strategy with patience 5, i.e., the training stops if the performance on the development set does not increase for 5 epochs. The temperature parameter used in prediction-level distillation is set to 2. The weighted factors for the loss function α and β are computed based on the number of learned event types and new types.

The parameters of each prompt in EMPs are initialized with the corresponding event type name. Specifically, there are three cases in the initialization: (1) If the type name is *single-token* and it is contained in BERT's vocabulary, we directly use the pre-trained embedding of this token to initialize the prompt; (2) If the type name is *multiple-token* and the tokens are contained in BERT's vocabulary, we take the average of the pre-trained embeddings of these tokens to initialize the prompt; (3) If the type name contains *Out-of-Vocabulary (OOV)* tokens, we replace the OOV tokens with the synonyms that are contained in BERT's vocabulary. It is worth noting that we randomly initialize the prompt for the *Other* type and keep updating it throughout all lifelong tasks. We leave how to incorporate more effective prior knowledge into prompts for future work.

B Failure Cases

We show some of typical failure cases in Table 3. We have following observations: (1) the first three examples illustrate the catastrophic forgetting problem in class-incremental event detection. While the model predicted correct event types right after it was trained on those types, it starts to predict wrong types in subsequent tasks. Interestingly, we observed that the model typically predicts the *Other* type or the types relevant to triggers (e.g., *Creating*) when forgetting occurs; (2) the 4th and 5th examples showed that the model sometimes keeps predicting the old types while it is supposed to predict new types in subsequent tasks. (3) the 7th example showed that the model can sometimes correct itself in subsequent tasks, which indicates the experience replay and knowledge distillation have the potentials of improving old types; (4) the last example indicates that in some cases, the model is interfered after trained on a task contained ambiguous types even though it predicts the correct type in all other tasks.

Text	Gold Event Type(s)	Predicted Event Type(s)
The Minnesota Territory itself was **formed** only in 1849 but the area had a rich history well before this.	Coming_to_be ($\mathcal{D}_2$)	$\underline{f_2}$: Coming_to_be; $f_{3:4}$: Other; f_5: Creating ($\mathcal{D}_5$)
He **informed** the Air France chief executive in writing "I did not believe the captain capable of qualifying in the 707."	Telling ($\mathcal{D}_3$)	$\underline{f_3}$: Telling; f_4: Other; f_5: Request ($\mathcal{D}_5$)
Unprepared for the attack, the Swedish attempted to **save**$_{[1]}$ their ships by cutting their anchor ropes and to **flee**$_{[2]}$.	[1] Rescuing ($\mathcal{D}_2$) [2] Escaping ($\mathcal{D}_3$)	[1] $\underline{f_{2:4}}$: Rescuing; f_5: Other [2] $\underline{f_{3:4}}$: Escaping; f_5: Other
After the uprising in Germany was **suppressed**, it flared briefly in several Swiss Cantons.	Control ($\mathcal{D}_3$)	$f_{3:5}$: Hindering ($\mathcal{D}_2$)
Brazilians and Chinese living in the region have been **evacuated**.	Escaping ($\mathcal{D}_3$)	$f_{3:5}$: Removing ($\mathcal{D}_1$)
A surveillance video of the incident was **released** by police four days after the shooting, on 26 November.	Releasing ($\mathcal{D}_3$)	$f_{3:4}$: Other; f_5: Publishing ($\mathcal{D}_5$)
Giral **agreed** to arm the trade unionists in defence of the Republic, and had 60, 000 rifles delivered to the CNT and UGT headquarters, although only 5, 000 were in working order.	Agree_or_refuse_to_act ($\mathcal{D}_4$)	f_4: Other; $\underline{f_5}$: Agree_or_refuse_to_act
Meanwhile, in the city, the Republican government had **reformed** under the leadership of socialist leader Francisco Largo Caballero.	Reforming_a_system ($\mathcal{D}_1$)	$\underline{f_{1,2,4,5}}$: Reforming_a_system; $\underline{f_3}$: Change_of_leadership ($\mathcal{D}_3$)

Table 3: Failure analysis of our EMP on the first permutation of MAVEN. The targeted triggers are highlighted in **bold**. $\mathcal{D}_i$ after the event types indicate the type is introduced at i-th task. f_i indicates the model trained after i-th task. We highlight the models predicted the correct types with underline. For example, "Coming_to_be ($\mathcal{D}_2$)" indicates the *Coming_to_be* type is introduced at the 2nd task. "$f_{3:4}$: Other" indicates the models trained after the 3rd and 4th task both predict the *Other* type.