Lifelong Event Detection via Optimal Transport

Anonymous ACL submission

Abstract

Continual Event Detection (CED) poses a formidable challenge due to the catastrophic forgetting phenomenon, where learning new tasks (with new coming event types) hampers performance on previous ones. In this paper, we introduce a novel approach, Lifelong Event Detection via Optimal Transport (**LEDOT**), that leverages optimal transport principles to align the optimization of our classification module with the intrinsic nature of each class, as defined by their pre-trained language modeling. Our method integrates replay sets, prototype latent representations, and an innovative Optimal Transport component. Extensive experiments on MAVEN and ACE datasets demonstrate LEDOT's superior performance, consistently outperforming state-of-the-art baselines. The results underscore LEDOT as a pioneering solution in continual event detection, offering a more effective and nuanced approach to addressing catastrophic forgetting in evolving environments.

1 Introduction

Event Detection (ED) presents a pivotal challenge in the domain of Information Extraction, tasked with identifying event triggers and their associated types from natural language text. However, the conventional ED training paradigm, characterized by its static nature, falls short in capturing the dynamic nature of real-world data. As highlighted by Yu et al. (2021), the ontology of events in ED research has been exhibiting a constant shift since its introduction, prompting the exploration of Continual Event Detection (CED), where data arrives continuously as a sequence of non-overlapping tasks. Although large language models (LLMs) have recently emerged, showcasing the ability to tackle numerous problems using only prompts without the need for fine-tuning, they fall short in the domains of information extraction (IE) (Han et al., 2023; Gao et al., 2023) and continual learning (Shi et al., 2024). Continual event detection, in particular, remains a difficult task that is not effectively addressed by LLMs.

CED presents many issues, most notably the *catastrophic forgetting* (McCloskey and Cohen, 1989; Ratcliff, 1990) phenomenon, where the training signal from new task hampers performance on past tasks. To provide a solution for this issue, numerous methods have been proposed, which usually fall into one of the three eminent approaches: *Regularization-based* (Chaudhry et al., 2021; Saha et al., 2021); *Architecture-based* (Yoon et al., 2017; Sokar et al., 2021); and *Memory-based* (Belouadah and Popescu, 2019; Rolnick et al., 2019). Out of these three, Memory-based methods have demonstrated superiority, leveraging access to the *Replay buffer*, a memory of limited size containing a portion of data from previously learned tasks for the model to rehearse during the training of new tasks.

Despite the promise of Memory-based methods, challenges abound. First, the finite capacity of the Replay buffer results in the eviction of valuable information, leading to incomplete representations of past tasks and hence, inadequate generality. Furthermore, the process of sampling and replaying data might not be optimally curated, potentially hindering the model's ability to generalize across tasks effectively.

This setback arises because the conventional practice of discarding the original head of pretrained language models (PLMs) during fine-tuning on downstream tasks overlooks valuable linguistic information encoded within it. In training the classifier module, state-of-the-art approaches (Qin et al., 2024; Wang et al., 2023; Liu et al., 2022; Yu et al., 2021) often do so in isolation, devoid of any priors or foundations. Discarding the language modeling head in PLMs is highly wasteful. The language modeling head contains essential information about vocabulary distribution based on contextual representations. Losing this head sacrifices crucial linguistic nuances, making it harder to align the classifier module and ensure efficient fine-tuning. Aligning our classifier module to this information is an essential but also formidable challenge. This alignment is crucial for ensuring a more efficient fine-tuning process, as it provides a foundational standard of learning that mitigates unnecessary overplasticity and prevents catastrophic forgetting.

To address the limitations discussed, this paper introduces a method to enhance Memory-based CED by integrating Optimal Transport (OT) principles, which provide a robust framework for measuring the distance between probability distributions. By incorporating OT into the fine-tuning process, we aim to retain essential linguistic information from the PLM head, ensuring the model remains invariant to specific tasks. This integration involves defining an appropriate cost matrix, a key challenge that we address by proposing a novel construction tailored to our method. Our approach ensures effective alignment between the PLM head and the classifier's output, leveraging OT to enhance the model's performance and robustness across various tasks while preserving the PLM's inherent linguistic knowledge.

2 Background

2.1 Event Detection

Following previous works, we formalize Event Detection as a Span-based Classification task. Given an input instance $x = (w_{1:L}, s, e)$ consisting of a L-token context sequence $w_{1:L}$, a start index s, and an end index e, an ED model has to learn to assign the text span $w_{s:e}$ into a label y from a set of pre-defined event types $\mathcal{Y}$, or NA if $w_{s:e}$ does not trigger a known event.

Generally, we use a language model $\mathcal{M}$ to encode the context sequence $w_{1:L}$ into contextualized representation $w'_{1:L}$. Then, a classifier is utilized to classify the representation of the trigger span:

$$p(y|x) = \texttt{softmax}(\texttt{Linear}(\texttt{FNN}([w'_s, w'_e]))). \quad (1)$$

Here, FNN denotes a feed-forward neural network, $[\cdot, \cdot]$ is the concatenation operation, h is the hidden vector representing $w_{s:e}$, and $p(y|x)$ models the probability of predicting y from the input x.

The model is trained on a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ using cross-entropy loss:

$$\mathcal{L}_C(\mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \log p(y|x). \quad (2)$$

To mitigate the imbalance between the number of event triggers and the number of NA spans, we re-weight the loss with a hyperparameter η:

$$\mathcal{L}_C = \eta \mathcal{L}_C(\mathcal{D}_{\mathsf{NA}}) + (1-\eta)\mathcal{L}_C(\mathcal{D} \setminus \mathcal{D}_{\mathsf{NA}}) \quad (3)$$

where $\mathcal{D}_{\mathsf{NA}}$ is the set of NA instances.

2.2 Continual Event Detection

The training data in CED is not static but arrives sequentially as a stream of T non-overlapping tasks $\{\mathcal{D}_t | \bigcup_{t=1}^T \mathcal{D}_t = \mathcal{D}; \mathcal{D}_t \cap \mathcal{D}_{t'} = \emptyset\}$. At each timestep t, the t^{th} task data only covers a set of event types $\mathcal{Y}_t = \{y_t^1, y_t^2, \dots y_t^{n_t}\}$, which is a subset of the full ontology of event types $\mathcal{Y}$. Here, unseen events and negative instances (i.e. text spans that do not trigger any event) are treated as NA. After training on $\mathcal{D}_t$, the model is expected to be able to detect all seen events thus far, i.e. $\mathcal{Y}_1 \bigcap \mathcal{Y}_2 \dots \bigcap \mathcal{Y}_t$. To this end, we employ two commonly used techniques in Rehearsal-based Continual Learning: *Naive Replay*, and *Knowledge Distillation* (Hinton et al., 2015). Let R_{t-1} be the replay buffer up to task $t-1$, the Replay Loss and Knowledge Distillation loss are written as follows:

$$\mathcal{L}_R = -\frac{1}{|\mathcal{R}_{t-1}|} \sum_{(h,y)\in R_{t-1}} \log p^t(y|h), \quad (4)$$

$$\mathcal{L}_D = -\sum_{(h,y)\in R_{t-1}} p^{t-1}(y|h) \log p^t(y|h), \quad (5)$$

where p^t denotes the probability of predictions given by the model instance at timestep t.

3 Lifelong Event Detection via Optimal Transport

We incorporate Optimal Transport (OT) as a foundational element of our methodology. OT is a mathematical framework designed to compute the distance between two probability distributions with different supports.

In our methodology, OT is applied to align the probability distribution output of the classifier head with the distributional characteristics inherent in the vocabulary of the Pre-trained Language Model (PLM) head. The softmax class probabilities from the classifier head are transported to closely match the pre-trained distribution, facilitating a seamless integration of task-specific knowledge while minimizing the divergence from the model's preexisting linguistic understanding.

We forward each event trigger through a pre-trained language model and its original language modeling head, and obtain a distribution over a dictionary of V words:

$$x_s = Softmax(LMH(w_s')/\tau)$$
$$x_e = Softmax(LMH(w_e')/\tau)$$
$$\tilde{x} = (x_s + x_e)/2$$

where LMH is a pre-trained language model head, τ is temperature coefficient, and $\tilde{x}$ is distribution of the event trigger over dictionary.

Each event trigger is associated with a distribution over C classes: $\boldsymbol{p} \in \Delta^C$, where each entry indicates the probability that the event trigger belongs to a class in the ontology. An encoder is employed to generate $\boldsymbol{p}$ from $\boldsymbol{x}$, defined as $\boldsymbol{p} = \mathrm{softmax}(\theta(x))$, where θ represents the parameters of the neural network as described in Section 2.1.

Given that $\tilde{\boldsymbol{x}}$ and $\boldsymbol{p}$ are distributions with different supports for the same event trigger, we aim to train the model by minimizing the following Optimal Transport (OT) distance to push $\boldsymbol{p}$ towards $\tilde{\boldsymbol{x}}$:

$$d_{\mathbf{M}}(\tilde{\boldsymbol{x}}, \boldsymbol{p}) := \min_{\mathbf{P} \in U(\tilde{\boldsymbol{x}}, \boldsymbol{p})} \langle \mathbf{P}, \mathbf{M} \rangle, \tag{6}$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product; the cost matrix $\mathbf{M} \in \mathbb{R}{\geq}\, 0^{V \times C}$ captures semantic distances between class c and word v, with each entry m_{vc} signifying the importance of words in the corresponding class; $\mathbf{P} \in \mathbb{R}_{>0}^{V \times C}$ denotes the transport plan; and and $U(\tilde{\boldsymbol{x}}, \boldsymbol{p})$ is defined as the set of all viable transport plans. Considering two discrete random variables $X \sim \mathrm{Categorical}(\tilde{\boldsymbol{x}})$ and $Y \sim \mathrm{Categorical}(\boldsymbol{p})$, where the transport plan $\mathbf{P}$ becomes a joint probability distribution of (X, Y), i.e., $\mathrm{p}(X = i, Y = j) = p_{ij}$: the set $U(\tilde{\boldsymbol{x}}, \boldsymbol{p})$ encompasses all possible joint probabilities that satisfy the specified constraints, forming a transport polytope.

Directly optimizing Eq. (6) poses a time-consuming challenge. To address this, an entropic-constrained regularized optimal transport (OT) distance is introduced, known as the Sinkhorn distance:

$$s_{\mathbf{M}}(\tilde{\boldsymbol{x}}, \boldsymbol{p}) := \min_{\mathbf{P} \in U(\tilde{\boldsymbol{x}}, \boldsymbol{p})} \langle \mathbf{P}, \mathbf{M} \rangle - \mathbf{H}(\mathbf{P}), \tag{7}$$

where the entropy function of the transport plan $\mathbf{H}(\mathbf{P}) \overset{\text{def}}{=} -\sum_{i,j} \mathbf{P}_{i,j}(\log(\mathbf{P}_{i,j} - 1))$ is the regularizing function (Cuturi, 2013).

The cost matrix $\mathbf{M}$ is a trainable variable in our model. To overcome the challenge of learning the cost function, we propose a specific construction for $\mathbf{M}$:

$$m_{vc} = 1 - \cos(\mathbf{e}_v, \mathbf{g}_c), \tag{8}$$

where $\cos(\cdot, \cdot)$ represents the cosine similarity, and $\mathbf{g}_c \in \mathbb{R}^D$ and $\mathbf{e}_v \in \mathbb{R}^D$ are the embeddings of class c and word v, respectively. After training on one task, the learned class embeddings are frozen. We then expand the size of the class embeddings and train the newly initialized embeddings on the new task.

Frogner et al. (2015) further suggested combining the OT loss with a conventional cross-entropy loss to better guide the model. By parameterizing $\boldsymbol{M}$ with $\boldsymbol{G}$, the collection of class embeddings, the final OT objective function is expressed as:

$$\mathcal{L}_{\mathcal{OT}} = \min_{\theta, \mathbf{G}} \left[s_{\mathbf{M}}(\tilde{\boldsymbol{x}}, \boldsymbol{p}) - \epsilon \tilde{\boldsymbol{x}} \log\phi(\boldsymbol{p}) \right]. \tag{9}$$

To maintain the consistency of class representations across tasks, an additional regularization term enforces the proximity of class representations in the current task to those in the most recent task:

$$\mathcal{L}_G = ||\mathbf{G}_t - \mathbf{G}_{(t-1)}||^2. \tag{10}$$

Finally, we can write our final objective function:

$$\mathcal{L} = \mathcal{L}_C + \mathcal{L}_R + \mathcal{L}_D + \mathcal{L}_{OT} + \alpha \mathcal{L}_G, \tag{11}$$

where α is the regularization coefficient.

Avoiding Catastrophic Forgetting Similar to many CED baselines, our method incorporates a replay process. However, our approach to constructing the memory buffer is distinct. For each class in the training data, we retain the prototype mean μ and diagonal covariance Σ of its trigger representations encountered by the model, rather than storing explicit data samples. During replay, synthetic samples are generated from these prototypes and combined with the replay buffer $\mathcal{R}$ to form the effective buffer $\tilde{\mathcal{R}}$. This modified buffer replaces $\mathcal{R}$ in the computation of $\mathcal{L}_R$ (4) and $\mathcal{L}_D$ (5).

4 Experiments

4.1 Settings

Datasets We employ two datasets in our experiments: ACE 2005 (Walker et al., 2006) and MAVEN (Wang et al., 2020); both are preprocessed

Task	MAVEN					ACE				
	1	2	3	4	5	1	2	3	4	5
BIC	63.16	55.51	53.96	50.13	49.07	55.88	58.16	61.23	59.72	59.02
KCN	63.16	55.73	53.69	48.86	47.44	55.88	58.55	61.40	59.48	58.64
KT	62.76	58.49	57.46	55.38	54.87	55.88	57.29	61.42	60.78	59.82
EMP	66.82	58.02	58.19	55.07	54.52	59.05	57.14	55.80	53.42	52.97
ESCO	67.50	61.37	60.65	57.43	57.35	—-	—-	—-	—-	—-
SCR	**76.52**	57.97	57.89	52.74	53.41	**75.24**	**63.3**	61.07	55.05	55.37
SharpSeq	62.28	61.85	62.92	61.31	60.27	56.47	56.99	64.44	62.47	62.60
LEDOT-OT	63.34	59.89	59.28	56.24	55.20	58.74	58.08	61.81	58.32	59.76
LEDOT-R	63.01	60.16	59.76	56.75	54.59	58.30	58.60	63.14	58.82	60.18
LEDOT-P	63.01	59.95	59.32	56.10	55.21	59.95	56.63	62.09	60.08	61.41
LEDOT	62.98	60.47	60.78	58.53	57.53	58.30	59.69	63.52	61.05	63.22
LEDOT + SharpSeq	63.30	**61.97**	**63.00**	**61.81**	**61.49**	60.15	59.73	**64.55**	**63.65**	**64.27**
Upperbound	/	/	/	/	*64.14*	/	/	/	/	*67.95*

Table 1: Classification F1-scores (%) on 2 datasets MAVEN and ACE. *Upperbound* indicates the theoretical maximum achievable performance when BERT is frozen.

similar to Yu et al.'s (2021) work. To ensure fairness, we rerun all baselines on the same preprocessed datasets. The detailed statistics of the two datasets can be found in Appendix A.2.

Experimental Settings We adopt the Oracle negative setting, as mentioned by Yu et al. (2021), to evaluate all methods in continual learning scenario. This setting involves excluding the learned types from previous tasks in the training set of the new task, except for the NA (Not Applicable) type. Labels for future tasks are treated as NA type. Assessments are conducted using the exact same task permutations as in Yu et al.'s (2021) work. The performance metric is the average terminal F1 score across 5 permutations after each task. Recently, (Le et al., 2024) introduced a multi-objective optimization method that is compatible with our proposed LEDOT approach. To examine the impact of LEDOT on SharpSeq, we conducted an experiment referred to as LEDOT+SharpSeq. For details on other baselines and the integration of LEDOT with SharpSeq, please refer to Appendix A.1.

4.2 Experimental Results

Table 1 showcases the impressive results of our proposed method, **LEDOT**, compared to state-of-the-art baselines in continual event detection. On both the MAVEN and ACE datasets, LEDOT consistently achieves higher F1 scores, surpassing most baseline methods. When combined with SharpSeq, LEDOT further enhances performance, increasing the F1-score by a significant margin of 1.22% on MAVEN and 1.67% on ACE after five tasks.

We also conduct further ablation studies to evaluate variants of **LEDOT**: LEDOT-OT (without Optimal Transport), LEDOT-R (without the replay set), and LEDOT-P (without prototype latent representations). Even without prototype rehearsal, LEDOT-P with OT surpasses the replay-based baseline KT by 0.34% on MAVEN and 1.59% on ACE. Moreover, LEDOT outperforms LEDOT-OT, highlighting the crucial role of OT in preventing catastrophic forgetting. Specifically, OT improves F1 scores by 2.33% on MAVEN and 3.46% on ACE. These results emphasize the importance of OT in mitigating catastrophic forgetting in continual event detection.

5 Conclusion

Harnessing the inherent linguistic knowledge from pre-trained language modeling heads in encoder-based language models play a pivotal role in enhancing performance in downstream tasks. With the introduction of LEDOT, we present a novel approach utilizing optimal transport to align the learning of each task with a common reference—the pre-trained distribution of the vocabulary. This alignment mitigates overfitting to the current task and effectively addresses the challenge of catastrophic forgetting. Our method, demonstrating superior performance across various benchmarks, stands as a testament to the effectiveness of leveraging pre-trained language modeling heads for continual event detection, offering a promising avenue for future research in this domain.

Limitations

Being an empirical study into the effectiveness of Optimal Transport in aligning the output distribution of Continual Event Detection models, our work is not without limitations. We acknowledge this, and would like to discuss our limitations as follows:

- The method proposed in this paper is orthogonal to the tasks of interest and the specific techniques to solve them. With that being said, our method is applicable to a wide range of information extraction tasks, such as Named Entity Recognition, and Relation Extraction, as well as other text classification tasks, such as Sentiment Analysis. However, given limited time and computational resources, we limit the scope of our experiments to only Event Detection. The extent to which our proposed method can work with other NLP problems can be an interesting topic that we leave for future work. Nevertheless, our experimental results suggest that using Optimal Transport to align the output distribution of the model with the pre-trained language modeling head has the potential to improve continual learning performance on other problems as well.

- This paper presents the empirical results of our LEDOT method using a pre-trained encoder language model (i.e. BERT) as the backbone. Meanwhile, large decoder-only language models, with their heavily over-parameterized architectures, amazing emergent ability, and great generalization capability, have emerged and become the center of focus of NLP research in recent years. Though they have proved to be able to understand language and solve almost all known NLP tasks without needing much fine-tuning, many studies (Lai et al., 2023; Qiu and Jin, 2024; Zhong et al., 2023) suggested that even the largest models like ChatGPT (Ouyang et al., 2022) still lag behind smaller but specialized models such as BERT (Devlin et al., 2019) and T5 (Raffel et al., 2023) by a significant margin on tasks like Event Detection. We thus believe that studies on the applications of encoder language models in Continual Event Detection are still needed.

References

Eden Belouadah and Adrian Popescu. 2019. Il2m: Class incremental learning with dual memory. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 583–592.

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.

Arslan Chaudhry, Albert Gordo, Puneet Dokania, Philip Torr, and David Lopez-Paz. 2021. Using hindsight to anchor past knowledge in continual learning. In *Proceedings of the AAAI conference on artificial intelligence*, pages 6993–7001.

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.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

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.

Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya, and Tomaso A Poggio. 2015. Learning with a wasserstein loss. *Advances in neural information processing systems*, 28.

Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. 2023. Exploring the feasibility of chatgpt for event extraction. *arXiv preprint arXiv:2303.03836*.

Ridong Han, Tao Peng, Chaohao Yang, Benyou Wang, Lu Liu, and Xiang Wan. 2023. Is information extraction solved by chatgpt? an analysis of performance, evaluation criteria, robustness and errors.

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.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Viet Lai, Nghia Ngo, Amir Pouran Ben Veyseh, Hieu Man, Franck Dernoncourt, Trung Bui, and Thien Nguyen. 2023. ChatGPT beyond English: Towards a comprehensive evaluation of large language models in multilingual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13171–13189, Singapore. Association for Computational Linguistics.

Thanh-Thien Le, Viet Dao, Linh Van Nguyen, Thi-Nhung Nguyen, Linh Van Ngo, and Thien Huu Nguyen. 2024. Sharpseq: Empowering continual event detection through sharpness-aware sequential-task learning. In *2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Minqian Liu, Shiyu Chang, and Lifu Huang. 2022. Incremental prompting: Episodic memory prompt for lifelong event detection. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2157–2165, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Chengwei Qin, Ruirui Chen, Ruochen Zhao, Wenhan Xia, and Shafiq Joty. 2024. Lifelong event detection with embedding space separation and compaction.

Yunjian Qiu and Yan Jin. 2024. Chatgpt and finetuned bert: A comparative study for developing intelligent design support systems. *Intelligent Systems with Applications*, 21:200308.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

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

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32.

Gobinda Saha, Isha Garg, and Kaushik Roy. 2021. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, and Hao Wang. 2024. Continual learning of large language models: A comprehensive survey.

Ghada Sokar, Decebal Constantin Mocanu, and Mykola Pechenizkiy. 2021. Spacenet: Make free space for continual learning. *Neurocomputing*, 439:1–11.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 multilingual training corpus LDC2006T06. Web Download. Philadelphia: Linguistic Data Consortium.

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. *arXiv preprint arXiv:2004.13590*.

Zitao Wang, Xinyi Wang, and Wei Hu. 2023. Continual event extraction with semantic confusion rectification.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. 2019. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382.

Weimin Xiong, Yifan Song, Peiyi Wang, and Sujian Li. 2023. Rationale-enhanced language models are better continual relation learners. *arXiv preprint arXiv:2310.06547*.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. 2017. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*.

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.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45.

Kang Zhao, Hua Xu, Jiangong Yang, and Kai Gao. 2022. Consistent representation learning for continual relation extraction. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3402–3411, Dublin, Ireland. Association for Computational Linguistics.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2023. Can chatgpt understand too? a comparative study on chatgpt and fine-tuned bert.

A Additional Experimental Details

A.1 Baselines

The following continual learning and continual ED methods are employed as baselines in this paper:

- **BIC** (Wu et al., 2019) addresses model bias towards new labels via an affine transformation.
- **KCN** (Cao et al., 2020) employs a limited set to store data for replay, utilizing knowledge distillation and prototype-enhanced retrospection to alleviate catastrophic forgetting.
- **KT** (Yu et al., 2021) follows a memory-based approach, combining knowledge distillation with knowledge transfer. This method utilizes new-label samples to reinforce the model's retention of old knowledge and employs old-label samples to initialize representations for new-label data in the classification layer.
- **EMP** (Liu et al., 2022) also leverages knowledge distillation and introduces straight prompts into the input text to retain previous knowledge.
- **ESCO** (Qin et al., 2024) introduce ESCO, a method combining Embedding Space Separation and Compaction. ESCO pushes the feature distribution of new data away from old data to reduce interference and pulls memory data towards its prototype to improve intra-class compactness and alleviate overfitting on the replay dataset.
- **SharpSeq** The framework introduced in **SharpSeq** (Le et al., 2024) integrates multi-objective optimization (MOO) with sharpness-aware minimization (SAM). In the context of continual learning, handling multiple losses often involves simply summing them with fixed coefficients. However, this approach can lead to gradient conflicts that hinder the discovery of optimal solutions. MOO algorithms address this issue by dynamically estimating coefficients based on the gradients of the losses. To refine the results of MOO, (Le et al., 2024) employs SAM to identify flat minima along the Pareto front.
- **SCR** (Wang et al., 2023) employs a training approach involving both BERT and the classifier layer. Initially, this yields high F1 scores on early tasks, but performance deteriorates rapidly as more tasks are encountered. In contrast, our method maintains BERT's parameters fixed during training. The SCR approach, which fine-tunes BERT, presents challenges for continual event detection. Despite having different label sets, many sentences are recurrent across tasks. SCR tackles this by using pseudo labels from the previous stage to predict labels on new datasets, containing sentences from previous tasks. However, this strategy leads to data leakage from old tasks to new ones, significantly inflating SCR's replay dataset beyond what is allowed in strict continual learning setups. In contrast, our method relies on a frozen BERT for feature extraction, ensuring consistency in trigger representations over time. Our approach aligns with the principles of continual learning, where the model solely accesses data relevant to the current task. Moreover, the evaluation metric in SCR differs from our approach, as they do not account for the NA label despite it being the most common label in these datasets. Therefore, we have reproduced the results and reported them in Table1.
- **LEDOT + SharpSeq** Our proposed method incorporates two key objectives: one focusing on the OT loss for the language modeling head and another serving as a regularization

term to ensure the proximity of class representations. Instead of treating these objectives as separate entities within a multi-objective optimization algorithm, we integrate them directly into the overall loss calculation using the same data. This approach maintains the original number of losses, streamlining the optimization process.

A.2 Datasets

Detailed statistics regarding the datasets used for all empirical assessments can be found in Table 2.

A.3 Implementation details

In our experiments, the encoder and language model head is taken from BERT-large-cased (Devlin et al., 2019) and they are freezed in the training process. We employ the AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of 1×10^{-4} and a weight decay of 1×10^{-2}. Model training continues until there is no increase in performance on the development set. The replay setting remains consistent with KT Yu et al.'s (2021), where the number of instances for each label in the replay set is set to 20. Since the size of the vocabulary is large and it contains many subwords and completely unrelated words, to reduce the computation, we select only a subset of words that are verbs. In each batch, we combine that set with tokens in the batch to compute the OT loss.

For each method, we determine the appropriate settings through a grid search. The hyperparameter search ranges are as follows:

- The batch size ranges from 128 to 512.
- The number of epochs ranges from 15 to 30.
- The temperature of the language modeling head τ is in $[0.01, 0.1, 1, 2, 3, 4, 5]$.
- The regularization coefficient α is in $[0.1, 0.2, 0.5, 1]$
- The prototype sampling ratio r is in $[1, 5, 10, 20]$.
- the balancing coefficient η to balance NA label and valid labels is in $[\frac{4}{5}, \frac{10}{11}, \frac{20}{21}, \frac{30}{31}, \frac{40}{41}]$

All implementations are coded using PyTorch, and the experiments were carried out on NVIDIA A100 and NVIDIA V100 GPUs.

B Ablation Study

B.1 Temperature of Language Modeling Head

We conduct an ablation study to explore the impact of different temperatures in the language modeling head within the LEDOT method. The motivation behind this study lies in the stochastic nature of the language modeling process, where a higher temperature introduces more randomness. This increased stochasticity can influence the generation not only of the primary label (event type) but also of other words related to the topic. By systematically varying the temperature parameter, denoted as τ, we aim to understand how these different levels of stochasticity affect LEDOT's performance. The results are presented in Table 3.

B.2 Quantity of Generated Samples

In Table 1, we observe that the performance of LEDOT significantly improves when synthesizing representations from prototypes. To further investigate this effect, we conducted additional experiments with LEDOT, varying the ratios (r) between the number of generated samples and the replay set. The outcomes for four r values are presented in Table 4. Notably, on MAVEN, the highest performance is achieved with $r = 10$, yielding a 57.53% F1 score in the fifth task. Conversely, for the fifth task on ACE, the optimal r value is 2020, resulting in a 63.22% score. The influence of prototype sampling on early tasks is relatively marginal, but it becomes more pronounced in later tasks. It is important to note that an increased r value does not necessarily guarantee improved LEDOT performance. This can be attributed to the noise introduced by random processes during representation sampling. The noise can impact the outcome of the language modeling head in LEDOT and potentially misguide the classification head during model optimization. Therefore, when generating more samples, careful consideration is required to mitigate noise effects and avoid adversarial impacts.

B.3 Others

We conduct additional ablation studies to gain deeper insights into the performance of LEDOT. First, we compare the impact of two different initialization methods for Optimal Transport—random initialization and initializing labels by mapping them to their corresponding word embeddings in the vocabulary. The results of this comparison are detailed in Table 5, shedding light on the influence of initialization strategies on the overall effectiveness of LEDOT. Second, we explore the sensitivity of our method to the coefficient of regularization applied to the cross-task class representations. The results of this investigation are presented in Table 6, providing valuable information about the robustness of LEDOT to variations in the regularization coefficient. These ablation studies contribute to a comprehensive understanding of the factors influencing LEDOT's performance in continual event detection scenarios.

C Optimal Transport on Continual Relation Extraction

Our proposed Optimal Transport alignment extends beyond Continual Event Detection: it can also enhance other continual NLP solutions utilizing various kinds of pre-trained language models. To substantiate this claim, we demonstrate its effectiveness in Continual Relation Extraction (CRE) (Han et al., 2020; Cui et al., 2021; Zhao et al., 2022; Xiong et al., 2023) using an *encoder-decoder* language model, specifically T5 (Raffel et al., 2020).

Our experiments are centered around the state-of-the-art CRE baseline RationaleCL (Xiong et al., 2023). This method leverages rationales generated by `ChatGPT-3.5`[1] during training to enhance the T5 model for CRE. RationaleCL operates by first generating rationales for current relation samples using an LLM. These rationales are then integrated into the original training dataset for multi-task rationale tuning. Formally, RationaleCL introduces three key objectives:

$$Task_c : x_i \longrightarrow y_i \tag{12}$$

$$Task_r : x_i \longrightarrow r_i + y_i \tag{13}$$

$$Task_d : x_i + r_i \longrightarrow y_i \tag{14}$$

where x_i represents the input text, y_i denotes the relation label, and r_i stands for the rationale. $Task_c$ directly generates the label y_i from the input x_i, while $Task_r$ requires the model to generate an explanation before generating an answer. $Task_d$ uses both the input and rationale in the encoder part to answer the question. It is noteworthy that, similar to most continual learning methods, RationaleCL employs a replay process. This process trains the model on both newly encountered data and a limited amount of samples from previously encountered tasks stored in the buffer.

[1] https://chat.openai.com/

The state-of-the-art performance achieved by RationaleCL in CRE underscores its efficacy. However, our integration of Optimal Transport (OT) methodologies aims to elevate the method to new heights. We introduce OT objectives to align the learned language-modeling head with T5's original language-modeling head, resulting in the enhancements observed over the baseline on the TACRED dataset (Zhang et al., 2017), as showcased in Table 7.

Our integration of OT objectives not only mitigates the detrimental effects of catastrophic forgetting but also emerges as a compelling solution for enhancing the fine-tuning process across various downstream tasks.

D Reproducibility Checklist

- **Source code with the specification of all dependencies, including external libraries**: The source code and the necessary documentation for reproducibility are submitted together with this paper via the ACL Rolling Review submission system.

- **Description of computing infrastructure used**: We use a Tesla V100-SXM2 GPU with 32GB memory operated by Ubuntu Server 18.04.3 LTS, a Tesla A100-SXM GPU with 40GB memory operated by Ubuntu 20.04, and NVIDIA Tesla T4 with 16GB operated by Ubuntu 20.04. PyTorch 1.9.1 and Huggingface-Transformer 4.23.1 (Apache License 2.0) (Wolf et al., 2019) are used to implement the models.

- **Number of parameters in the model**: The total number of parameters of our model is 335M parameters. Since we freeze the BERT model; the number of trainable parameters is thus only 1.4M.

- **Explanation of evaluation metrics used, with links to code**: We use the same performance measures (average F1-scores on 5 permutations of task orders) as in previous work (Yu et al., 2021) for fair comparisons.

- **Bounds for each hyper-parameter**: Please refer to Section A.3.

- **The method of choosing hyper-parameter values and the criterion used to select among them**: The hyperparameters are tuned using random search. Hyper-parameters are chosen based on F1 scores on the development set.

	MAVEN				ACE			
	#Doc	#Sentence	#Mention	#Negative	#Doc	#Sentence	#Mention	#Negative
Train	2522	27983	67637	280151	501	18246	4088	261027
Dev	414	4432	10880	46318	41	1846	433	53620
Test	710	8038	18904	79699	55	689	790	93159

Table 2: Statistics of two datasets. #Doc stands for the total number of documents.

	MAVEN					ACE				
Task	1	2	3	4	5	1	2	3	4	5
$\tau = 5$	**63.15**	**60.78**	60.66	58.51	57.37	57.41	59.00	63.60	60.87	61.81
$\tau = 4$	63.08	60.72	60.71	**58.76**	**57.71**	61.09	**60.12**	63.36	61.09	61.15
$\tau = 3$	63.06	60.77	60.70	58.43	57.30	58.09	59.46	63.98	**61.63**	62.36
$\tau = 2$	63.11	60.64	60.70	58.45	57.50	58.30	59.69	63.52	61.05	**63.22**
$\tau = 1$	62.98	60.47	**60.78**	58.53	57.53	60.42	59.76	**64.28**	61.52	62.84
$\tau = 0.1$	62.52	60.31	60.51	58.31	57.13	61.51	57.01	62.94	60.18	61.22
$\tau = 0.01$	62.60	60.3	60.43	58.22	57.15	**62.15**	57.08	63.51	59.48	61.29

Table 3: Ablation results for the temperature of the language modeling head in the LEDOT method.

	MAVEN					ACE				
Task	1	2	3	4	5	1	2	3	4	5
$r = 20$	63.01	60.12	60.26	57.96	56.87	58.30	59.69	63.52	61.05	**63.22**
$r = 10$	62.98	**60.47**	**60.78**	**58.53**	**57.53**	58.30	**60.80**	**64.63**	**62.47**	62.63
$r = 5$	63.01	60.30	60.54	58.22	57.01	58.30	61.06	64.67	60.59	62.29
$r = 1$	**63.07**	60.16	60.00	57.07	55.84	58.30	60.51	64.24	60.15	62.18

Table 4: Ablation results for the number of generated representations in the LEDOT method.

	MAVEN					ACE				
Task	1	2	3	4	5	1	2	3	4	5
random	**63.15**	**60.78**	60.66	58.51	57.37	57.41	59.00	**63.60**	60.87	**61.81**
mapping	63.08	60.72	**60.71**	**58.76**	**57.71**	**61.09**	**60.12**	63.36	**61.09**	61.15

Table 5: Ablation results for the initialization of Optimal Transport in the LEDOT method. "mapping" indicates initializing labels by mapping them to their corresponding word embeddings in the vocabulary.

	MAVEN					ACE				
Task	1	2	3	4	5	1	2	3	4	5
$\alpha = 1$	63.01	60.36	60.67	58.33	57.41	58.30	59.72	64.41	60.97	62.29
$\alpha = 0.5$	62.98	60.47	**60.78**	**58.53**	**57.53**	58.30	59.69	63.52	61.05	**63.22**
$\alpha = 0.2$	**63.07**	**60.66**	60.67	58.37	57.16	58.72	59.39	64.55	61.88	62.68
$\alpha = 0.1$	63.01	60.45	60.60	57.79	57.02	58.72	**60.01**	**64.61**	**62.49**	62.82

Table 6: Ablation results for regularization on cross-task class representations in the LEDOT method.

	TACRED									
Task	1	2	3	4	5	6	7	8	9	10
RCL	100.00	94.80	92.20	89.24	86.56	84.74	80.57	77.46	80.98	79.11
OT RCL	97.76	98.40	93.17	87.94	90.18	86.05	82.73	80.61	82.61	79.36

Table 7: Classification accuracy (%) on the TACRED dataset. RCL abbreviates for RationaleCL.