Event Process Typing via Hierarchical Optimal Transport

Bo Zhou[1,2], **Yubo Chen**[1,2], **Kang Liu**[1,2,3], **Jun Zhao**[1,2]

[1] School of Artificial Intelligence, University of Chinese Academy of Sciences
[2] National Laboratory of Pattern Recognition, CASIA
[3] Beijing Academy of Artificial Intelligence
{bo.zhou,yubo.chen,kliu,jzhao}@nlpr.ia.ac.cn

Abstract

Understanding intention behind event processes in texts is important to many applications. One challenging task in this line is event process typing, which aims to tag the process with one action label and one object label describing the overall action of the process and object the process likely affects respectively. To tackle this task, existing methods mainly rely on the matching of the event process level and label level representation, which ignores two important characteristics: Process Hierarchy and Label Hierarchy. In this paper, we propose a **H**ierarchical **O**ptimal **T**ransport (HOT) method to address the above problem. Specifically, we first explicitly extract the process hierarchy and label hierarchy. Then the HOT optimally matches the two types of hierarchy. Experimental results show that our model outperforms the baseline models, illustrating the effectiveness of our model.

Introduction

Being able to understand the overall intention behind event processes in texts is important to many artificial intelligence applications which require the ability to reason about chains of activities, such as script event prediction (Lv, Zhu, and Hu 2020), question answering (Li et al. 2020a), and event based summarization (Li and Zhang 2020). One challenging task in this line is event process typing (Chen et al. 2020). Given an event process composed of a sequence of events, the task aims to tag the process with two labels: one action label describing the overall action of the process and one object label describing the overall object the process likely affects. An example of the event process typing is showed in Figure 1, given an event process of four events, the task tags one action label *book* and one object label *flight*.

To tackle the event process typing task, previous methods (Chen et al. 2020) leverage pre-trained language model to encode the whole process and the action label as process feature vector and action label vector respectively, then the action whose feature vector is the closest to the process is predicted as the action label for the process. The object label for the process is predicted in the same way. Though effective, they ignore two important characteristics: Process Hierarchy and Label Hierarchy, which can be leveraged to improve performance of the task. In the following, we will elaborate the two characteristics and the reasons why they are important for the event process typing task.

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

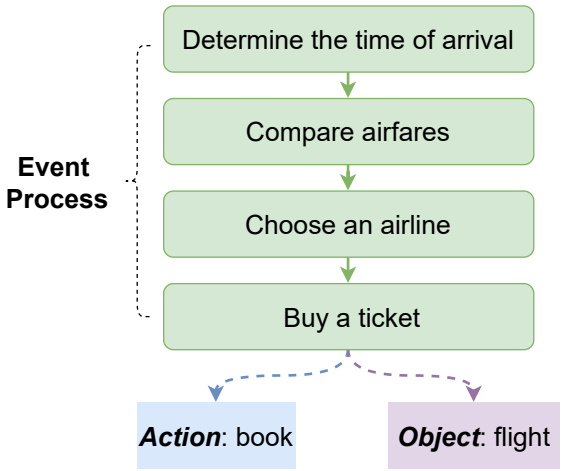

Figure 1: An example of the event process typing task. The action label of the event process is *book* and the object label is *flight*.

Process Hierarchy: Event process is composed of events, and event is composed of words, previous methods ignore this hierarchy since their predictions mainly rely on the event process-level representation. Thus they failed to directly match labels with events in the process and labels with words in the events, which can provide fine-grained prediction clues for the task. For example in Figure 1, the words *airfares* and *airline* strongly indicate the object label of *flight*, while the event *Buy a ticket* implies that the action is likely to be *book*. Thus, a challenging problem is to model the process hierarchy and leverage it to capture the matching with action and object labels.

Label Hierarchy: Due to the large label space (1,336 action types and 10,441 object types) of the event process typing, it's difficult to directly capture the association between a process and a single label. Meanwhile, the action label and object label of this task are verb and noun respectively, both of which have hypernyms. The verb-hypernyms (noun-hypernyms) hierarchy can provide additional information for the original label consisting of only one verb (noun).

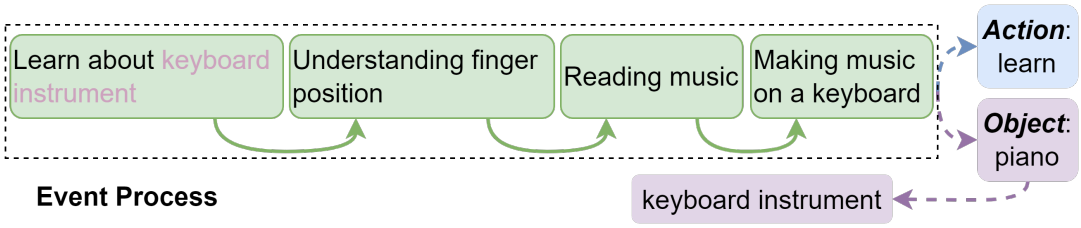

Figure 2: An event process containing four events, its action label and object label are *learn* and *piano* respectively. The *keyboard instrument* is a hypernym of the object *piano*.

For example in Figure 2, the hypernym *keyboard instrument* of the object *piano* can match the first event in the process, hence it can provide additional information and help the object prediction. Therefore, another problem is how to leverage label hierarchy for event process typing.

In order to utilize both the process hierarchy and the label hierarchy and optimally capture the interaction between them, we propose a novel method termed **H**ierarchical **O**ptimal **T**ransport (HOT). Specifically, for the process hierarchy, we explicitly extract semantics of each event in the process and semantics of each word in event. For the label hierarchy, we leverage external knowledge base to extract hierarchical label for each label. Then the HOT is utilized to optimally match label semantics with event semantics and word semantics hierarchically.

To recap, our contributions can be summarized as follows:

- We consider two types of hierarchy: process hierarchy (word-event-event process) and label hierarchy (verb-hypernyms and noun-hypernyms) to help the event process typing task.
- We propose a novel method for event process typing termed Hierarchical Optimal Transport, which optimally matches label semantics with event semantics and word semantics hierarchically.
- Experimental results show that our model outperforms the baseline models, which demonstrate the effectiveness of our model.

Related Work

Script Event Learning

Researchers have done a lot of work on script events. The first is script event prediction, which is to predict the end event of a given event chain. It includes the early script event prediction method based on statistical information (Chambers and Jurafsky 2008; Jans et al. 2012; Peyré, Cuturi et al. 2019), and the recent script event prediction method based on neural network (Pichotta and Mooney 2016; Granroth-Wilding and Clark 2016; Lv, Zhu, and Hu 2020; Bai et al. 2021; Zhou et al. 2021; Wang et al. 2021). The second line of work on script events is script event generation. Zhang et al. (2020) propose an analogous process structure induction framework that leverages analogies between processes and the conceptualization of subevent instances to predict subevent sequences of previously unseen processes. To handle the task of story ending generation, Huang et al. (2021) propose a context-aware multi-level graph convolutional network dependency analysis tree to capture dependencies and context clues more efficiently. Li et al. (2022b) propose a novel coarse-grained to fine-grained two-stage approach that more explicitly generates subsequent events in open-ended text generation. Another task related to script events is event process typing which we focus on in this paper, Chen et al. (2020) propose a model that utilizes a pre-trained language model to encode the entire process and action labels into process feature vectors and action label vectors, respectively, and then predicts the action whose feature vector is closest to the process as the action label of the process. Object labels for processes are predicted in the same way. In this paper, we consider two types of hierarchy and propose a method which optimally matches the hierarchy for event process typing task.

Optimal Transport in NLP

Many recent NLP tasks have exploited optimal transport. In information extraction related tasks, Li et al. (2021) introduce a time-aware optimal transport distance to learn the compression model for compressing the whole event-graph to its salient sub-graph for the timeline summarization task. Veyseh et al. (2022) employ optimal transport to induce document structures based on sentence-level syntactic structures for document-level event argument extraction task. Li et al. (2022a) design an event graph alignment loss based on optimal transport to capture event argument structures, to enforce vision-language pretraining models to comprehend events and associated argument roles. In text generation task, to tackle the exposure bias problem, Li et al. (2020b) propose using student-forcing optimal transport to train the text-generation model together with maximum likelihood estimation. In multilingual word representation task, an alignment objective using optimal transport during fine-tuning is proposed (Alqahtani et al. 2021) to further improve multilingual contextualized representations for downstream cross-lingual transfer. In text style transfer task, to simultaneously incorporate syntactic and semantic information, Nouri (2022) proposes a novel method based on optimal transport for text style transfer to compute similarity between the source and the converted text. In this paper, we exploit optimal transport and propose a method termed hierarchical optimal transport to match different hierarchies in event process typing task.

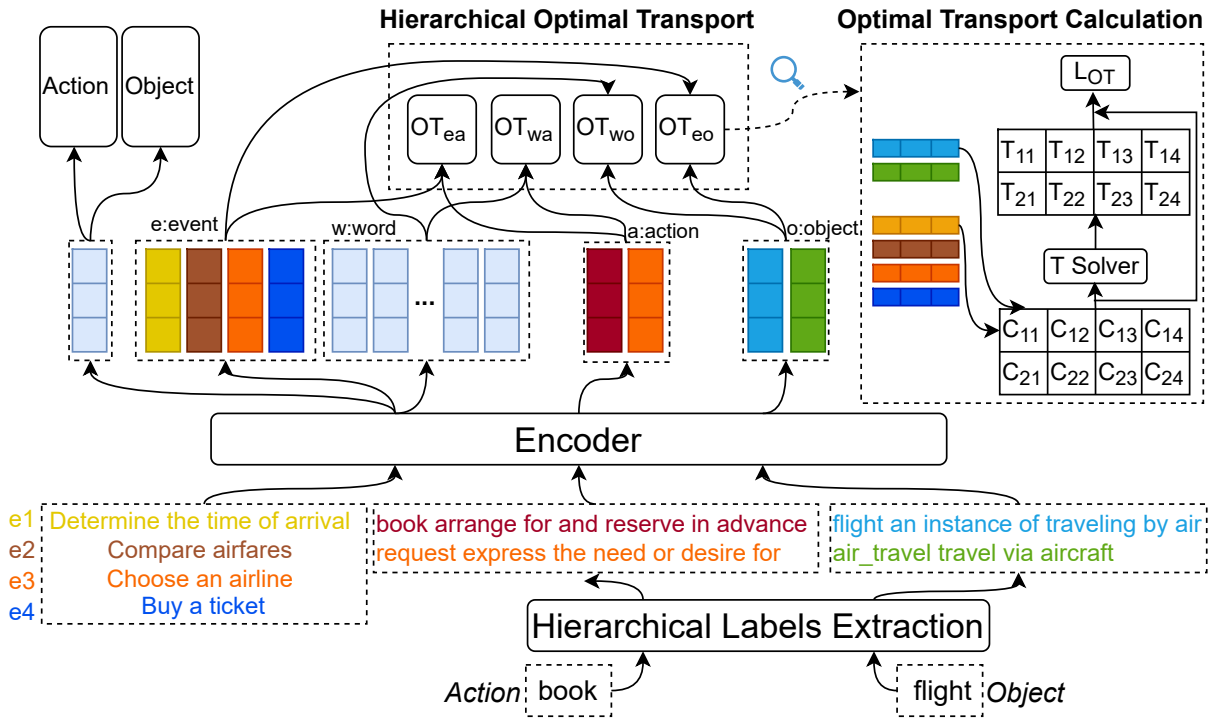

Figure 3: The Hierarchical Optimal Transport (HOT) architecture. The HOT calculates four losses OT_{ea}, OT_{wa}, OT_{wo} and OT_{eo}, each of which is obtained by the same optimal transport calculation process on the far right in the figure. For brevity, we only show one hypernym and its definition for the action label and object label.

Methodology

Task Formulation

Given an event process $p = (e_1, e_2, ..., e_n)$, the goal of the task (Chen et al. 2020) is to conceptualize the overall intention of the process into two labels, one is action label $a \in A$ and the other is object label $o \in O$, where A is the set of verbs and O is the set of nouns.

Following Chambers and Jurafsky (2008), we define the event process as a sequence of primitive events with a sharing protagonist. Each event e_i consists of a predicate a_i describing the action performed by the protagonist, and an object o_i describing the object of the action a_i. The action label a of the process p describes the overall action of p, and the object label o describes the most probable object the process p affects.

Hierarchical Label Extraction

The overall structure of our model is illustrated in Figure 3. We first leverage WordNet (Miller 1995) to extract hierarchical label for each action label $a \in A$ and object label $o \in O$. Specifically, for an action label a, we leverage its sense of verb type in WordNet to obtain its definition $f = (w_1, w_2, ..., w_{L_f})$, where w_i is a word and L_f is the length of the definition. We then obtain k hypernyms $(a_1, a_2, ..., a_k)$ of action label a together with their definitions $(f_1, f_2, ..., f_k)$ with length $L_{f_1}, L_{f_2}, ..., L_{f_k}$ respectively. Finally, we concatenate these information as the hierarchical label for the action label a, which is denoted as a sequence $H_a = (a, f, a_1, f_1, ..., a_k, f_k)$. For example, for the action label buy and $k = 1$, we finally obtain $H_{buy} = (buy, obtain\ by\ purchase, get, come\ into\ the\ possession\ of\ something)$.

For an object label o, we leverage its sense of noun type in WordNet to extract the definitions and hypernyms, and the hierarchical label of o is $H_o = (o, f, o_1, f_1, ..., o_k, f_k)$.

Feature Extraction

Following previous work (Chen et al. 2020), we use pretrained language model RoBERTa-base (Liu et al. 2019) to extract event process features and label features.

Event Process Feature Extraction Given an event process $p = (e_1, e_2, ..., e_n)$, we first concatenate the predicate a_i and object o_i of event e_i. Then the separator token $\langle /s \rangle$ of RoBERTa is added between events to help the model to differentiate them. That is, the final input sequence to RoBERTa is $s_p = (a_1, o_1, \langle /s \rangle, ..., \langle /s \rangle, a_n, o_n)$:

$$(\boldsymbol{h}_1, \boldsymbol{h}_2, ..., \boldsymbol{h}_{L_p}) = \text{RoBERTa}(s_p) \quad (1)$$

where L_p is the total number of tokens in the input sequence and $\boldsymbol{h}_i$ is the hidden state of the ith token. We then use mean-pooling of these hidden states to obtain two types of representations. One type is the representation $\boldsymbol{p}$ of the process by mean-pooling of all the hidden states, and the other is representations of events in the process by mean-pooling

of hidden states corresponding to each event, denoted as $(\boldsymbol{h}_{e_1}, \boldsymbol{h}_{e_2}, ..., \boldsymbol{h}_{e_n})$.

Label Feature Extraction For the hierarchical label of the action label a which is denoted as $H_a = (a, f, a_1, f_1, ..., a_k, f_k)$, we also add the separator token $\langle/s\rangle$ between hypernyms to obtain an input sequence $(a, f, \langle/s\rangle, a_1, f_1, \langle/s\rangle, ..., \langle/s\rangle, a_k, f_k)$ to RoBERTa. Then hidden states corresponding to tokens in input sequence are obtained in the same way as equation (1). We then use mean-pooling of hidden states corresponding to each hypernym and definition to get representations of the hierarchical label for the action label a, denoted as $(\boldsymbol{h}_1^a, \boldsymbol{h}_2^a, ..., \boldsymbol{h}_{k+1}^a)$.

For the hierarchical label $H_o = (o, f, o_1, f_1, ..., o_k, f_k)$ of the object label o, we use the same method as above to get the representations of the hierarchical label, denoted as $(\boldsymbol{h}_1^o, \boldsymbol{h}_2^o, ..., \boldsymbol{h}_{k+1}^o)$.

Hierarchical Optimal Transport

Discrete optimal transport. We first briefly introduce the optimal transport between discrete probability measures, which aims to minimize the distance between them. Assume two discrete probability measures α and β, which can be denoted as $\alpha = \sum_{i=1}^{n} a_i \delta_{\boldsymbol{x}_i}$ and $\beta = \sum_{j=1}^{m} b_i \delta_{\boldsymbol{y}_j}$ where $\delta_{\boldsymbol{x}}$ is the Dirac measure sitting at $\boldsymbol{x}$. The coefficients $\boldsymbol{a} = (a_1, a_2, ..., a_n)$ of α and the coefficients $\boldsymbol{b} = (b_1, b_2, ..., b_m)$ of β both sum to 1 because they are both probability measures. Under the setting of the above discrete probability measures, the optimal transport problem can be transformed into the following constrained optimization problem (Luise et al. 2018):

$$\mathcal{L}(\alpha, \beta) = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{a}, \boldsymbol{b})} \sum_{i=1}^{n} \sum_{j=1}^{m} \boldsymbol{T}_{ij} \cdot c\left(\boldsymbol{x}_i, \boldsymbol{y}_j\right) \tag{2}$$

where $\Pi(\boldsymbol{a}, \boldsymbol{b}) = \{\boldsymbol{T} \in \mathbb{R}_+^{n \times m} \mid \boldsymbol{T}\mathbf{1}_m = \boldsymbol{a}, \boldsymbol{T}^\top \mathbf{1}_n = \boldsymbol{b}\}$, and $\mathbf{1}_n$ is an n-dimensional all-one vector. $\boldsymbol{C}$ is the cost matrix with $\boldsymbol{C}_{ij} = c\left(\boldsymbol{x}_i, \boldsymbol{y}_j\right)$ and c is the cost function.

Optimal transport calculation. The minimum value $\boldsymbol{T}^*$ of Equation (2) can be approximated by the following IPOT algorithm (Xie et al. 2020). Given the number of iterations T, perform the following iterations,

$$\boldsymbol{Q} = \boldsymbol{A} \odot \boldsymbol{T}^{(t)}, \quad \boldsymbol{T}^{(t+1)} = diag(\boldsymbol{\delta}) \boldsymbol{Q} diag(\boldsymbol{\sigma}) \tag{3}$$

where $\odot$ is Hadamart product. The $\boldsymbol{\delta}$ and $\boldsymbol{\sigma}$ are obtained by iterating a fixed number of K times as shown below,

$$\boldsymbol{\delta} = \frac{1}{n \boldsymbol{Q} \boldsymbol{\sigma}}, \quad \boldsymbol{\sigma} = \frac{1}{m \boldsymbol{Q}^\top \boldsymbol{\delta}} \tag{4}$$

where n and m are the lengths of the two input sequences. The variables involved in equation (3) and equation (4) are initialized as follows,

$$\boldsymbol{\sigma} = \frac{1}{m} \mathbf{1}_{\boldsymbol{m}}, \boldsymbol{T}^{(1)} = \mathbf{1}_{\boldsymbol{n}} \mathbf{1}_{\boldsymbol{m}}{}^\top, \boldsymbol{A}_{ij} = e^{-\frac{\boldsymbol{C}_{ij}}{\beta}} \tag{5}$$

where $\frac{1}{\beta}$ is the generalized stepsize.

Now for the event process, through the feature extraction we get the event vector sequence $S_e = \{\boldsymbol{h}_{e_i}\}_{i=1}^{n}$ and the word vector sequence $S_w = \{\boldsymbol{h}_i\}_{i=1}^{L_p}$. For the action label, we also obtain the vector sequence $S_a = \{\boldsymbol{h}_i^a\}_{i=1}^{k+1}$. Then we compute the optimal transport for the action using the IPOT algorithm:

$$\mathcal{L}_a = OT_{ea} + OT_{wa} = \mathrm{IPOT}(S_e, S_a) + \mathrm{IPOT}(S_w, S_a) \tag{6}$$

where $OT_{ea} = \mathrm{IPOT}(S_e, S_a)$ is the loss bewteen the event vector sequence S_e and vector sequence S_a, obtained by the IPOT algorithm.

For the object label, we obtain the vector sequence $S_o = \{\boldsymbol{h}_i^o\}_{i=1}^{k+1}$, and we get the loss $\mathcal{L}_o$ for object similar to equation (6).

Algorithm 1: HOT algorithm

Input: Set of event processes with action and object $\{(p_i, a_i, o_i)\}$
Parameter: Batch size M

```
Initialize model parameters.
for epoch = 1, ..., MaxEpoch do
    for k = 1, ..., M do
        Draw an event process with action and object (p_i, a_i, o_i).
        Extract hierarchical labels for a_i and o_i.
        Extract word-level and event-level features for p_i.
        Extract features of hierarchical labels for a_i and o_i.
        Compute the losses OT_ea, OT_wa as equation (6), and OT_eo, OT_wo similar to (6).
    end for
    Update model parameters by optimizing loss in (9).
end for
```

Training and Optimization

Recall that in section of Feature Extraction we obtain the representation $\boldsymbol{p}$ of the process. Here we use $\boldsymbol{p}$ to score all the action labels and object labels. Specifically, $\boldsymbol{p}$ is input to a linear layer to score action labels:

$$\boldsymbol{p}_a = \mathrm{softmax}\left(\boldsymbol{p}^T \boldsymbol{W}_a + \boldsymbol{b}_a\right) \tag{7}$$

here $\boldsymbol{W}_a \in \mathbb{R}^{d \times n_a}$ is a weight matrix and $b_a \in \mathbb{R}^{n_a}$ is a bias vector, n_a is the number of action label and $\boldsymbol{p}_a \in \mathbb{R}^{n_a}$ is the action probability distribution for $\boldsymbol{p}$. We use cross entropy loss as the action prediction loss:

$$\mathcal{L}_{ap} = -\sum_{j=1}^{n_a} \left(\boldsymbol{l}_a^j \log\left(\boldsymbol{p}_a^j\right)\right) \tag{8}$$

where $\boldsymbol{l}_a$ is the label vector of action.

Similarly, we get the loss $\mathcal{L}_{op}$ for object label. We then combine these losses to obtain the following loss:

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} (\mathcal{L}_{ap} + \mathcal{L}_{op} + \alpha(\mathcal{L}_a + \mathcal{L}_o)) \tag{9}$$

where N is the number of training pairs. The parameters are updated by minimizing this loss, and the full algorithm is summarized in Algorithm 1.

Label Type	Action			Object		
Metrics	MRR	recall@1	recall@10	MRR	recall@1	recall@10
Sequence-to-label(mean)	3.72	1.96	5.95	1.01	0.80	1.66
Sequence-to-label(BiGRU)	7.94	4.40	12.71	4.20	2.72	6.19
Sequence-to-label(RoBERTa)	8.36	5.31	14.69	4.88	3.24	8.10
Single MFS-P2G (partial)	18.03	14.36	17.16	10.36	6.37	17.64
Single WSD-P2G (partital)	18.07	14.05	17.82	10.72	6.68	18.03
Single MFS-P2G	24.10	19.67	32.40	13.71	8.86	23.09
Single WSD-P2G	25.83	19.93	37.50	14.19	9.32	24.84
Joint MFS-P2G	28.57	20.63	43.14	15.26	10.62	25.01
Joint WSD-P2G	29.11	21.21	42.84	15.70	11.07	25.51
Model w/o hierarchy	36.80	28.34	53.71	11.79	7.66	19.25
HOT	**41.44**	**31.94**	**60.02**	**21.19**	**15.43**	**32.23**

Table 1: The action label and object label prediction results. The Model w/o hierarchy does not use the process hierarchy and label hierarchy compared with HOT.

Experiment

Experimental Setup

Dataset We use dataset released by Chen et al. (2020), in which each event process with one action and one object label is extracted from wikiHow[1]. It's an online wiki-style publication featuring expert co-authored, how-to articles on a variety of topics and each article consists of a title of how to do something and a series of sentences describing the steps. The events are extracted from these sentences by leveraging AllenNLP (Gardner et al. 2018) to perform SRL to extract the VERB term (action of an event) and ARG1 term (object of an event). These extracted events are then organized into an event process in the order of the steps. To extract action label and object label of the event process, SRL is also used for the title of the article to obtain the VERB and the head word of ARG1, which are chosen as action label and object label respectively. Finally, 62,277 event processes with 1,336 action types and 10,441 object types are extracted by the above procedure. The split of the training/dev./test set is 80/10/10%, and we report three ranking metrics, i.e. MRR (mean reciprocal rank), recall@1 and recall@10 following Chen et al. (2020).

Parameter Settings The RoBERTa-base pretrained model from Hugging-Face's Transformers library (Wolf et al. 2020) is used as the base encoder which is the same as previous work. The cosine function is chosen as the cost function in the hierarchical optimal transport and the hyper-parameter α is set to 0.1. We set the number of labels in label hierarchy to 2. Adam (Kingma and Ba 2015) is used for optimization with an initial learning rate 1e-4. The models are trained for 50 epochs with batch size of 64.

Baselines We compare our model with the following baseline methods for event process typing:

Sequence-to-label generators (S2L) (Rashkin et al. 2018): the model is based on encoder-decoder architecture and map sequences to unigrams of the type vocabulary directly. Three variants of S2L using different encoders are employed: Sequence-to-label(mean), Sequence-to-label(BiGRU) and Sequence-to-label(RoBERTa), which use mean-pooling of Skip-Gram word embeddings, BiGRU, and RoBERTa as encoders respectively.

Process-to-gloss based typing (P2G) (Chen et al. 2020): the model is based on pre-trained language model RoBERTa and tag the process with labels by directly matching the process level representation with the label gloss representation. We adopt some of its variants as our baseline models including separately learning for the two type axes, instead of performing joint training (Single or Joint), different gloss selection strategy (Pre-trained WSD models or Most frequent senses, denoted as WSD or MFS), and representing event using only a_i or o_i (Partial event).

Comparison with Baseline Methods

Experimental results are shown in Table 1, from which we can observe that:

(1) Our HOT outperforms all the baseline models in all metrics in terms of action label and object label prediction. In particular, it outperforms the Model w/o hierarchy which does not use the process hierarchy and label hierarchy. This shows the effectiveness of matching hierarchy.

(2) For our model and all baseline models, all the performance on object prediction is lower than the performance on action label prediction. The possible reason is that the object label space is larger than the action label space, which increases the difficulty of object prediction.

(3) Although the pre-trained language model RoBERTa is used as the encoder, the performance of the S2L model is still poor. It illustrates the necessity of considering more semantics of the label, so our method takes into account the label hierarchy to enrich the label semantics.

(4) As can be seen from the table, the performance of predicting the label is relatively poor if only the action or object of the event is used to represent the event in the process (partial event). This shows that extracting richer semantics of event process is helpful for predicting labels, just as our method considers the process hierarchy.

[1]https://www.wikihow.com/

Label Type	Action			Object		
Metrics	MRR	recall@1	recall@10	MRR	recall@1	recall@10
single-label-def	41.06	31.60	60.05	20.91	15.21	31.76
words-in-hierarchical-label	41.00	31.42	59.68	20.88	15.25	31.66
names-of-hypernyms	40.27	30.75	59.02	20.90	15.03	31.57
HOT	41.44	31.94	60.02	21.19	15.43	32.23

Table 2: Results on the impact of label hierarchy.

Label Type	Action			Object		
Metrics	MRR	recall@1	recall@10	MRR	recall@1	recall@10
events-only	40.64	31.79	57.88	20.48	15.06	31.17
words-only	40.80	31.41	59.06	20.19	14.60	30.56
HOT	41.44	31.94	60.02	21.19	15.43	32.23

Table 3: Results on the impact of process hierarchy.

Impact of Label Hierarchy

Recall that in our model, the hierarchical label for the action label a is a sequence $H_a = (a, f, a_1, f_1, ..., a_k, f_k)$, and the vector sequences $(\boldsymbol{h}_1^a, \boldsymbol{h}_2^a, ..., \boldsymbol{h}_{k+1}^a)$ corresponding to hypernyms are obtained to construct HOT loss. We now consider three variants of label representations to construct HOT loss: using the representation of the single label with its definition $\boldsymbol{h}_1^a$, using all hidden states of words in the hierarchical label sequence H_a, and using all hidden states of words in the sequence $(a, a_1, ..., a_k)$. For the object label, we use three variants similar to the action label. The above three variants are called *single-label-def*, *words-in-hierarchical-label* and *names-of-hypernyms* respectively. Table 2 shows the results and we can observe that:

(1) The HOT method performs better than the *single-label-def* method on the whole, which shows that considering the label hierarchy information is helpful for the predictions of action and object labels.

(2) Compared with *names-of-hypernyms*, HOT performs better on all metrics. The reason is that the HOT also considers the definition of each label word in the knowledge base, thereby introducing more information to enrich the semantics of the label.

(3) Compared with *words-in-hierarchical-label*, HOT also performs better. The possible reason is that although the labels of *words-in-hierarchical-label* contain similar information to labels in HOT, the labels of this method are matched based on words, and the semantics of each label in the hierarchical label is not explicitly considered.

Impact of Process Hierarchy

In our HOT method, both the sequence of events and the sequence of words in the event process are used to take into account the process hierarchy. Here we only use the sequence of events or the sequence of words in the event process, which are called *events-only* and *words-only* respectively. Results are shown in Table 3, from which we can make the following observations:

(1) Only using events or words in the event process can also achieve good performance, because they can provide some clues for the action label and object label prediction.

(2) When considering process hierarchy, that is, when combining event and word sequences in the process, better performance than *events-only* and *words-only* can be obtained. This also validates our motivation: both events and words can provide clues for label prediction, and combining the two can further improve the prediction performance.

Impact of Different Event Semantics Extraction

In our method HOT, mean-pooling of hidden states of words in each event is leveraged to get the representations of events in the process. Here we try to extract event semantics in other ways. One is to use the sum of the representation of each word in the event as the representation of the event, one is to take the maximum value of each dimension of the representation of all words in the event as the event representation, and one is to use the attention mechanism to obtain the representation of the event for the representation of the word in the event. The above three methods are called *sum*, *max* and *att* respectively. Table 4 shows the results and we can observe that:

(1) The HOT method which uses mean-pooling performs better than *sum* and *max*. The possible reason is that compared with *sum*, the mean-pooling can eliminate the influence of sequence length and obtain a more stable event representation. Compared with *max*, the mean-pooling can more comprehensively consider the semantic information of each word in the event.

(2) The HOT method is also better than *att*, which shows that in our task, using the mean-pooling method to extract the semantics of the event is more effective than using the attention mechanism. The possible reason is that there are already complex aggregation functions such as attention in RoBERTa, so it is enough to use mean-pooling as the aggregation function of the RoBERTa output.

Impact of Number of Labels in Label Hierarchy

We now investigate the impact of number of labels in label hierarchy, by changing the number and observe its effect on the model. The results are shown in Table 5 and we can observe that:

Label Type	Action			Object		
Metrics	MRR	recall@1	recall@10	MRR	recall@1	recall@10
sum	40.11	31.04	58.45	19.56	13.99	30.11
max	40.80	31.57	58.94	20.24	14.31	30.89
att	40.74	31.50	59.34	20.41	14.68	31.31
HOT	41.44	31.94	60.02	21.19	15.43	32.23

Table 4: Results on the impact of different event semantics extraction.

Label Type	Action			Object		
Metrics	MRR	recall@1	recall@10	MRR	recall@1	recall@10
number=1	41.06	31.60	60.05	20.91	15.21	31.76
number=2	41.44	31.94	60.02	21.19	15.43	32.23
number=3	40.55	31.54	58.30	19.50	13.92	30.17

Table 5: Results on the impact of number of labels in label hierarchy.

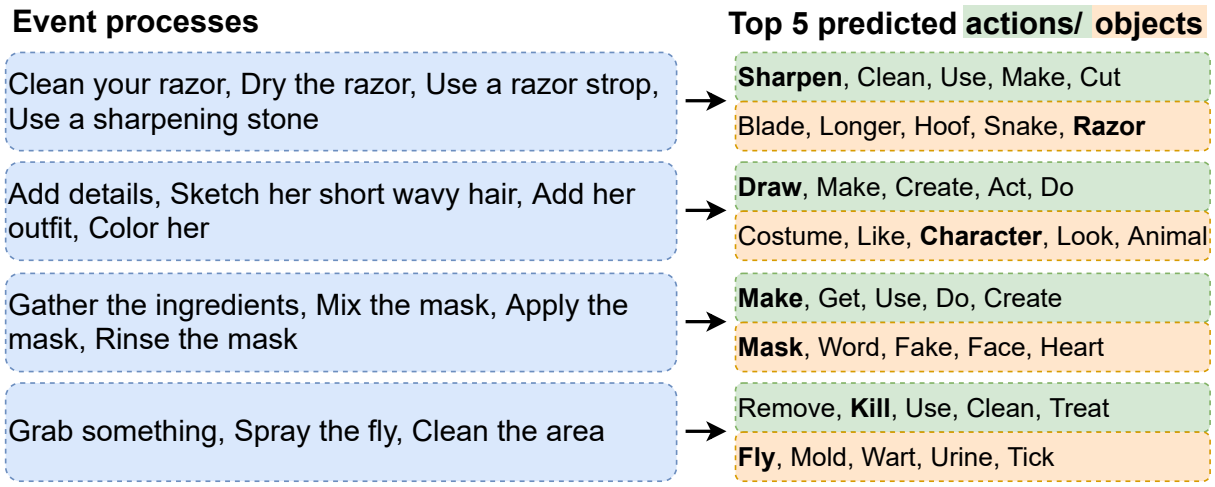

Figure 4: Top 5 predictions of action label and object label by our model HOT on examples of test data. Ground truths are in bold.

(1) When the number is equal to 2, the performance of the model is generally better than that when the number is equal to 1, that is, when the number of labels in the label hierarchy is appropriately increased, the performance of the model can be improved. The reason is that more label information is introduced, which further illustrates the effect of considering the label hierarchy.

(2) When the number continues to increase, the performance of the model begins to deteriorate. The reason may be that as the label hierarchy increases, the parent categories of some labels of different categories will tend to be the same, which affects label prediction and model performance.

Case Study

We conduct a case study using examples of test data. Figure 4 shows 4 event processes, and for each process our model HOT predicts the top 5 action and object labels. Although it is difficult to always rank the ground-truth labels on the top, the model can often infer similar labels to ground-truth as top predictions. For example in Figure 4, the ground-truth action label and object label of the first event process are *sharpen* and *razor* respectively. Although the top 1 prediction of object label by our model is *Blade*, it's similar to the ground-truth object label *Razor*. A similar situation exists for the other examples in Figure 4.

Conclusion

In this paper, we consider two types of hierarchy: process hierarchy and label hierarchy to help the event process typing task. For the former, semantics of each event in the process and semantics of each word in event are extracted, for the latter, hierarchical label for each label are extracted from external knowledge base. Then we propose a novel method Hierarchical Optimal Transport to deal with event process typing, which optimally match label semantics with event semantics and word semantics hierarchically. Experimental results demonstrate that our proposed model outperforms baseline methods.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No. 2020AAA0106400), the National Natural Science Foundation of China (No. 61922085, 61976211, 62176257). This work is also supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant No. XDA27020200), the Youth Innovation Promotion Association CAS, and Yunnan Provincial Major Science and Technology Special Plan Projects (No.202202AD080004).

References

Alqahtani, S.; Lalwani, G.; Zhang, Y.; Romeo, S.; and Mansour, S. 2021. Using Optimal Transport as Alignment Objective for fine-tuning Multilingual Contextualized Embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 3904–3919. Punta Cana, Dominican Republic: Association for Computational Linguistics.

Bai, L.; Guan, S.; Guo, J.; Li, Z.; Jin, X.; and Cheng, X. 2021. Integrating Deep Event-Level and Script-Level Information for Script Event Prediction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 9869–9878.

Chambers, N.; and Jurafsky, D. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*, 789–797.

Chen, M.; Zhang, H.; Wang, H.; and Roth, D. 2020. What Are You Trying to Do? Semantic Typing of Event Processes. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, 531–542.

Gardner, M.; Grus, J.; Neumann, M.; Tafjord, O.; Dasigi, P.; Liu, N. F.; Peters, M.; Schmitz, M.; and Zettlemoyer, L. 2018. AllenNLP: A Deep Semantic Natural Language Processing Platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, 1–6.

Granroth-Wilding, M.; and Clark, S. 2016. What happens next? event prediction using a compositional neural network model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Huang, Q.; Mo, L.; Li, P.; Cai, Y.; Liu, Q.; Wei, J.; Li, Q.; and Leung, H.-f. 2021. Story ending generation with multi-level graph convolutional networks over dependency trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 13073–13081.

Jans, B.; Bethard, S.; Vulić, I.; and Moens, M. F. 2012. Skip n-grams and ranking functions for predicting script events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 336–344.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.

Li, F.; Peng, W.; Chen, Y.; Wang, Q.; Pan, L.; Lyu, Y.; and Zhu, Y. 2020a. Event extraction as multi-turn question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, 829–838.

Li, J.; Li, C.; Wang, G.; Fu, H.; Lin, Y.; Chen, L.; Zhang, Y.; Tao, C.; Zhang, R.; Wang, W.; et al. 2020b. Improving text generation with student-forcing optimal transport. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 9144–9156.

Li, M.; Ma, T.; Yu, M.; Wu, L.; Gao, T.; Ji, H.; and McKeown, K. 2021. Timeline summarization based on event graph compression via time-aware optimal transport. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 6443–6456.

Li, M.; Xu, R.; Wang, S.; Zhou, L.; Lin, X.; Zhu, C.; Zeng, M.; Ji, H.; and Chang, S.-F. 2022a. Clip-event: Connecting text and images with event structures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16420–16429.

Li, Q.; Li, P.; Bi, W.; Ren, Z.; Lai, Y.; and Kong, L. 2022b. Event Transition Planning for Open-ended Text Generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, 3412–3426.

Li, Q.; and Zhang, Q. 2020. A Unified Model for Financial Event Classification, Detection and Summarization. In *IJCAI*, 4668–4674.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Luise, G.; Rudi, A.; Pontil, M.; and Ciliberto, C. 2018. Differential Properties of Sinkhorn Approximation for Learning with Wasserstein Distance. *Advances in Neural Information Processing Systems*, 31: 5859–5870.

Lv, S.; Zhu, F.; and Hu, S. 2020. Integrating External Event Knowledge for Script Learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, 306–315.

Miller, G. A. 1995. WordNet: a lexical database for English. *Communications of the ACM*, 38(11): 39–41.

Nouri, N. 2022. Text Style Transfer via Optimal Transport. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2532–2541. Seattle, United States: Association for Computational Linguistics.

Peyré, G.; Cuturi, M.; et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6): 355–607.

Pichotta, K.; and Mooney, R. 2016. Learning statistical scripts with LSTM recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Rashkin, H.; Sap, M.; Allaway, E.; Smith, N. A.; and Choi, Y. 2018. Event2Mind: Commonsense Inference on Events, Intents, and Reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 463–473.

Veyseh, A. P. B.; Van Nguyen, M.; Dernoncourt, F.; Min, B.; and Nguyen, T. 2022. Document-Level Event Argument Extraction via Optimal Transport. In *Findings of the Association for Computational Linguistics: ACL 2022*, 1648–1658.

Wang, S.; Cai, X.; Wang, H.; and Yuan, X. 2021. Incorporating Circumstances into Narrative Event Prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 4840–4849.

Wolf, T.; Chaumond, J.; Debut, L.; Sanh, V.; Delangue, C.; Moi, A.; Cistac, P.; Funtowicz, M.; Davison, J.; Shleifer, S.; et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45.

Xie, Y.; Wang, X.; Wang, R.; and Zha, H. 2020. A fast proximal point method for computing exact wasserstein distance. In *Uncertainty in Artificial Intelligence*, 433–453. PMLR.

Zhang, H.; Chen, M.; Wang, H.; Song, Y.; and Roth, D. 2020. Analogous Process Structure Induction for Sub-event Sequence Prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1541–1550.

Zhou, B.; Chen, Y.; Liu, K.; Zhao, J.; Xu, J.; Jiang, X.; and Li, J. 2021. Multi-Task Self-Supervised Learning for Script Event Prediction. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 3662–3666.