Similarity Measurement Between Event Sequences with Penalty Adjustment

Daisuke Machizawa
Department of Computer Science
Takushoku University
Tokyo, Japan
25m312@st.takushoku-u.ac.jp

Naoki Sawahata
Department of Computer Science
Takushoku University
Tokyo, Japan
sumikawa.lab@gmail.com

Yasunobu Sumikawa
Department of Computer Science
Takushoku University
Tokyo, Japan
ysumikaw@cs.takushoku-u.ac.jp

***Abstract*—Analyzing event sequences offers the advantage of connecting the past and present, providing a foundation for applying historical knowledge to present contexts. To facilitate such connections for different types of event sequences, developing a search algorithm that takes an event sequence as input and identifies other similar sequences is essential. In this research, we propose PASS, an algorithm for measuring the similarity between event sequences. PASS increases the similarity score when similar events occur in the same order and reduces the score in the presence of dissimilar events or mismatches in the number of events. PASS considers two events similar if their similarity score, as determined by any text similarity measurement method, exceeds a predefined threshold. Conversely, if the score falls below this threshold, the events are regarded as dissimilar. To validate the effectiveness of our proposed algorithm, we conducted experiments using an event sequence dataset extracted from the Wikipedia Current Events portal. We compared PASS with two type baseline approaches: one that considers the entire event sequence as a single text and another that computes the similarity between event sequences. The accuracy of each algorithm was evaluated using the mean squared error, and the results demonstrated that the proposed algorithm outperformed the baselines.**

***Index Terms*—Event sequence, similarity measurement, timeline, dynamic programming, information retrieval**

I. Introduction

The adage *History does not repeat itself, but it rhymes* highlights the recurrence of analogous event patterns across different time periods. The analysis of such recurring phenomena is widely acknowledged for its importance, primarily in enabling the application of historical knowledge to the formulation of strategies for contemporary and future challenges. Furthermore, this approach enhances the comprehension of the developmental processes that have shaped modern society. To conduct such analyses effectively, it is particularly important not to focus solely on individual events but to consider event sequences, which are often referred to as causal relationships or timelines and include the contexts and consequences of those events. Indeed, emphasizing causal relationships that encompass multiple events is regarded as a key element in developing the ability to identify the differences between past and present contexts, which is a skill that is prioritized in many countries [1], [2].

The analysis of diverse types of event sequences necessitates substantial datasets comprising such sequences. Although the modern Web contains an extensive amount of event data, most consist of individual events recorded in isolation. Thus, the following two steps must be implemented to analyze diverse event sequences: 1) construct appropriate event sequences from data available on the Web, and 2) search for sequences similar to a given input event sequence from the constructed dataset. The first step has long been the focus of various methods, such as Topic Detection and Tracking (TDT) [3], [4] and Timeline Summarization (TLS) [5]. The second step requires a search algorithm that considers multiple events as both the input and output. One such approach is the Event Causality relationship similarity Measurement (ECM) [6], which extends maximum weight matching to measure the similarity between event sequences. ECM focuses on identifying the combination of events that maximizes the overall similarity score between them. Thus, it disregards dissimilar events that are considered as noise within event sequences, potentially leading to inaccurate similarity evaluations.

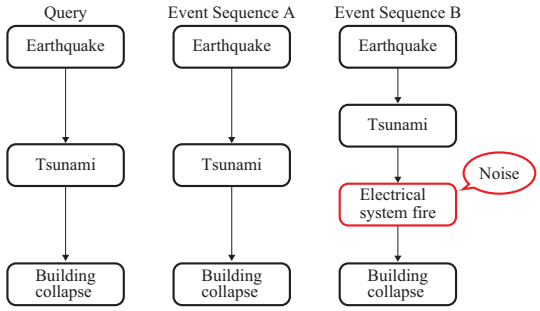

Fig. 1. A motivation example

In this study, we propose an algorithm named penalty-adjusted similarity for event sequences (PASS), which evaluates the similarity between event sequences arranged in chronological order. Unlike existing algorithms, PASS decreases the similarity score when dissimilar events are included in the sequences. PASS uses event sequences as the input and computes the similarity between the constituent events of the input sequence and those of the candidate output sequences. During this process, PASS identifies pairs of similar events that occur in the same order. If one sequence contains events without corresponding similar events in the other sequence,

PASS inserts dummy events, which are referred to as "gaps," into the other sequence and assigns a negative similarity score. To achieve this, the proposed algorithm extends the Needleman-Wunsch (NW) algorithm [7], which is a widely used pairwise alignment technique for measuring the similarity between two nucleotide sequences, to compute the similarity between event sequences.

Fig. 1 depicts an example of event sequences in which PASS inserts gaps. Although PASS operates on textual data, this example only displays the titles of events for simplicity and clarity. When comparing the query event sequence with event sequences `A` and `B`, it can be observed that both share three common events: `Earthquake`, `Tsunami`, and `Building collapse`. However, sequence `B` includes an additional event, `Electrical system fire`, which is not present in the query sequence. Because of this extra event in sequence `B`, PASS interprets the gap as occurring between `Tsunami` and `Building collapse` in the query sequence and assigns a gap penalty. Consequently, the similarity score for sequence `B` is reduced compared with that of sequence `A`.

To evaluate the effectiveness of PASS, we used an event sequence dataset constructed from Wikipedia. The performance was assessed based on two metrics: the reduction of noise and similarity scores in the top-k search results (k=10, 50, and 100) measured using the Mean Squared Error (MSE) and precision, respectively. We compared PASS with two types of baselines: one that considers an event sequence as a single document and another that evaluates the similarity between event sequences by distinguishing individual events. The results showed that PASS achieved the best MSE scores across all values of k, demonstrating its ability to minimize noise effectively. However, the precision scores indicate that considering event sequences as a single document yielded the best results. These findings suggest that PASS provides the highest overall similarity for the input event sequence. However, further improvements are necessary to handle cases in which one event sequence partially matches another.

II. Related Work

Previous studies focusing on events can be categorized into two main types: those that identify relationships between events, as exemplified by TDT and TLS, and those that retrieve events similar to those input by users. The following subsections discuss these two categories in detail.

A. Identifying Relationships between Events

Research on automatically analyzing the relationships between events reported in the news and linking related events has been conducted for several years. Studies that analyze the topics to which individual events belong and connect events within the same topic into a single chain are known as TDT [3], [4], [8]. Research that has expanded upon the study of TDT includes investigations to clarify whether it is possible to make future predictions by utilizing its outcomes [9], as well as studies on TLS that extract and generate only significant sentences rather than merely listing events [5]. Other studies have represented relationships using tree structures [10], [11] or graphs [12], [13] instead of linear representations.

All of the aforementioned studies are useful when structuring and accumulating data, or when understanding the content of a single sequence of events. Therefore, these previous studies are orthogonal to the present study and their results may be useful for this study.

B. Event Retrieval

Research aimed at event retrieval, similar to this study, has also been conducted. Many previous methods focused on retrieving individual events. Examples include a method that outputs sentences in other languages describing the content of an event when the event is input as a sentence [14] and a technique in which an image is used as a query to retrieve information regarding where the event depicted in the image occurred, thereby facilitating news retrieval based on this result [15]. Research has been conducted on retrieving multiple events with a particlar focus on messages posted on microblogs [16]. This prior study, similar to others on event retrieval, assumed a single event as the query.

The method most closely related to the present study is ECM, which outputs similar event sequences when multiple events are input. ECM solves the maximum weight matching problem with adding a constraint that ensures that similar events occur in the same order for calculating the similarity between the input and candidate event sequences. Because the maximum weight matching seeks the set of edges with the greatest weight on a bipartite graph, it effectively ignores noise. Although a method that extends ECM to evaluate the similarity between past and present event sequences has been proposed [17], the issue of ignoring noise, which is a limitation of ECM, remains an unresolved problem in existing research.

In this study, to address the noise present in event sequences for which ECM cannot account, we based our approach on the alignment technique rather than maximum weight matching, allowing for a reduction in the similarity between event sequences that contain noise.

III. Needleman-Wunsch algorithm

The NW algorithm is an algorithm that measures the similarity between two sequences, $X = \{x_1, x_2, ..., x_m\}$ and $Y = \{y_1, y_2, ..., y_n\}$, whose elements belong to an alphabet, while aligning them to maximize the number of common parts.

To calculate the similarity score, NW algorithm defines a score matrix S of size $m \times n$, where each element corresponds to one of three possibilities: match, mismatch, or gap. A match occurs when two compared elements are exactly identical. A gap occurs when one of the elements is shifted to align with the element of the other sequence, whereas a mismatch occurs when a gap is inserted but the two elements still do not match. For any pair of elements x_i and y_j, NW algorithm determines whether the elements constitute a match, mismatch, or gap. If they are a match or mismatch, the algorithm computes the score by adding the appropriate score to the value of the (i-1, j-1) cell in the score matrix S. This value is then

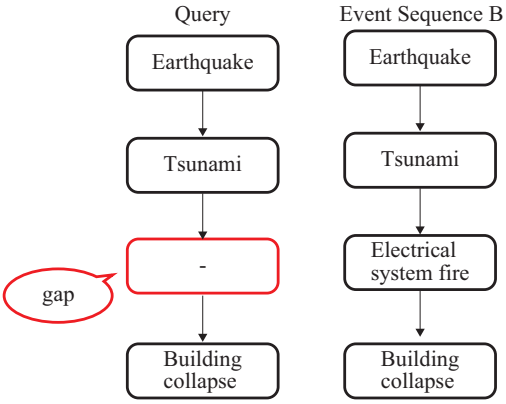

Fig. 2. An example of alignment

compared with the scores obtained by adding the gap penalty to the values of the (i-1, j) and (i, j-1) cells, respectively. Subsequently, the maximum of these values is assigned to the (i, j) element of S. The calculation process is repeated for all elements of the matrix. The similarity score between the two sequences is recorded in the (m, n) cell of the matrix. The optimal alignment path can be extracted by performing a traceback process in reverse order, showing how the elements of the two sequences match most similarly.

IV. PASS Algorithm

PASS treats an event sequence as an array of event sentences, and extends NW algorithm to compute the similarity between event sequences. In this section, we first describe the preprocessing required to calculate the similarity between sentences, and then provide a detailed explanation of how we extend NW algorithm to compute the similarity between event sequences.

A. Preprocessing

We assume that each event is described in natural language; thus, preprocessing steps such as stopword removal, stemming, and subword extraction are required to have been carried out in advance. As PASS is designed as a language-independent algorithm, language-specific preprocessing steps are assumed to be appropriately executed at this stage. For example, if the event sentences are written in languages with case distinctions, such as English, they will be converted into lowercase sentences. If the sentences are in languages such as Japanese or Chinese, where word boundaries are not explicitly marked, morphological analysis will be performed.

In the remainder of this section, we assume that the features of the event sentences have been transformed into bag-of-words (BoWs) representations after undergoing the aforementioned preprocessing.

B. Similarity Evaluation Algorithm

PASS calculates the similarity between arrays that are assumed to consist of sentences as elements to determine the match, mismatch, and gap. In the following section, we

		Earthquake	Tsunami	Electrical system fire	Building collapse
	0	-2	-4	-6	-8
Earthquake	-2	1	-1	-3	-5
Tsunami	-4	-1	2	0	-2
Building collapse	-6	-3	0	1	1

Fig. 3. Tables used in dynamic programming solver

explain the proposed algorithm using two event sequences es_A and es_B, which are defined as $es_A = \{a_1, a_2, \ldots, a_m\}$ and $es_B = \{b_1, b_2, \ldots, b_n\}$.

Fig. 2 shows the gaps that PASS inserts into the query event sequence in response to event sequence es_B, which contains events that should have their similarity reduced, as shown in Fig. 1. In this example, the first two and last events are identical in the two event sequences. However, the event sequence es_B contains one more event than the query event sequence. To highlight this difference in numbers clearly, a gap representing an empty event is inserted at the third position of the query event sequence. This insertion indicates that the only difference between the two event sequences is the additional event in es_B. PASS uses the table shown in Fig. 3 to calculate the similarity between the two event sequences. Specifically, when comparing the elements of two event sequences, the algorithm must determine whether the two elements are a match or mismatch, or if a gap should be inserted into one of the sequences. In this example, we assign 1, -1, and -2 as the match score, mismatch score, and gap penalty, respectively. During this process, the scores for the cells in the table are calculated using the values in the top-left, left, and top cells corresponding to the current comparison. The first row and column of the table are initialized with the values resulting from inserting gaps at the beginning of each event sequence, following the respective sequence order. The process begins by comparing the first events of the two event sequences. In this case, both events are labeled as `Earthquake`; thus, a match score of 1 is added to the $(1, 1)$ cell. This score is greater than the value obtained by inserting gaps where the calculation is -2 added to the $(0, 1)$ or $(1, 0)$ cells; thus, the value 1 is placed in the $(1, 1)$ cell. The remaining cells are computed in a similar manner. For example, the $(3, 2)$ cell, which represents the insertion of a gap into the query event sequence, results in a score that is 2 smaller than the $(2, 2)$ cell. This value is larger than that of the alternatives obtained by adding the mismatch score to the $(2, 1)$ cell or the gap penalty to the $(3, 1)$ cell. Thus, this calculation appropriately indicates the insertion of a gap at the third position of the query event sequence. Through these steps, the table effectively captures the decision-making process for gap insertion and aligning event sequences, while accounting for differences in the number of events.

As explained in the above example, PASS computes the similarity score between event sequences using a table DP. The values in the first row and first column of the DP are

initialized using the values shown in the following equation:

$$\begin{cases} DP_{0,0} = 0 \\ DP_{i,0} = gap \times i \quad (1 \leq i \leq m) \\ DP_{0,j} = gap \times j \quad (1 \leq j \leq n) \end{cases} \tag{1}$$

where gap is a hyperparameter representing gap insertion. This initialization step sets the scores for the first row and first column based on the assumption that gaps are inserted into the event sequences up to that point.

The remaining cells of the table DP are computed using the following equation:

$$DP_{i,j} \overset{def}{\Leftrightarrow} \max \begin{cases} DP_{i-1,j-1} + s(a_i, b_j) \\ DP_{i-1,j} - gap \\ DP_{i,j-1} - gap \end{cases} \tag{2}$$

The function s determines whether the two elements a_i and b_j that are provided as arguments are a match, mismatch, or gap, and returns the corresponding value. This function is defined as follows:

$$s(a_i, b_j) \quad \overset{def}{\Leftrightarrow} \quad \begin{cases} m\_val & \text{if } sim(a_i, b_j) \geq thre \\ mis\_val & \text{otherwise} \end{cases} \tag{3}$$

The function sim computes the similarity between two sentences by applying similarity measures such as Jaccard coefficient or cosine similarity. Within this function, necessary preprocessing steps such as conversion from BoW to TF-IDF and applying topic modeling (e.g., latent Dirichlet allocation) are performed as required. The parameters m_val and mis_val are hyperparameters that specify the scores for matches and mismatches, respectively. In the proposed algorithm, if the sentence similarity exceeds a certain threshold $thre$, the two sentences are considered a match; otherwise, they are treated as a mismatch. The three hyperparameters m_val, mis_val, and $thre$ depend on the sentence similarity measurement and should be tuned experimentally based on the selected method.

Finally, PASS outputs the similarity score in the (m, n) cell of the table. This score reflects the optimal alignment between the two event sequences after considering both matches and mismatches, along with any gaps.

Algorithm 1 presents the pseudocode for the similarity measurement between two event sequences using PASS. First, the table DP is initialized. The functions $ColumnSize$ (line 2) and $RowSize$ (line 3) return the numbers of columns and rows in the table, respectively. These values correspond to the lengths of the two event sequences, es_A and es_B, that are provided as arguments. Lines 5∼8 initialize the first row and column of DP using the gap penalty; these are equal to the Equation 1. Lines 10∼17 are the main processes of PASS; these lines perform the calculation described in Equation 2. Initially, the algorithm computes the value if the elements are a match or mismatch (line 12). It then computes the value by adding the gap penalty gap to the cell directly above the current cell (line 13) and to the left side of the current cell (line 14). Subsequently, the function max (line 15) is applied to select the maximum value among the three calculated values.

Algorithm 1 PASS algorithm

Input: Event sequence A: es_A, event sequence B: es_B: a table: DP

Output: a similarity score

1: **Function** f (es_A, TL_b, DP)
2: $c \leftarrow ColumnSize(DP)$ // = length of es_A
3: $r \leftarrow RowSize(DP)$ // = length of es_B
4: // Table initialization
5: **for** $i = 1$ to c
6: $DP_{0,i} \leftarrow DP_{0,i-1} + gap$
7: **for** $i = 1$ to r
8: $DP_{i,0} \leftarrow DP_{i-1,0} + gap$
9: // Calculation of each cell
10: **for** $i = 1$ to c
11: **for** $j = 1$ to r
12: $up\_left \leftarrow DP_{i-1,j-1} + s(es_{A_i}, es_{B_j})$
13: $up \leftarrow DP_{i,j-1} + gap$
14: $left \leftarrow DP_{i-1,j} + gap$
15: $DP_{i,j} \leftarrow max(up, up\_left, left)$
16: **end for**
17: **end for**
18: **return** $DP_{r,c}$

Once all cells have been calculated, the algorithm returns the value at the bottom-right corner of the table $DP[m][n]$, which represents the similarity between es_A and es_B (line 18).

V. Experimental Evaluation

A. Evaluation Metrics

PASS aims to reduce the similarity score for inconsistencies between event sequences appropriately. Therefore, in this experiment, we evaluated the accuracy of each method using MSE, which calculates the mean squared difference between the predicted and true values. Lower MSE values indicate better prediction accuracy. In addition, we examined the precision to determine the number of relevant results. Higher precision indicates a greater proportion of relevant results. As a practical usage of PASS as a search engine, each method in this evaluation presents the results in the form of the top k ranked items. We analyzed how the two metrics, MSE and precision, changed across different values of k. Specifically, we report MSE@k and p@k for values of k=10, 50, and 100.

B. Research Questions

We conducted the experiments based on the following research questions (**RQs**):

- **RQ1**: Which algorithm outputs event sequences exhibiting the fewest dissimilarities relative to the input query sequence?
- **RQ2**: Which algorithm is most effective at retrieving event sequences that demonstrate high relevance to the input query sequence?
- **RQ3**: Did the lengths of the top-ranked event sequences in the output exhibit a trend similar to that of the query?

First, we assessed whether PASS could output event sequences with minimal noise using the MSE@k values by evaluating **RQ1**. Next, in **RQ2**, we verified whether PASS could retrieve event texts similar to queried texts using precision@k. Finally, for **RQ3**, an error analysis was performed on the event sequences output using the top-scored baseline and PASS.

C. Data Collection

We designed PASS as a language- and dataset-independent algorithm and evaluated its effectiveness using various types of event sequences. To achieve this goal, the dataset needs to possess two key characteristics: 1) It must contain a diverse range of event types and event sequences. 2) It must allow for the evaluation of similarity between individual events. While several datasets satisfying the first characteristic have been proposed, such as CStory [18] and W2E [19], they lack features enabling the second requirement. Fulfilling the second characteristic comprehensively would require considering all possible pairs of events, but manually evaluating similarity for every pair in a large-scale dataset incurs prohibitive costs. To achieve the characteristic effectively and efficiently, we employed the attention *similar events are assigned to the same category.* For example, the `2011 Tohoku earthquake and tsunami` and `2007 Peru earthquake` share the `Earthquake`, `Tsunami`, and `Natural disasters` categories. Based on this attention, we considered that *if two events share the same category, they are similar.* That is, we converted the components of the event sequence into the category of that event, and then calculated MSE and precision scores based on whether the categories were consistent.

Therefore, this study required not only a large number of events and event sequences, but also the categories to which each event is assigned. Because no large-scale event sequence dataset with these characteristics was available, we created a new dataset for this study.

1) Data resource: To create the new dataset, we used events recorded in the current portal of the English version of Wikipedia. Specifically, we extracted the events from 2016 to 2017. We also collected event categories such as `Politics and elections`, `Sports`, and `Disasters and accidents`, along with the corresponding event descriptions for evaluation.

2) Event sequence creation: We manually created event sequences for 2016 to 2017. This task was carried out by three workers. Initially, two workers independently reviewed all data for one year and created their respective event sequences. During this process, the workers created an event sequence that contained at least two events. Subsequently, a third worker reviewed all data and created event sequences in a similar manner. In cases where discrepancies arose between the event sequences of the two workers, they discussed the differences and modified the sequences accordingly.

Tab. I shows an example of the event sequences created in this study. The first column of the table indicates the categories assigned to each event by Wikipedia editors in the current events portal. The second column lists the dates of event occurrence, and the third column contains the event descriptions. The event sequence presented in this table comprised six events.

TABLE I
EXAMPLE OF AN EVENT SEQUENCE. `PE` IS AN ABBREVIATION FOR THE POLITICS AND ELECTIONS CATEGORY, AND `IR` IS AN ABBREVIATION FOR INTERNATIONAL RELATIONS.

Category	Date	Event description
`PE`	2016-06-10	In a substantial swing, the "Leave" camp is 10 points ahead of "Remain" with less than two weeks to go before Britain's referendum on whether to stay in the European Union, according to a poll by ORB.
`PE`	2016-06-14	Rupert Murdoch-controlled tabloid The Sun, Britain's second biggest selling newspaper, endorses Brexit, saying that the people of UK should "Take back control and BeLeave in Britain".
`PE`	2016-06-22	European Commission president Jean-Claude Juncker warns the British voters, saying that there would be no renegotiations with Britain whatever outcome of its membership referendum.
`PE`	2016-06-23	Voters in the United Kingdom go to the polls to vote in a referendum on whether the UK should leave the European Union.
`PE`	2016-06-24	The United Kingdom votes, 52% to 48%, to leave the European Union. Prime Minister David Cameron, who called the referendum three years ago, announces his resignation indicating he will leave office by October.
`IR`	2016-06-25	Diplomats from Germany, France, Italy, the Netherlands, Belgium, and Luxembourg meet in Berlin to discuss the consequences of the United Kingdom voting to leave the European Union.

3) Statistics of Dataset: As a result of the dataset creation process described in Section V-C2, we constructed 2,879 event sequences comprising a total of 5,204 events. Analysis of the length distribution of the constructed event sequences revealed that sequences of length 2 were the most frequent, and the number of sequences generally decreased as the length increased. The longest event sequence identified in the dataset contains 47 events.

D. Baselines

We employed the following methods as baselines:

- **Cos**: This is the cosine similarity, which considers two feature vectors as similar if the angle between them is small.
- **Jaccard**: This refers to the Jaccard coefficient, which determines the similarity between two texts based on the proportion of shared words.
- **BM25**: This method calculates the relevance between a query and document by considering the frequency of word occurrences and document length, and then ranks the documents accordingly.

In the above three methods, the event texts within a single event sequence are concatenated into one unified text. In this evaluation, for the Cos method, we considered the text of an event sequence as a single document and converted it into a feature vector using TF-IDF. We also used Cos and Jaccard in the function sim defined in Equation 3 of PASS. For clarity, we refer to the versions of PASS using these implementations as PASS_cos and PASS_jaccard, respectively.

In addition to the above methods, we compared PASS with the following existing methods, which calculate the similarity by handling each event as a separate document and considering its order:

- **Dynamic time warping (DTW)**: This method measures the similarity between two time-series datasets. We constructed feature vectors for each event text using TF-IDF, and the similarity between the data points was calculated using the Euclidean distance.
- **ECM**: This algorithm calculates the similarity between event sequences based on the maximum weight matching.

E. Hyperparameter setting

PASS uses four hyperparameters: $thre$, m_val, mis_val, and gap. $thre$ is the threshold used to determine whether two events match. m_val and mis_val are the scores when events are considered a match and mismatch, respectively. gap represents the gap penalty. To find the optimal combination of these parameters, we defined the following search spaces: $thre \in [0.00, 1.00]$, $m\_val \in [0.0, 10.0]$, $mis\_val \in [-10.0, 0.0]$, and $gap \in [-10.0, 0.0]$. We performed a grid search within these spaces to identify the optimal parameter combination. Each value of parameter was varied in increments of 0.01, starting from the minimum value in its respective range. The parameter combination that resulted in the minimum MSE value was selected as the optimal setting. As a result, we set the values of $thre$, m_val, mis_val, gap to 0.21, 0.9, -0.7, and -0.6, respectively, for Cos implementation. In addition, we set the values to 0.1, 0.6, -0.4, and -0.2 for Jaccard.

F. Results

RQ1: Which algorithm outputted event sequences exhibiting the fewest dissimilarities relative to the input query sequence?
A1. PASS achieved the best results, particularly when using the Jaccard coefficient to measure the similarity between event texts.
A2. Using separated event texts obtained better results than treating the event sequence as a single document.
A3. All implementations of PASS outperformed DTW and ECM, highlighting the importance of introducing penalties.

Tab. II presents the MSE scores obtained when applying all methods. Upon reviewing these results, it is evident that the proposed algorithm, especially, PASS_jaccard, consistently outperformed the baselines across all values of k. To better understand these results, we compared PASS and the baselines in detail.

TABLE II
MSE VALUES FOR ALL EVALUATED METHODS. THE BEST SCORE IS SHOWN IN BOLD.

Method	k=10	k=50	k=100
Cos	36.44	41.69	36.87
Jaccard	20.64	21.43	27.27
BM25	31.78	31.81	36.84
DTW	434.22	179.84	117.90
ECM	85.87	91.86	80.90
PASS_cos	27.07	26.83	26.23
PASS_jaccard	**12.27**	**19.50**	**20.36**

Cos, Jaccard, and BM25 vs. PASS. To apply the cosine similarity, Jaccard coefficient, and BM25, we concatenated all event texts within a single document and computed the similarity between the two resulting texts. The key difference between these methods and PASS is the reduced number of explicit events owing to concatenation. This result implies that when multiple events are combined into a single sentence, multiple stories are often mixed, and the effect of noise is relatively reduced. This emphasizes the importance of maintaining events separate in tasks involving event-sequence matching or retrieval.

DTW and ECM vs. PASS. All versions of PASS achieved better results than DTW and ECM for all k values, although these baselines define separate feature vectors for each individual event within an event sequence. In particular, compared with DTW, PASS_jaccard achieved approximately 83% to 97% better scores. These results suggest that reducing the similarity when dissimilar events are present leads to better outcomes.

Cos vs. Jaccard within PASS. Because two different implementations of sentence similarity were used in PASS, we compared the results of these implementations. PASS_jaccard consistently exhibited the best accuracy across all values of k. This outcome is likely owing to the characteristics of the dataset, which contain short event descriptions. These characteristics led to the cosine similarity implementation producing sparse feature vectors. However, the Jaccard coefficient focuses on the overlap of words between sentences, making it more effective for this particular dataset. Given the short length of the event descriptions, the emphasis of Jaccard coefficient on shared terms likely resulted in a better representation of similarity, yielding improved performance.

TABLE III
$p@k$ VALUES FOR ALL EVALUATED METHODS. THE BEST SCORE IS SHOWN IN BOLD.

Method	k=10	k=50	k=100
Cos	0.20	0.22	0.19
Jaccard	0.67	**0.59**	0.50
BM25	**0.69**	0.57	**0.53**
DTW	0.30	0.26	0.252
ECM	0.25	0.238	0.24
PASS_cos	0.11	0.14	0.15
PASS_jaccard	0.41	0.35	0.28

RQ2: Which algorithm retrieved event sequences exhibiting the highest degree of relevance to the input query sequence?
A1. BM25 and Jaccard coefficient achieved the highest $p@k$ results.
A2. PASS_jaccard performed better than both DTW and ECM.

Tab. III shows the $p@k$ scores for all of the methods. It can be observed that BM25 and Jaccard achieved the best performance for k=10 and 100 and k=50, respectively. These results suggest that combining events in an event sequence into a single document leads to better performance. One reason that PASS had worse results than the baselines is its current design. When comparing event sequences of different lengths, PASS inserts gaps to align their lengths. This introduces a penalty and reduces the similarity score. Consequently, even if partial matches exist in the query event sequence, PASS reduces the similarity score of event sequences with different lengths.

Fig. 4 shows an example of partial matching between the query event sequence shown in (a) and an event sequence from the dataset shown in (b). All of these events are related to earthquakes; thus, `earthquake` appears in many event texts. In addition, other common words, such as `struck` and `death`, appear in many of the event texts. The presence of these common terms contributed to the improved accuracy when using similarity measures such as the Jaccard coefficient and BM25. In contrast, the query and dataset event sequences contained five and three events, respectively. When applying PASS, two gaps were inserted to account for the difference in the number of events, regardless of content matching. This implies that even if partial matching exists, the ranking of the event sequence from the dataset is reduced. Thus, the presence of gaps reduces the overall ranking of event sequences with partial matches, reflecting the impact of sequence length differences in the proposed approach.

Comparing the results of DTW and ECM reveals that PASS_jaccard achieved better accuracy across all values of k. It is evident that reducing the similarity when event sequences contain dissimilar events in shorter sequence is desirable. These results suggest that, an important future challenge is to extend PASS to allow for the identification of partial matches by shifting the shorter event sequence to achieve better results

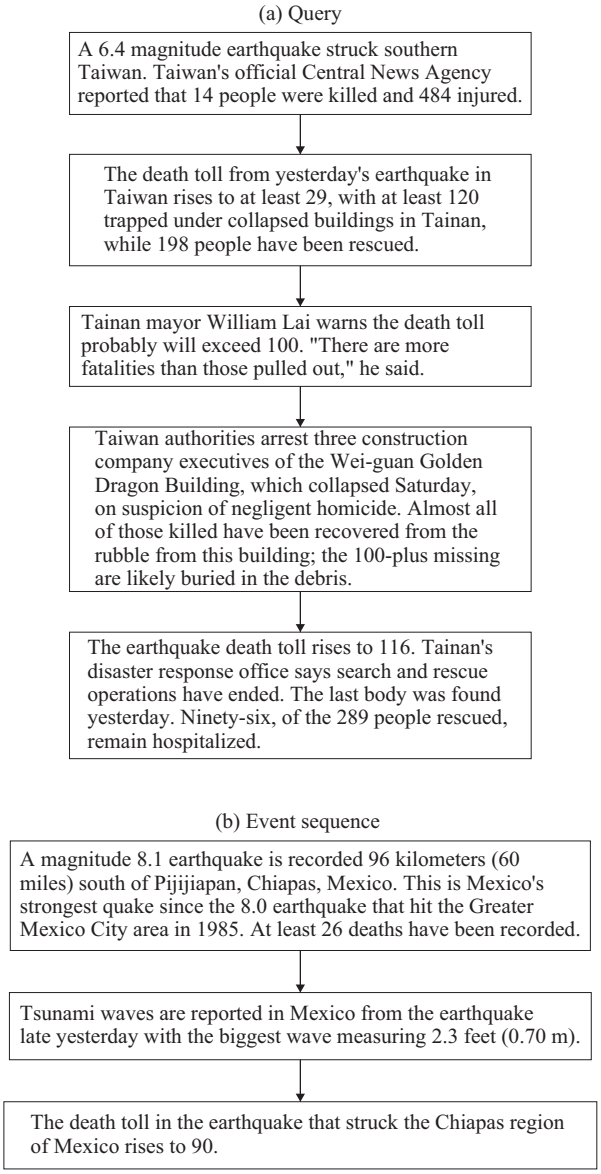

Fig. 4. Example of event sequences with partial matching

by comparing event sequences of different lengths. This extension could enable more effective searching and matching of event sequences with varying lengths.

RQ3: Did the lengths of the top-ranked output sequences tend to match the query length?
A. PASS often output event sequences of the same length as the input event sequence, whereas the baselines output longer event sequences if the number of output results increased.

Finally, we conducted a quantitative analysis to understand why PASS achieved improvements in MSE but showed a decline in $p@k$ compared to the baselines. As discussed in the analysis of **RQ2**, the current design of PASS tends to produce output sequences with lengths similar to those of the

input event sequences. As a result, the top-ranked outputs are typically expected to match the input length. To assess the effect of gap insertion in the proposed algorithm, we compared the different lengths of output generated by Jaccard, which performed well in both MSE and $p@k$ among the baselines, with those of PASS_jaccard. The results of Jaccard for k=1 and k=5 were 1.32 and 1.58, respectively. In contrast, the results of PASS_jaccard for k=1 and k=5 were 0.32 and 0.33, respectively. For both values of k, the results of applying PASS_jaccard showed that the difference in length between the input and output event sequences was approximately 0.3. These values were smaller than those of Jaccard. Furthermore, comparing the results for k=1 and k=5, PASS_jaccard only increased by approximately 3%, whereas Jaccard increased by approximately 20%.

From the above results, it can be confirmed that PASS tends to output event sequences that are similar to the input event sequence in terms of both content and length.

VI. Conclusions

This study has proposed PASS, an algorithm for calculating the similarity between two event sequences. PASS considers not only event-to-event similarity but also incorporates gaps representing null events. This mechanism penalizes discrepancies in event counts, consequently reducing the overall similarity score. Use of this method enables the search for event sequences containing numerous similar events while featuring few inappropriate noisy events.

To evaluate the effectiveness of this approach, experiments were conducted using an event sequence dataset constructed from Wikipedia events. Baseline methods implemented for comparison included widely used document similarity approaches like Jaccard coefficient and TF-IDF with cosine similarity, treating the entire event sequence as one document. Other baselines involved sequence similarity methods like DTW and ECM, which consider individual events as distinct units. Evaluation of these methods using MSE confirmed that the proposed method achieved the best results. Conversely, evaluations using precision at k precision@k revealed baseline methods such as Jaccard coefficient and BM25 achieved higher precision than the proposed method. These findings demonstrate that PASS effectively ranks event sequences higher when both the number of events and their content closely match the query sequence.

Future work involves extending PASS to accommodate searches for partially matching event sequences.

Acknowledgements. This work was supported in part by MEXT Grant-in-Aids (#25K15357)

References

[1] Ministry of Education, Culture, Sports, Science and Technology (MEXT), "Course of study for upper secondary schools (public notice of heisei 30)," 2018, [In Japanese]. [Online]. Available: https://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2019/11/22/1407073_03_2_2.pdf

[2] V. Boix-Mansilla, "Historical understanding: Beyond the past and into the present," *Knowing, teaching, and learning history: National and international perspectives*, pp. 390–418, 2000.

[3] A. J. Kurtz and J. Mostafa, "Topic detection and interest tracking in a dynamic online news source," in *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '03. USA: IEEE Computer Society, 2003, pp. 122–124.

[4] B. Mansouri, R. Campos, and A. Jatowt, "Towards timeline generation with abstract meaning representation," in *Companion Proceedings of the ACM Web Conference 2023*, ser. WWW '23 Companion. New York, NY, USA: Association for Computing Machinery, 2023, pp. 1204–1207.

[5] D. Gholipour Ghalandari and G. Ifrim, "Examining the state-of-the-art in news timeline summarization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Association for Computational Linguistics, Jul. 2020, pp. 1322–1334.

[6] Y. Sumikawa, "Event causal relationship retrieval," in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, ser. WI-IAT '21. New York, NY, USA: Association for Computing Machinery, 2022, pp. 318–325.

[7] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, Mar. 1970.

[8] O. Amayri and N. Bouguila, "Online news topic detection and tracking via localized feature selection," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 2013, pp. 1–8.

[9] K. Radinsky and E. Horvitz, "Mining the web to predict future events," in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, ser. WSDM '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 255–264.

[10] C. Zhang, J. Lyu, and K. Xu, "A storytree-based model for inter-document causal relation extraction from news articles," *Knowl. Inf. Syst.*, vol. 65, no. 2, pp. 827–853, Nov. 2022.

[11] B. Liu, F. X. Han, D. Niu, L. Kong, K. Lai, and Y. Xu, "Story forest: Extracting events and telling stories from breaking news," *ACM Trans. Knowl. Discov. Data*, vol. 14, no. 3, May 2020.

[12] C. C. Yang, X. Shi, and C.-P. Wei, "Discovering event evolution graphs from news corpora," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 39, no. 4, pp. 850–863, 2009.

[13] D. Huang, S. Hu, Y. Cai, and H. Min, "Discovering event evolution graphs based on news articles relationships," in *2014 IEEE 11th International Conference on e-Business Engineering*, 2014, pp. 246–251.

[14] S. M. Sarwar and J. Allan, "Query by example for cross-lingual event retrieval," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1601–1604.

[15] G. Tahmasebzadeh, E. Kacupaj, E. Müller-Budack, S. Hakimov, J. Lehmann, and R. Ewerth, "Geowine: Geolocation based wiki, image, news and event retrieval," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '21. New York, NY, USA: Association for Computing Machinery, 2021, pp. 2565–2569.

[16] D. Metzler, C. Cai, and E. Hovy, "Structured event retrieval over microblog archives," in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, ser. NAACL HLT '12. USA: Association for Computational Linguistics, 2012, pp. 646–655.

[17] K. Matsumaru, R. Ikejiri, and Y. Sumikawa, "Present causal relationship retrieval for historical analogy," in *Culture and Computing*, M. Rauterberg, Ed. Cham: Springer Nature Switzerland, 2023, pp. 536–547.

[18] K. Shi, X. Wang, J. Yu, L. Hou, J. Li, J. Wu, D. Yong, J. Xiao, and Q. Liu, "Cstory: A chinese large-scale news storyline dataset," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, ser. CIKM '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 4475–4479.

[19] T.-A. Hoang, K. D. Vo, and W. Nejdl, "W2e: A worldwide-event benchmark dataset for topic detection and tracking," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, ser. CIKM '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1847–1850.