Set-Aligning Framework for Auto-Regressive Event Temporal Graph Generation

Xingwei Tan[1,2], **Yuxiang Zhou**[2], **Gabriele Pergola**[1], **Yulan He**[1,2,3]
[1]Department of Computer Science, University of Warwick, UK
[2]Department of Informatics, King's College London, UK
[3]The Alan Turing Institute, UK
{Xingwei.Tan, Gabriele.Pergola.1}@warwick.ac.uk
{Yuxiang.Zhou, Yulan.He}@kcl.ac.uk

Abstract

Event temporal graphs have been shown as convenient and effective representations of complex temporal relations between events in text. Recent studies, which employ pre-trained language models to auto-regressively generate linearised graphs for constructing event temporal graphs, have shown promising results. However, these methods have often led to suboptimal graph generation as the linearised graphs exhibit set characteristics which are instead treated sequentially by language models. This discrepancy stems from the conventional text generation objectives, leading to erroneous penalisation of correct predictions caused by the misalignment of elements in target sequences. To address these challenges, we reframe the task as a conditional set generation problem, proposing a Set-aligning Framework tailored for the effective utilisation of Large Language Models (LLMs). The framework incorporates data augmentations and set-property regularisations designed to alleviate text generation loss penalties associated with the linearised graph edge sequences, thus encouraging the generation of more relation edges. Experimental results show that our framework surpasses existing baselines for event temporal graph generation. Furthermore, under zero-shot settings, the structural knowledge introduced through our framework notably improves model generalisation, particularly when the training examples available are limited.[1]

1 Introduction

Understanding the temporal relation between events mentioned in long documents is crucial to modelling complex text with articulated narratives. One of the widely adopted benchmarks for event temporal relation understanding is the SemEval 2013 TempEval-3 (UzZaman et al., 2013), requiring end-to-end generation of event temporal graphs directly from raw text. An event temporal graph is a natural representation of temporal information, with the nodes representing events and the edges the temporal relationships between them, such as "*before*", "*after*", or "*simultaneous*".

Existing studies typically approach the problem of constructing event temporal graphs through a two-step pipeline, with the first step focusing on detecting events in text, and the second step on classifying the temporal relations between them (McDowell et al., 2017; Ning et al., 2018b). However, such pipeline-based approaches suffer from well-known limitations, including (i) the need for fine-grained annotations at each step; and (ii) the potential for error propagation throughout the pipeline. In the first step, the event extractor aims at locating as many event triggers as possible in the given documents, leading to the inclusion of numerous trivial events that often lack relevance to the narrative and have no relation with other events. As a result, the next step for temporal relational extraction becomes burdened with many noisy events, significantly impacting the overall accuracy and efficiency of the models.

To address these limitations, Madaan and Yang (2021) introduced a reformulation of the task by generating event temporal graphs directly through conditional text generation. This approach allows for the use of pre-trained language models and, more importantly, overcomes the typical limitations associated with the pipeline architecture. While this method involved fine-tuning a text generation model, such as GPT-2, for the generation of linearised event temporal graphs as sequences, it fails to consider an important aspect. Specifically, it does not account for the fact that the target sequence (i.e. the list of event temporal relations) is order-invariant, and should therefore be treated as a *set* rather than as an ordered sequence. For example, the following two sequences represent the same temporal graph:

[1]For access to experimental code and data, please refer to: https://github.com/Xingwei-Warwick/Set-Aligning-Event-Temporal-Graph-Generation

```
S1: [(Cuomo leaving his office, before, speak to reporters),
     ···(Cuomo leaving, before, met with representatives)]
S2: [(Cuomo leaving, before, met with representatives),
     ···(Cuomo leaving his office, before, speak to reporters)]
```

In this scenario, the conventional text generation loss will (mistakenly) yield a high value because most of the tokens in the corresponding positions do not match, even though the event relations are the same. This issue has a detrimental effect on the model performance for several reasons. First, it discourages the language model from generating additional edges. Generating more edges implies a greater number of potential permutations in the edge sets, making it less likely to match the target. Secondly, if the initially generated edge in the sequence differs in token count from the one in the target, it causes all subsequent edges to misalign with the target, even if they are identical, leading to a high loss value.

In this work, we propose a Set-Aligning Framework (SAF) that enables efficient employment of LLMs for auto-regressive event temporal graph generation. SAF incorporates a group of novel regularisations, named Set Property Regularisations (SPR), along with augmented data, which aims at tackling the problems associated with the use of LM loss in contextualised graph generation by mitigating its penalisation towards the target sequences. For example, the S1 and S2 above are different sequences of the same edge set. Even if S1 has the same order as the target edge sequence and thus has a lower LM loss than S2, both of them will be added with the same SPR. Therefore, the relative difference of their loss values becomes smaller, which avoids overfitting the model towards the specific edge order of S1. Moreover, if the model explores generating one more edge after S2 and the edge is correct, the SPR value will decrease while the LM loss will probably increase.

Using the proposed SAF, we fine-tune language models from the T5 (Raffel et al., 2020) family with weak supervision. Additionally, we introduce the first human-annotated dataset for contextualised event temporal graph generation built on the New York Times, which we combine with existing event relation extraction datasets to evaluate the effectiveness of the SAF framework. Experiments on the newly annotated New York Times corpus[2] show that SAF significantly increases the number of generated edges, resulting in improved recall. Furthermore, we assess the performance of our approach on existing sentence-level event temporal relation extraction datasets, namely MATRES (Ning et al., 2018a) and TB-Dense (Cassidy et al., 2014), under zero-shot settings, and we find that the structural knowledge introduced through the proposed SAF has an even greater impact on model generalisation when the training examples available are limited.

[2]https://doi.org/10.35111/77ba-9x74

Our contributions are three-folded:

- We introduce a model-agnostic framework, called SAF, for event temporal graph generation. SAF incorporates novel Set-Aligning regularisations, data augmentation, and weak supervision techniques.

- We offer a human-annotated test set and a weakly-supervised dataset specifically designed for document-level event temporal generation.

- Our extensive experimental results in various settings demonstrate the effectiveness of our proposed model. Our thorough analysis shows that our SAF framework encourages language models to generate at least 24% more edges than previous graph generation approaches across various datasets.

2 Related Work

2.1 Event Temporal Graph

The task of event temporal graph extraction serves as an important task for evaluating an end-to-end system which takes raw text as input and output TimeML annotations (i.e., temporal relations) (UzZaman et al., 2013). Early attempts on the task include CAEVO (McDowell et al., 2017) and Cogcomptime (Ning et al., 2018b), which relied on a combination of statistical and rule-based methods. In recent years, more efforts have been put into developing specialised sub-systems with neural network-based approaches (Ning et al., 2019; Han et al., 2019a; Tan et al., 2021a). The emergence of large language models has paved the way for end-to-end learning, treating temporal graph generation as conditional text generation (Madaan and Yang, 2021). To tackle the set misalignment issue which remained unexplored in Madaan and Yang (2021), we propose a framework based on a group of novel Regularisations, aiming at enhancing autoregressive event temporal graph generation.

It is worth noting that there is another related and more widely-recognised task called *temporal* *relation extraction*, which aims at classifying the type of temporal links between pre-extracted events (Wang et al., 2020; Wen and Ji, 2021; Tan et al., 2023). While Han et al. (2019b) proposed a joint extraction model for events and event temporal relations, they rely on event extraction supervision signals, which our work does not need.

2.2 Graph Generation with Language Models

Generating graphs with language models has been explored in many areas. For example, Bosselut et al. (2019) fine-tunes GPT on the ATOMIC commonsense knowledge graph (Sap et al., 2019). Melnyk et al. (2022) proposed a multi-stage system for knowledge generation based on T5. However, these studies do not generate an entire graph in one generation. In contrast, Madaan et al. (2021) generated inference graphs using a combination of a graph generator and a graph corrector for queries in defeasible reasoning. Zaratiana et al. (2023) generate entities and entity relations with an auto-regressive LM, but they did not consider the set property of the target. Different from them, we focus on the set property of the generation sequence, which is particularly important in the setting where both the input document and output sequence are considerably longer.

2.3 Conditional Set Generation

Text generation models are primarily designed for generating text with strict linear orders, making them suboptimal for generating sets. This limitation has been acknowledged in recent NLP research, where efforts have been made to adapt seq2seq frameworks for tasks like multi-label classification and keyword generation (Qin et al., 2019; Ye et al., 2021). Vinyals et al. (2016) studied the general challenge of using sets as either input or target output for text generation models. They found in both cases, the order of elements in the set has a significant impact on convergence and final perplexity. This implies that there may exist an optimal order for the input or output set sequence, and they proposed allowing the model to search for this order during training. Instead of resorting to exhaustive search, Madaan et al. (2022) proposed a data augmentation method to enforce order-invariance and prepend the set's cardinality to the target sequence to ensure the correct cardinality. While previous research has tackled multi-label prediction and keyphrase generation, our work delves into the unique challenges presented by event temporal graph generation, which involves long sequences and partially ordered properties.

In a more general sense, the object detection task from computer vision also involves set prediction (Chen et al., 2022). Carion et al. (2020) use parallel decoding to generate the elements in a set based on object queries. Tan et al. (2021b) adopted a similar approach in name entity recognition with a non-autoregressive decoder. Different from the entities in images, the set elements (event relations) in our task are not concrete spacial objects or text spans but instead are varied in length and scattered across each document. This makes object queries and non-autoregressive decoding inapplicable in our settings.

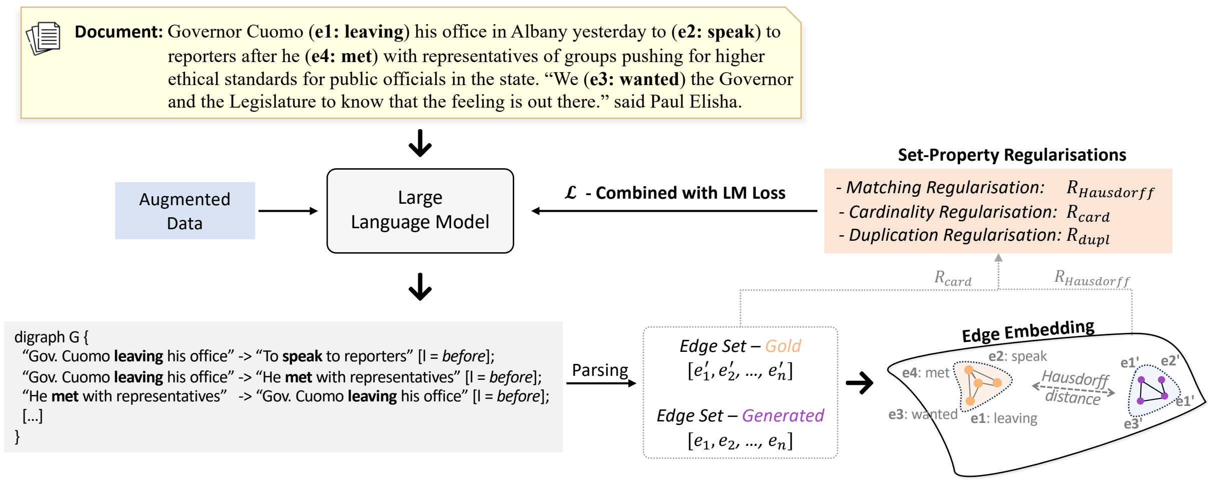
3 Set-Aligning Framework

Madaan and Yang (2021) first explored the possibility of end-to-end event temporal graph generation using neural language modelling. Since then, however, this task has remained under-explored, with numerous unresolved issues. To elaborate, the first concern is that Madaan and Yang (2021) framed graph generation as a conventional sequence generation problem, whereas it is fundamentally a set generation problem. Secondly, the dataset they built primarily consists of small-sized graphs, failing to challenge the model in terms of document-level understanding. Lastly, their investigation mainly centred on GPT-2, while the landscape of LLMs has evolved with the emergence of models featuring distinct structures (e.g., encoder-decoder) and new paradigms (e.g., in-context learning) in recent years. In this study, we address these three aspects to enhance the understanding of sequence-to-sequence temporal graph generation.

Although our proposed framework is designed to be model-independent, several factors have led us to choose Flan-T5 as the base model for our experiments: (i) Based on our preliminary experiments, Flan-T5-base hits the sweet spot in terms of performance vs. resource consumption, allowing us to test more variants; (ii) its encoder-decoder structure is well-suited to document-level graph generation, due to its efficiency in processing comprehensive information in lengthy documents.

3.1 Event Temporal Graph Modelling as Edge Set Generation

An event temporal graph is a directed graph with no isolated vertex. Each edge in the graph describes a temporal relation between two events, and self-loops are not permitted. Following Madaan and



Document: Governor Cuomo (**e1: leaving**) his office in Albany yesterday to (**e2: speak**) to reporters after he (**e4: met**) with representatives of groups pushing for higher ethical standards for public officials in the state. "We (**e3: wanted**) the Governor and the Legislature to know that the feeling is out there." said Paul Elisha.

```
digraph G {
  "Gov. Cuomo leaving his office" -> "To speak to reporters" [l = before];
  "Gov. Cuomo leaving his office" -> "He met with representatives" [l = before];
  "He met with representatives"   -> "Gov. Cuomo leaving his office" [l = before];
  [...]
}
```

Augmented Data
Large Language Model
$\mathcal{L}$ - Combined with LM Loss
Set-Property Regularisations
- Matching Regularisation: $R_{Hausdorff}$
- Cardinality Regularisation: R_{card}
- Duplication Regularisation: R_{dupl}
R_{card}
$R_{Hausdorff}$
Parsing
Edge Set – Gold
$[e_1', e_2', \ldots, e_n']$
Edge Set – Generated
$[e_1, e_2, \ldots, e_n]$
Edge Embedding
e4: met
e2: speak
e3: wanted
e1: leaving
Hausdorff distance
e1'
e2'
e3'
e1'

Figure 1: Set-Aligning framework (SAF).

Yang (2021), we represent these graphs by linearizing them into strings using the DOT graph description language (Gansner, 2006) (example shown in Figure 1). Given that event temporal graphs do not have isolated vertices, the sequence essentially represents the edge set of the graph.

We model the probability of generating a string y, which is a linearised representation of the event temporal graph G, conditioned on a document $X = (x_1, x_2, \ldots, x_n)$ using a language model:

$$p_{\mathrm{LM}}(y|X) = \prod_{t=1}^{T} p(y_t|X, y_{<t}) \tag{1}$$

where y is a string formatted in DOT notation.

3.2 Data Augmentation

The target sequences of event temporal graph generation are essentially sets rather than strictly ordered text sequences. Therefore, conventional text generation loss can inadvertently penalise the token order and force the arrangement of elements to match the order in the target sequence, which is not necessarily the optimal order. This enforced order may lead to sub-optimal performance (Vinyals et al., 2016). A potential solution is to introduce random permutations of set elements as augmented training examples, which has already been shown effective in tasks like multi-label classification and keyphrase generation (Madaan et al., 2022). Specifically, in the context of event temporal graph generation, the elements correspond to the edges in the target string. The substrings representing the edges are randomly shuffled, while the rest of the string remains unchanged.

Prepending the set cardinality of the ground-truth edge set to the generation target may also help constrain the generation model to avoid over-generation (Madaan et al., 2022). However, such attempts in our preliminary experiment led to an approximate 4% drop in edge F_1 score, despite a significant reduction in the number of generated edges. Thus, we decided not to incorporate the cardinality into the final framework.

3.3 Set Property Regularisations (SPR)

Simply adding augmented data to train models does not address the fundamental issue of set alignment. Several challenges arise in this approach. First of all, it is unrealistic to add all permutations, especially when dealing with long documents containing numerous event relations, as the training data will grow at a rate proportional to the factorial of the cardinality of the target set. More importantly, with each augmented example, the loss function would still penalise the unobserved permutations of the set. This would make the training unstable.

The core challenge lies in finding an effective way to compare the linearized target graph with the linearized generated graph, without relying on a strict token-by-token comparison as in conventional text generation. To tackle this issue, we propose introducing modifications to the generation objective. As the linearized graph essentially represents the edge set of the graph, we can simplify the graph comparison problem into a set comparison problem. Our approach involves several components. Firstly, we add a set cardinality regularisation to encourage the model to generate an adequate number of temporal relation edges. Then, we introduce a duplication regularisation to penalise any repetition of elements in the edge set. Lastly, we design a set matching regularisation that assesses the semantic similarity between elements in the target edge set and those in the generated edge set. Collectively, the above regularisations are referred to as Set Property Regularisations (SPR). They are integrated with the token-level cross-entropy loss through a weighted average.

To compute the set property regularisations, a graph string needs to be first sampled from a language model given a training input. Then, this sequence is parsed into a list of edges E, where each edge e is a triplet consisting of a head event, a relation type, and a tail event (h, r, t). The parsing is done with a rule-based parser which turns the graph text string into a structured data representation. As the edges are loaded in a structured list, the number of edges and duplicated edges can be counted. Let $\mathcal{E}$ denote the set of all the unique edges in E. The values for the set cardinality regularisation and the duplication regularisation can be computed as follows:

$$\mathcal{E} = \{e|e \in E\} \tag{2}$$

$$R_{\text{dupl}} = \frac{|E| - |\mathcal{E}|}{|\mathcal{E}|} \tag{3}$$

$$R_{\text{card}} = \frac{\text{abs}(|\mathcal{E}'| - |\mathcal{E}|)}{|\mathcal{E}|} \tag{4}$$

where function abs$(\cdot)$ denotes taking the absolute value, $\mathcal{E}'$ denotes the ground-truth edge set.

To compute the set matching regularisation, we assess the similarity between the generated set and the target set by comparing the semantic similarity of the edges across the two sets. We take the last layer of the decoder's representations of the respective tokens as the semantic representations of the events and the relation type. Then, we concatenate these representations as the semantic representation of each edge:

$$z_h = H_{[h_1, h_2, ..., h_m]} \tag{5}$$

$$z_r = H_{[r_1, r_2, ..., r_s]} \tag{6}$$

$$z_t = H_{[t_1, t_2, ..., t_n]} \tag{7}$$

$$\bar{e} = \left[\text{pool}(z_h); \text{pool}(z_r); \text{pool}(z_t)\right] \tag{8}$$

where H is the last-layer hidden states of the decoder. $[h_1, ..., h_m]$, $[r_1, ..., r_s]$, and $[t_1, ..., t_n]$ are the indices of the head event, relation type, and tail event, respectively. z_h, z_r, z_t denote the semantic representations of the head event, relation type, and tail event, respectively. pool$(\cdot)$ represents the average pooling function. $\bar{e}$ denotes the semantic representation of the edge.

We now possess two sets of embeddings: one compassing the edge embeddings extracted from the target graph, and the other containing the edge embeddings derived from the generated graph. Essentially, they can be considered as two sets of points in the representation space. Thus, we can measure the similarity of the two graphs by measuring the distance between the two point sets (manifolds) in the representation space. The Hausdorff distance, originally defined to measure the separation between two subsets within a metric space, has recently found applications in machine learning for measuring the distance between two sets of embeddings (Schutze et al., 2012; Wang et al., 2023). We compute the average Hausdorff distance as the measure:

$$\begin{aligned} d_H(\mathcal{E}', \mathcal{E}) = & \frac{1}{|\mathcal{E}'|} \sum_{\bar{e'} \in \mathcal{E}'} \min_{\bar{e} \in \mathcal{E}} d_{cos}(\bar{e'}, \bar{e}) \\ & + \frac{1}{|\mathcal{E}|} \sum_{\bar{e} \in \mathcal{E}} \min_{\bar{e'} \in \mathcal{E}'} d_{cos}(\bar{e'}, \bar{e}) \end{aligned} \tag{9}$$

where the distance of an edge pair is computed by the cosine distance $d_{cos}(\cdot)$.

When the model generates the set elements in a different order than the target sequence, the token-level cross-entropy loss would be high. If the model generates more correct elements as suffixes of the sequence with a wrong order, the loss value would probably increase further. However, the SPR will have a lower value and thus alleviate the discouragement for generating more elements caused by the token-level cross-entropy loss.

3.4 Fine-tuning with Set Property Regularisations

Unlike the set prediction methods based on parallel decoding (Carion et al., 2020; Tan et al., 2021b), SPR cannot be directly used as the main objective in auto-regressive generation. There are two primary reasons for this. The first reason is that obtaining the SPR requires sampling from the decoder, which would reduce the training speed significantly. Moreover, the second reason is that the language model will struggle to generate sequences in DOT format accurately because learning the token dependency for such format requires the language modelling objective. Consequently, the sequence parser will fail to recognize any valid edges within the sequence, resulting in high SPR values and hindering the training.

To avoid the problems mentioned above, we introduce the SPR after a certain number of fine-tuning iterations. Once the model has acquired a basic proficiency in generating correct DOT sequences, the SPR can function as intended. SPR can prevent the language model from overfitting to the order of the target set shown in the training samples.

We explored alternative approaches to incorporate SPR, but they reported inferior performance compared to the method eventually included in our framework. We discuss those alternative methods in the Appendix A.

4 Experiment

4.1 NYT Temporal Event Graph Dataset

	NYT-train	NYT-test	NYT-human
Total documents	$18,263$	$1,000$	22
Total events	$846,022$	$47,251$	661
Node degree	2.52	2.54	2.34
Total relations	$1,066,264$	$60,056$	528
before	$578,216$	$32,729$	465
after	$412,704$	$23,200$	0
includes	$7,922$	450	12
is_included	$41,964$	$2,332$	0
simultaneous	$25,458$	$1,345$	51

Table 1: The statistics of the NYT temporal event graph dataset. Node degree represent the average number of relations each event has.

There are several event temporal relation extraction datasets with pairwise event relation annotations, such as MATRES and TBD. It is theoretically possible to convert these annotations into document-level event temporal graphs. However, our preliminary experiments have shown that even when merging all of these datasets (resulting in 4,684 training documents), it is not sufficient to fine-tune a large language model to achieve acceptable performance. To address this limitation, we opted to build a significantly larger dataset on a selection of data from the New York Times (NYT) corpus using a weak supervision approach, drawing inspiration from the work of Madaan and Yang (2021). Nevertheless, we introduced additional steps in the data selection process to ensure that the selected documents contain high-quality event temporal graphs, which were not taken in Madaan and Yang (2021).

Firstly, we performed topic modelling using Latent Dirichlet Allocation (LDA) on the MATRES and TBD datasets to extract a set of topics. Then, we identified general descriptors that are semantically similar to these topics (e.g., politics, diploma, sports, etc.). This selection process was crucial because, following training with noisy labels, our intention was to evaluate the model's performance on these datasets under zero-shot settings. We further analysed the most noteworthy events in these descriptors to ensure they were narrative-oriented, because articles that weave stories tend to contain a wealth of event temporal relations. To identify the most significant events, we employed a metric similar to TF·IDF which we could describe as "event frequency × inverse-descriptor frequency".

$$\text{ef}\cdot\text{idf} = \frac{f_{\mathfrak{e},d}}{\sum_{\mathfrak{e}'\in d} f_{\mathfrak{e}',d}} \cdot \log \frac{|D|}{|\{d \in D : \mathfrak{e} \in d\}|} \quad (10)$$

where $\mathfrak{e}$ is an event and d is a descriptor. $f_{\mathfrak{e},d}$ is the number of times that event $\mathfrak{e}$ occurs in the documents with the descriptor d. $\sum_{\mathfrak{e}'\in d} f_{\mathfrak{e}',d}$ is the total number of event occurrence in the descriptor d. $|D|$ is the total number of descriptors in the corpus. $|\{d \in D : \mathfrak{e} \in d\}|$ is the number of descriptors where the event $\mathfrak{e}$ appears.

The descriptors that are selected and the number of documents in them are listed in the Appendix D.1. After choosing the documents, we acquire the event temporal graph by running an off-the-shelf event and temporal relation extraction tool called CAEVO (McDowell et al., 2017). CAEVO is more scalable than Cogcomptime (Ning et al., 2018b), making it suitable for building a large-scale dataset.

Then, each temporal graph is represented in DOT format, and every event verb is prefixed and suffixed with its noun phrase and object, respectively. Note that we did not break the documents into short segments as Madaan and Yang (2021) did. Instead, we keep the data strictly at the document level which is a more challenging setting because the model needs to analyse the entire document and generate a much larger graph. In the dataset we built, a target graph has about 46 nodes and 58 edges on average. While in Madaan and Yang (2021), the average number of nodes is 4 and the average number of edges is 5 in a document-level event temporal graph. Moreover, their events have 1.54 relations on average, while events in our data have 2.52 relations on average, showing that the graphs in our dataset are much more complex. In practice, these complex documents are the ones that require analysis, and a model developed based on simpler inputs cannot handle them directly.

4.2 Human-annotated Test Data

Aside from testing with the CAEVO-created data, we recruited human annotators to annotate a test split of the NYT data. We performed a preprocessing step regarding the relation types by merging the reciprocal relations, such as transforming *after* into *before*, *is_included* into *includes* by swapping the head and tail events. For example, "I had dinner after I had lunch" is equivalent to "I had lunch before I had dinner". This processing not only streamlined the annotation process but also enhanced the model performance (refer to experimental results in Appendix C). We recruited crowd workers from Prolific[3] platform, which is a research-focused platform providing verified human workers. We recruited 24 participants in total (including pilot testing runs). To make sure the participants can understand and annotate the article efficiently, we only recruited native English speakers who have an education level higher than High school diploma/A-levels. We put 4 documents, which are randomly sampled from the same descriptor set as the training and testing of the selected NYT corpus, into each unit task. There is a shared document across all the tasks to compute the inter-annotator agreement (IAA). To minimize discrepancy, we asked 2 participants to first identify the event triggers in each unit task. We then merged the event annotations from the participants by taking the union of the spans (if there are overlapped spans, we take the longer span). Then, we asked another participant to annotate the event temporal relation based on the identified events. We also included the outputs from the CAEVO model to serve as examples, but we explicitly asked the participants to correct the annotations by adding, removing, or changing the CAEVO's annotations. In the end, we collected 22 documents as the human-annotated test set. On the event identification, we compute IOU (Intersection over Union) as a measure of agreement between the annotators. Average across 7 tasks, the IOU between the event spans is 0.8986. For the relation annotations, we compute the average Cohen's κ of every participant pair in the relation annotation task (on the shared document). The average Cohen's κ is 0.7465. Details of instructions and interfaces are in Appendix D.1.

The statistics of the constructed datasets are shown in Table 1. The distributions of relation types are highly imbalanced, with a majority falling into either the *before* or *after* categories. We also evaluated the trained models on the MATRES test set (comprising 20 documents) and TBD test set (consisting of 9 documents), both of which are based on human annotations and processed into DOT using the methods previously described.

[3]prolific.com

4.3 Model Setting

We employed **Flan-T5-base** as the backbone model for contextualised graph generation. We first trained a Flan-T5-base model following the same setup as in Madaan and Yang (2021) as the baseline. **SAF (w/o DA)** is our proposed framework without the augmentations of edge order but with Set Property Regularisations (SPR). **SAF (w/o SPR)** is the framework without the use of SPR but with the augmentations. As our SAF framework with SPR requires additional training steps and the augmentations enlarge the training set, we keep the number of training steps balanced in the methods to exclude the influence of seeing different amounts of training data. The model is trained for 10 epochs, with each document being augmented through 4 random permutations, followed by a further 3 epochs of training, during which the SPR are adopted without permutations. We use a learning rate of $2e-5$, along with a weight decay of 0.01. Batch size of 5 before SPR, and 3 during SPR because additional memory is required for sampling. We used AdamW optimizer (Loshchilov and Hutter, 2019). We use the beam search (Graves, 2012) with a beam size of 5 and a maximum length of 2048 to sample results. We balanced the training steps in the compared methods to make sure they saw the same amount of training data. Experiments are conducted on a GPU node under an HPC cluster using 4 Nvidia A100 80G GPUs. The models are trained based on 3 random seeds (ChatGPT was tested for 3 times) and the metrics are the average values of them. Training with the augmented data for 10 epochs requires approximately 19 hours. Training with SPR for 3 epochs takes about 27 hours. Training a vanilla Flan-T5-base for the same number of training steps demands approximately 20 hours.

4.4 Evaluation Metrics

Following the previous research (Madaan and Yang, 2021), we evaluate the results using the metrics of precision, recall, and F_1 score for both node set and edge set predictions. The primary metric is the edge F_1 because the quality of the node generation is also reflected in it.

	NYT-test			NYT-human		
	P^E	R^E	F_1^E	P^E	R^E	F_1^E
Flan-T5-base	51.27	32.43	39.73	22.61	25.88	24.14
SAF (w/o DA)	50.28	34.82	41.15	25.80	32.13	28.62
SAF (w/o SPR)	**51.88**	36.64	42.95	**27.08**	34.91	30.50
SAF	50.97	**39.96**	**44.80**	25.92	**40.21**	**31.52**

Table 2: Edge-based metrics on the NYT datasets

	NYT-test			NYT-human		
	P^N	R^N	F_1^N	P^N	R^N	F_1^N
Flan-T5-base	**75.52**	58.24	65.76	53.36	47.66	50.35
SAF (w/o DA)	75.34	60.64	67.20	**54.86**	50.43	52.55
SAF (w/o SPR)	75.43	62.36	68.27	54.14	51.59	52.84
SAF	75.47	**65.16**	**69.95**	53.63	**54.51**	**54.06**

Table 3: Node-based metrics on the NYT datasets

4.5 Results

As shown in table 2 and 3, SAF (w/o SPR) consistently outperform Flan-T5-base in terms of F_1 scores on the NYT-test and NYT-human datasets, suggesting the benefits of introducing permutated training examples. For example, SAF (w/o SPR) improves upon Flan-T5-base by about 3% on the NYT-test and 6% on the NYT-human in terms of edge F_1. SAF (w/o DA) achieves an improvement of approximately 1.5% on the NYT-test and 4.5% on the NYT-human datasets in terms of edge F_1, demonstrating the effectiveness of SPR alone. Furthermore, our SAF model yields the best performance when both SPR and augmentation are incorporated. We also observe that models utilizing SAF have much higher edge recalls while their edge precision scores are either similar or occasionally even lower than those of other models. This suggests that the performance improvement primarily comes from the generation of more edges. This observation is reinforced by the information presented in Figure 2, where models trained with SAF can generate $24\% - 48\%$ more edges compared to the conventional text generation framework on these datasets. These additional edges play a pivotal role in the improvement of the edge F_1 since precision stays nearly the same.

It is worth mentioning that the NYT-human dataset has a different label distribution compared to the NYT dataset used for training, where its events and event temporal relations were produced by CAEVO. Notably, the frequency of *simultaneous* is significantly higher, accounting for 9.66%, in contrast to the 2.39% observed in the training set (see Appendix D.1 for more comprehensive analyses). Based on our observation, it appears that

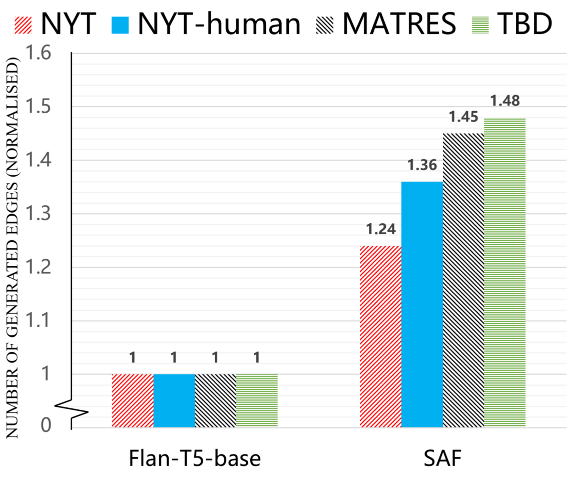


Figure 2: The comparison of generated edges between SAF and vanilla Flan-T5-base. The y axis is normalised by dividing the number of edges generated by Flan-T5-base in the respective datasets.

	MATRES-test			TBD-test		
	P^E	R^E	F_1^E	P^E	R^E	F_1^E
ChatGPT	10.58	6.56	8.09	25.92	5.94	9.66
Flan-T5-base	13.06	7.16	9.25	23.26	4.59	7.67
SAF	**18.05**	**14.31**	**15.96**	**37.53**	**11.04**	**17.05**

Table 4: Experiment results on human-annotated MATRES and TBD under the zero-shot setting.

human annotators tend to apply a more lenient criterion for the *simultaneous* label whereas CAEVO enforces a stricter definition of this label.

Similar trends are also observed in Table 4, which were obtained through evaluation on MATRES and TBD. We used the models trained on the NYT training set to test on these datasets under zero-shot settings. ChatGPT shows our best attempts to generate event temporal graphs with gpt-3.5-turbo model through Openai API. We used two hops: (i) ask ChatGPT to generate events from the documents, (ii) ask ChatGPT to generate an event temporal graph based on the generated events and the documents. The results show that ChatGPT is outperformed by fine-tuned models, which is in line with the recent papers on exploring ChatGPT's ability on event understanding (Li et al., 2023; Chan et al., 2023; Gao et al., 2023).

Upon examining the responses of ChatGPT, it appears that it conceptualises events as a broader and high-level notion which diverges from the definition commonly used by the information extraction community in event processing. In our task, each predicate can signify an event, but ChatGPT tends to approach event identification more like a summarisation task, where it summarises text chunks in a document. This likely explains why ChatGPT identified only approximately half of the events present in the target graph, resulting in low recall. These observations confirm that event temporal graph generation cannot be solved solely with prompt engineering on ChatGPT. It is noteworthy that ChatGPT consistently produces graphs in the correct DOT format across both the MATRES-test and TBD-test datasets, indicating that formatting issues are not the primary factor in ChatGPT's underperformance. The details regarding the inputs, outputs, and parameter settings for ChatGPT are presented in Appendix E.

4.6 Error Analysis

A major error type we found is that the model often fails to deduce temporal relationships that involve inference. This is due to the reliance of weak supervision signals provided by CAEVO, which primarily rely on syntactical rules. Consequently, this problem led to a lower edge F_1 on the human-annotated test set, as human annotators provided many temporal relations that were inferred through commonsense reasoning. For example, the model does not perceive a clear temporal sequence in the sentence:"<person A> won the gold medal in women's 1,500m. <person B> won the silver and <person C> won the bronze." However, human annotators can readily identify an obvious temporal order among "<person A> won", "<person B> won", and "<person C> won", as it aligns with the common knowledge that in a race, the first person who crossed the finish line won the gold, followed by the silver and the bronze winners.

5 Conclusion

This study proposes a framework for fine-tuning language models to generate event temporal graphs directly from raw documents in an end-to-end manner. The proposed framework includes a data augmentation method and set property regularisations to mitigate the problem caused by conventional generation loss, promoting the generation of more edges by language models and, consequently, leading to improved performance. Extensive experiments show the effectiveness of our proposed model on multiple widely used datasets with real-world articles. The thorough analysis demonstrates that our framework can encourage language models to generate more edges for constructing event temporal graphs in various settings.

Limitations

Due to the presence of noisy labels used in fine-tuning, a major limitation of the proposed method is the inclusion of many imaginary events, trivial events, and negative expressions of events. For example, CAEVO identified phrases like "<someone> did not **fire**" as an event. While "fire" serves as a predicate and the notion of "did not **fire**" can hold narrative significance, it may not be entirely suitable within the context of event temporal graphs. This is because it is not about the occurrence of an action or a change of state, but rather describes the absence of an event. Similarly, in some articles, there are descriptions of multiple potential future developments, such as "he might **buy** product A". Including such expressions as events might introduce confusion into the event temporal graph, as these represent possibilities rather than actual occurrences. This problem mainly arises from the behaviour of the CAEVO method, which primarily focuses on identifying fine-grained predicates as events. The resolution to this problem lies in obtaining better-quality supervision signals which focus on salient events (i.e., events which are mentioned frequently and are important to the narrative).

Ethics Statement

The proposed method analyses the text provided and extracts relevant information from it. The algorithm cannot acquire information beyond the boundary of the given text. Thus, any associated risks stem solely from the data itself. This research only utilised publicly available data. As long as the data input to the model is collected according to the relevant data policies and guidelines, the proposed method does not introduce further risks.

Acknowledgements

This work was supported in part by the UK Engineering and Physical Sciences Research Council through a Turing AI Fellowship (grant no. EP/V020579/1, EP/V020579/2). Computing facilities were provided by the Scientific Computing Research Technology Platform of the University of Warwick, and the CREATE HPC platform of the King's College London (KCL-CREATE, 2024).

References

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi.