國立臺灣大學電機資訊學院資訊工程研究所

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

SEEN: 以結構化事件增強網路偵測與解釋資訊召回需求

SEEN: Structured Event Enhancement Network for Explainable Need Detection of Information Recall Assistance

林佑恩

You-En Lin

指導教授: 陳信希 博士

Advisor: Hsin-Hsi Chen Ph.D.

中華民國 111 年 8 月

August, 2022

doi:10.6342/NTU202202609

國立臺灣大學碩士學位論文
口試委員會審定書

SEEN：以結構化事件增強網路偵測與解釋資訊召回需求

SEEN: Structured Event Enhancement Network for Explainable Need Detection of Information Recall Assistance

本論文係林佑恩君（學號 R09922A08）在國立臺灣大學資訊工程學系人工智慧碩士班完成之碩士學位論文，於民國 111 年 8 月 18 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

陳信希

鄭卜壬 （指導教授） 蔡銘峰

陳冠宇

系 主 任 洪士灝

Acknowledgements

衷心的感謝我的指導老師陳信希教授兩年來的指導，不僅提供良好的研究環境以及充足的資源，使我可以專心地投入研究，更感謝老師不論多麼煩忙都會撥空與學生討論研究方向並提供建議，使得研究成果得以完善。也十分感謝翰萱學長及重吉學長總是能在討論時提出不同的看法，並在卡關時提出各種建議。此外，特別感謝安孜學姊願意在煩忙之餘撥出時間與我討論，不管在研究題目的發想亦或是實作方法的討論，都能給於適當的建議，在探討學術期間遇到的疑惑也都願意與我分享經驗及解惑，使我對於研究的本質有更近一步的認識。

再者，十分感謝實驗室成員給予我的陪伴與提攜，感謝建宏學長、聖倫學長不僅能在研究卡關時指引迷津，更帶著我一步步熟悉實驗室與學校。感謝生活日誌組的泰德學長、宜珮學姊、宏哲、偉鋒、艾霓一起討論、探索生活日誌的相關研究，並分享彼此學習到的新知識。此外，感謝同屆的孟寰、之遙、實驗室的學弟妹承之、哲韋、承光、羿寧、恬儀大家一起做研究，以及實驗室的網管禹廷學長、韋霖、彥斌和我一起維護工作站機器，也感謝又慈協助打點實驗室的諸多瑣事，最後感謝家人的陪伴與支持，使我在研究的路上可以沒有後顧之憂一路往前。謝謝大家。

doi:10.6342/NTU202202609

摘要

在回憶生活經歷時，人們經常忘記或混淆生活事件，所以提供資訊召回的服務是需要的。而以前關於資訊召回的研究主要是被動式提供，也就是使用者透過給定生活事件來評估是否需要資訊召回服務。然而，很少有研究涉及由系統主動偵測人們是否需要資訊召回服務。在本文中，我們透過比較同一作者在兩個不同時間點、針對同一事件所寫的敘述，來確定用戶在描述他們的過往的生活經歷時是否遇到困難。因此，我們使用標記者根據個人真實生活經歷組成的資料集來偵測觸發資訊召回服務的正確時間。此外，我們也提出一個模型–結構化事件增強網路（SEEN），它可以檢測到標記者撰寫的生活經歷是否包含不一致、額外新增或是被遺忘的生活事件。而此模型中還包含我們提出的一種特殊機制，我們透過這種機制來融合以生活事件為基礎所構建的無向圖和語言模型所產生的文字嵌入向量。同時，為了進一步提供具解釋性的服務，我們的模型會從生活事件的無向圖中選擇相關的節點用以當作參考事件。而實驗結果也表明，我們的模型在偵測資訊召回需求的任務取得了很好的成果，提取出的參考事件也可以有效作為補充資訊，提醒用戶他們可能想要召回的生活事件。

關鍵字：生活日誌、資訊召回、個人知識庫

doi:10.6342/NTU202202609

Abstract

When recalling life experiences, people often forget or confuse life events, which necessitates information recall services. Previous work on information recall focuses on providing such assistance reactively, i.e., by retrieving the life event of a given query. What is rarely discussed, however, is a proactive system that is capable of detecting the need for information recall services. In this paper, we propose determining whether users are experiencing difficulty in recalling their life experiences by comparing the events described in two retold stories written at different times. We use a human-annotated life experience retelling dataset to detect the right time to trigger the information recall service. We also propose a pilot model–Structured Event Enhancement Network (SEEN) that detects life event inconsistency, additional information in life events, and forgotten events. A fusing mechanism is also proposed to incorporate event graphs of stories and enhance the textual representations. To explain the need detection results, SEEN simultaneously provides support evidence by selecting the related nodes from the event graph. Experi-

doi:10.6342/NTU202202609

mental results show that SEEN achieves promising performance in detecting information needs. In addition, the extracted evidence can be served as complementary information to remind users what events they may want to recall.

Keywords: lifelogging, information recall, personal knowledge base

Contents

doi:10.6342/NTU202202609

doi:10.6342/NTU202202609

doi:10.6342/NTU202202609

List of Figures

List of Tables

doi:10.6342/NTU202202609

Chapter 1 Introduction

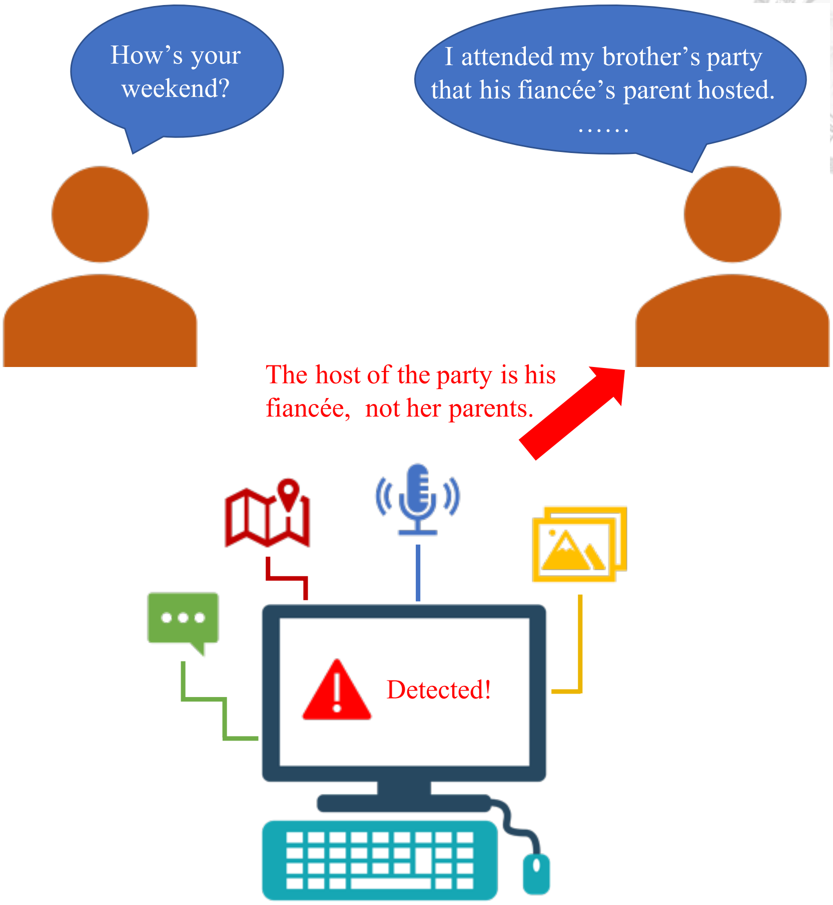
1.1 Motivation

People have to deal with many events in their daily life. As time passes, they might forget details about their past experiences. Forgetting the exact name of people or places or things and mixing up life events is a common occurrence. This explains the importance of an information recall system that helps people bring to mind what they are trying to recall. In recent years, people have often recorded their experiences via writing diaries or blogs and posting videos or photos on social networking services (Facebook, Instagram, Twitter, etc.). These personal records can be regarded as kinds of lifelogs. For instance, the posts on Twitter or Facebook are textual lifelogs, the GPS locations are numerical lifelogs, the talks on podcasts are auditory lifelogs, and photos on Instagram are visual lifelogs. However, although these services record our experiences in numerous aspects, we cannot utilize these records for information recall assistance. While recalling or retelling life experiences, the services only allow us to scan the records instead of proactively detecting the need for the information recall assistance, not to mention providing it.

Yen et al. (2021a) propose reactive and proactive service modes for an information recall system. In reactive mode, users directly ask the system about their life events, whereas in proactive mode, the system attempts to automatically detect whether users need memory

doi:10.6342/NTU202202609

Figure 1.1: Example of proactive information recall assistance.


How's your weekend?
I attended my brother's party that his fiancée's parent hosted. ……
The host of the party is his fiancée, not her parents.
Detected!

recall assistance and then provides the information they seek to recall. For reactive mode, studies have been done on visual lifelog recall (Chu et al., 2020, 2019; Gurrin et al., 2016, 2017, 2019, 2020), which focuses on the construction of a multimodal retrieval model that enables users to search through photos using textual queries. Yen et al. (2021b) propose an information recall system to answer questions about life experiences over a personal knowledge base. In contrast to reactively receiving users' requests, proactive mode, which detects the right time to trigger the information recall service, is still little explored. In this thesis, we propose a pilot study to proactively detect the user's need for information recall assistance.

One common use case of memory recall assistance occurs in human conversation. To identify whether people have difficulties in recalling past experiences, Wang et al.

doi:10.6342/NTU202202609

(2018) propose a model to detect speech hesitation. Here, we focus on detecting the need for information recall support in people's narratives. Specifically, we seek to detect the following four situations in narratives to determine whether to trigger the service:

1. If the description of the life event is consistent with the user's past experience, no memory recall assistance is needed.

2. Since people cannot remember every detail of their life experiences, we may unconsciously draw on similar but unrelated events to describe an experience that leads to a conflict with the established facts. It is essential to identify the description that is inconsistent with these facts, and retrieve those facts as an explanation to inform the user.

3. For the case where the narrative ends without relevant events mentioned, the user may have forgotten the events. The system must remind the user of these forgotten events.

4. The user may elaborate on additional events that were not logged before. This additional information could be details about events in lifelogs or they could be previously unlogged events. The system should distinguish whether events are additional or conflict with the facts, and should update the lifelogs with the new information.

To the best of our knowledge, no dataset is available for this purpose. For this reason, we extended the Hippocorpus dataset (Sap et al., 2020) with new life event annotations as cases where users encounter problems and require recall assistance. Sap et al. (2020) invited crowd-workers to write stories about their life experiences, and asked them to write those stories again a few months later. As such, the nature of Hippocorpus meets our

doi:10.6342/NTU202202609

requirement. In Hippocorpus, life experiences written the first and the second times are referred to here as pre-retold and post-retold stories, respectively. The need for information recall is detected by comparing the pre-retold and post-retold stories.

In this thesis, we propose a model to identify the event types in post-retold and pre-retold stories. The model is referred to as *structured event enhancement network* (SEEN). A transformer-based language model is used for encoding textual data. To encode the structured information of event description in stories, we construct an event graph by utilizing life event triples. To further capture the relations between events, the results of coreference resolution are incorporated into the event graph. The graph is encoded by the graph attention network (GAT) (Brody et al., 2022; Veličković et al., 2018) and fused with the language model for integrating textual and structured information.

In addition, our model will extract the relevant events in a story pair as support evidence to explain the decision of the prediction. In this way, the user will easily recall the forgotten events. In sum, the contributions of our work are threefold:

1. We introduce the task of detecting the need for information recall in a narrative and providing the related information as the support evidence.
2. We present the NIR dataset, a human-annotated life experience retelling dataset for detecting the **n**eeds of **i**nformation **r**ecall.
3. To detect information needs, we propose the structured event enhancement network (SEEN). The identified event types and extracted support evidence can assist users in recalling their past experiences and clarifying the confusing events.

doi:10.6342/NTU202202609

1.2 Thesis Organization

The remaining paragraphs of this thesis are organized as follows. Chapter 2 reviews the related work on lifelogging, structured information, and natural language inference. Chapters 3 introduce the dataset we extended, Hippocorpus, and the detail of our dataset construction. Chapter 4 define the goals of our tasks and the detail of our proposed model. Chapters 5 and 6 show the experiment results and the model analysis. Chapters 7 discuss the statistical data of our dataset. Chapter 8 concludes this thesis and addresses the future works.

doi:10.6342/NTU202202609

Chapter 2 Related Work

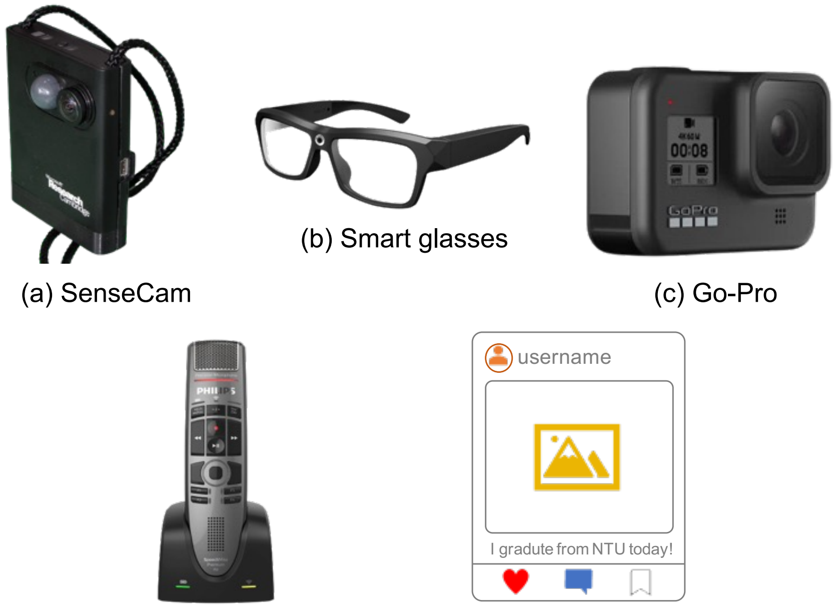
2.1 Lifelogging

The initial thought of lifelogging can date back to 1945, Bush et al. (1945) proposed a hypothetical system, Memex, allowing people to store all the knowledge collected in their lifetime and be capable of consulting with exceeding speed and flexity. In addition, Memex also attaches importance to tying two items together, which can refer to building a personal knowledge base. Recently, Gurrin et al. (2014) and Ksibi et al. (2021) explore the research trends, applications and the challenges of lifelogging. Some works have investigated activity capture via SenseCam (Gemmell et al., 2004), Smart glasses (Aiordachioae and Vatavu, 2019), and Go-Pro. In addition, several studies have worked on the lifelogging applications of lifestyle understanding (Doherty et al., 2011), diet monitoring (Maekawa, 2013), and contact tracing (Bengio et al., 2020).

As to the information recall service, there are several works that investigated the reactive mode. Gurrin et al. (2016, 2017, 2019, 2020) introduce visual lifelog retrieval tasks that aims at querying specific moments in a lifelogger's life. Chu et al. (2019) and Chu et al. (2020) construct a multimodal retrieval model that enables users to search their photos with textual queries. Yen et al. (2021b) propose a system to answer the questions about personal life experiences over personal knowledge base. They also tackle the unanswer-

doi:10.6342/NTU202202609

Figure 2.1: Lifelogs from different sourse.



(a) SenseCam

(b) Smart glasses

(c) Go-Pro

(d) Microphones for dictation

(e) Post on social network service

able question caused by the events in the question that are inconsistent with the personal knowledge base facts. They attempt to correct the unanswerable questions to answerable ones. However, the reason why the question is unanswerable is not explicitly explained. In this thesis, we focus on detecting the need for a proactive information recall service along with the support evidences.

2.2 Structured Information

Recently, Convolutional Neural Networks (CNNs) and Transformer-based Network have become popular methods to tackle numerous problems such as image classification (He et al., 2016), object detection (Bochkovskiy et al., 2020; Redmon et al., 2016), and machine reading comprehension (Devlin et al., 2018; Yu et al., 2018). However, many tasks or data cannot be formed into a grid-like structure, which makes them unable to benefit from CNNs or Transformers. One of them is graph structure such as social network, citation network (Leskovec and Krevl, 2014) and protein structure (Borgwardt et al., 2005). To tackle this problem, several graph models are proposed to encode the graphs, GCN (Kipf and Welling, 2016), SAGEConv (Hamilton et al., 2017), and GAT (Veličković et al., 2018). These models compute each node's representation via message passing from the neighbor nodes on the graph to achieve similar behavior to CNNs and Transformer.

Besides those data being original in the graph structure, recent works attempt to construct graphs from the grid-like data. One of the examples is the natural language. More and more approaches aim at integrating the structure information for improving the Nature Language Process (NLP) task. CAKE (Niu et al., 2022) integrates the knowledge graph to automatically extract commonsense from factual triples with entity concepts. TSQA (Shang et al., 2022) efficiently uses the facts contained in a temporal knowledge graph, which records entity relations and when they occur in time, to answer natural language questions. Apart from the knowledge graph, the structured information within the text, such as dependency parsing results, has proved the effectiveness in capturing the contextual interactions. For instance, the model proposed by Gong et al. (2022), BERT4GCN (Xiao et al., 2021), and SGNET (Zhang et al., 2020) integrate the dependency relations to leverage syntactic information. Sun et al. (2022) extract the structured knowledge from scripts and use it to improve machine reading comprehension (MRC) tasks. In addition, recent research proposes different methods to utilize textual and structured information. GreaseLM (Zhang et al., 2022) and LUKE (Yamada et al., 2020) integrate the external knowledge base by fusing token representations and entity representations from the language model and the additional embeddings, respectively. Here, we introduce an event graph into our proposed model, which contains a new fusion mechanism to capture the relations of the life events.

2.3 Nature Language Inference

The initial thought of Nature Language Inference (NLI) can date back to the semantic concept of entailment and contradiction (Katz and Fodor, 1963; Van Benthem, 2008). Using these relations, the NLI task is formed as a generic textual entailment task of determining the inference relation between the given hypothesis and premise (Dagan et al., 2005; MacCartney and Manning, 2008). However, the NLI task is challenging since it contains natural language understanding (NLU) and semantic analysis. Typically, the inference relations in NLI include entailment, contradiction, and neutral. Table 2.1 shows the examples derived from SNLI (Bowman et al., 2015). From the first sentence pair, we can infer that the reason for the contradiction label is that the driving scene is different and in conflict. From the second sentence pair, since the precise entails the hypothesis, the label is entailment.

Table 2.1: Examples of NLI.

Precises	Hypothesis	Label
A black race car starts up in front of a crowd of people.	A man is driving down a lonely road.	contradiction
A soccer game with multiple males playing.	Some men are playing a sport.	entailment

Due to the nature and the importance of NLI, its concept has been extended to many NLP applications. Harabagiu and Hickl (2006) investigate the contextual entailment relation between user scenario, question and the answer to improve a Question-Answering system. Bora-Kathariya and Haribhakta (2018) utilize the nature of NLI to evaluate the quality of abstractive summarization. In addition, many datasets for different purposes are proposed including: large corpus based on image captioning (SNLI Bowman et al., 2015), Multi-Genre Corpus (MultiNLI Williams et al., 2018), explanation for SNLI (e-

doi:10.6342/NTU202202609

SNLI Camburu et al., 2018) and cross-lignual corpus (XNLI Conneau et al., 2018). In this work, we introduce the event type identification task, which is also a sequence pair task. We further experiment on whether integrating the NLI task can improve our performance and analyze the difference between them.

Chapter 3 Dataset Construction

3.1 Hippocorpus

To investigate the need for information recall assistance, we need a dataset consisting of lifelogs of different times. To this end, since there is no dataset available for our purpose, we extend Hippocorpus, which contains the narratives of the event at different times. Sap et al. (2020) constructed Hippocorpus to investigate the difference in the narrative flow between relating life experiences and telling imaginative stories. They collected them from crowdsourcing, but the workers' IDs and names are not included. In other words, the dataset does not contain any personally identifiable information that would infringe on someone's privacy. They defined three story types and collected from three stages. In addition, the workers optionally provide their demographic information(age, gender, etc.) after writing the stories in each stage.

Recalled: Hippocorpus first asked the workers to write a memorable event they experienced in the last six months. The workers also were to write a 2-3 sentence summary of the event for the later stages.

Imagined: A different set of workers were randomly assigned a summary of the event they collected in the previous stage. They then wrote the imagined stories based on the

doi:10.6342/NTU202202609

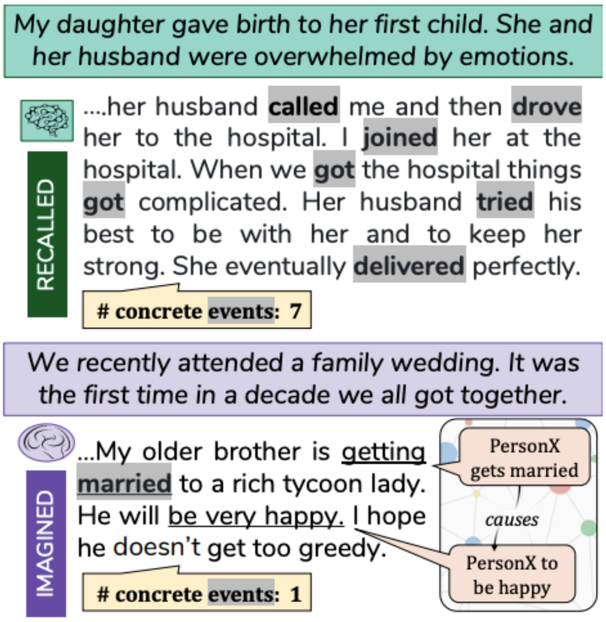

My daughter gave birth to her first child. She and her husband were overwhelmed by emotions.
RECALLED
....her husband called me and then drove her to the hospital. I joined her at the hospital. When we got the hospital things got complicated. Her husband tried his best to be with her and to keep her strong. She eventually delivered perfectly.
# concrete events: 7
We recently attended a family wedding. It was the first time in a decade we all got together.
IMAGINED
...My older brother is getting married to a rich tycoon lady. He will be very happy. I hope he doesn't get too greedy.
# concrete events: 1
PersonX gets married
causes
PersonX to be happy

Figure 3.1: A example of Hippocorpus.

given summaries.

Retold: After 2-3 months, the workers of the recalled stage were to write their stories again with the summary as the prompt.

Finally, Sap et al. (2020) collected 6,894 English diary-like stories, which consist of 2,779 recalled stories, 1,319 retold stories, and 2,756 imagined stories. In this thesis, the life experiences written the first and second time(i.e., Recalled and Retold, respectively) are referred to here as pre-retold and post-retold stories, respectively.

3.2 From Hippocorpus to NIR

In this work, we construct NIR by pruning the imaginative stories in Hippocorpus and retaining those stories about real-life events written by crowd-workers at two different times as pre-retold stories and post-retold stories. Following the four situations mentioned in Section 1.1, we summarize the following five event types from the story pairs in the

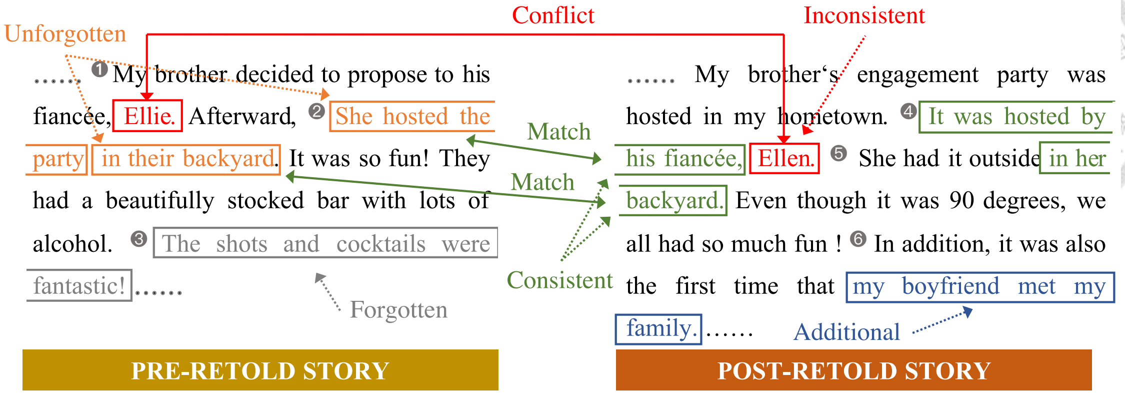

Unforgotten
Conflict
Inconsistent
...... ❶ My brother decided to propose to his fiancée, Ellie. Afterward, ❷ She hosted the party in their backyard. It was so fun! They had a beautifully stocked bar with lots of alcohol. ❸ The shots and cocktails were fantastic! ......
Forgotten
Match
Match
Consistent
...... My brother's engagement party was hosted in my hometown. ❹ It was hosted by his fiancée, Ellen. ❺ She had it outside in her backyard. Even though it was 90 degrees, we all had so much fun ! ❻ In addition, it was also the first time that my boyfriend met my family. ......
Additional
PRE-RETOLD STORY
POST-RETOLD STORY

Figure 3.2: Snippets from two stories in NIR.

dataset: *Consistent*, *Inconsistent*, *Additional*, *Forgotten*, and *Unforgotten*. The first three event types occur in the post-retold stories, and the last two event types occur in the pre-retold stories.

Figure 3.2 shows a pair of pre-retold and post-retold stories labeled with these five event types denoted by green, red, blue, gray, and orange boxes, respectively. The numbers in Figure 3.2 denote the sentences consisting of life events. The details of the five event types are listed as follows:

Consistent: The described event matches the user's life experiences. The event in Sentence (5) is *Consistent* because the event of the brother's fiancée hosting the party in the backyard matches the description in Sentence (2). In this case, the event in Sentence (2) is the support evidence.

Inconsistent: In contrast to *Consistent*, the description is inconsistent with life events. For example, although the description of the fiancée hosting the party matches Sentence (2), her name in the two stories is different. Thus, the event in Sentence (4) is *Inconsistent*. In other words, if the details of the event description in the post-retold story conflict with the facts described in the pre-retold story, it is an *inconsistent* event.

Additional: This is extra information about a life event that is not previously recorded in the collected lifelogs. The event in Sentence (6) is *Additional* due to the lack of similar event in the pre-retold story.

Forgotten: The life events that have been forgotten, i.e., are not mentioned here. As the event in Sentence (3) does not relate to other events in the post-retold story, it is a *Forgotten* event.

Unforgotten: In contrast to *Forgotten*, the life events in the pre-retold story and also mentioned in the post-retold story belong to *Unforgotten* events. As the events in Sentence (2) are also mentioned in the Sentence (4) and Sentence (5) in the post-retold story, they are *Unforgotten* events.

In our dataset, each life event in the pre-retold and post-retold story stories is labeled with one of five event types. The annotation of relevant events within another story of the story pair is also included to denote as support evidence of the event type. That is, we annotate event types and the corresponding support evidence in the pre-retold and post-retold stories. The construction of the dataset is described in Chapter 3.

3.3 Life Event Annotation

According to the definition of LiveKB (Yen et al., 2019, 2020) and ConvLogMiner (Kao et al., 2021), we define a life event as a life experience that is related to specific individuals. Note that a sentence may refer to multiple life events. We follow the work of Yen et al. (2019) to extract life events in the triple form (subject, predicate, object) and annotate each life event with polarity, explicit and implicit. In an explicit event, the predicate can be annotated by directly using the words in the story. In an implicit event, the predicate must be inferred from the context since the action of the event is not mentioned in the story. For implicit predicates, annotators were to choose the proper predicate by consulting FrameNet (Fillmore et al., 2003). Take Figure 3.2 and Table 3.1 as examples, two explicit events (She, hosted, party) and (She, hosted in, backyard) are included in Sentence (2). A single implicit life event (I, drink, the shots and cocktails) is described in Sentence (3).

For the life event annotation, we invited five annotators who majored in linguistics or were English native speakers. Given a story, the annotators were to annotate life events in the story in triple form. To verify the quality of the annotation results, we sampled five stories (i.e., a total of 100 sentences and 129 life events) as reference story and asked a supervisor to label the life events. These stories were also assigned to the other four annotators. Since the three components in the triple were annotated as free text, we joined each component into a sequence. We measured the agreement of each annotator with the supervisor via the Rouge-L (Lin, 2004) and F-scores for the life event triple and the explicitness of the life event, respectively. Here, the reason for utilizing the Rouge-L score to evaluate the agreement of life event triple annotation is that the components in a triple are text spans. We regard the annotation results of the supervisor as the reference to measure the annotation quality of the other annotators. The resulting average agreement of the life event triple and the explicitness of the life event were 0.87 and 0.80, respectively. Finally, we collected 60,889 events from 2,520 stories consisting of 44,199 sentences. The distribution of explicit and implicit events was 96.9% and 3.1%, respectively.

Table 3.1: Examples of life event annotation

Sentence: *She hosted the party in their backyard.*				
Subject	Predicate	Object	Frame	Explicitness
She	hosted	party		Explicit
She	hosted in	their backyard		Explicit
Sentence: *The shots and cocktails were fantastic!*				
Subject	Predicate	Object	Frame	Explicitness
I	drink	shots and cocktails	digest	Implicit

3.4 Event Type Annotation

Given the life event annotation of each sentence, we invited 11 annotators to label the event types of the life events, where the event types are *Consistent*, *Inconsistent*, *Additional*, *Forgotten*, and *Unforgotten*. Given the pairs of pre-retold and post-retold stories, the annotators were invited to first read the stories to understand the author's experiences. For each story pair, one story is viewed as the reference story, and another story is viewed as the target story. The annotators labeled the event type of each life event in the target story by consulting the reference story. The decision of event type is also based on whether the target story is a pre-retold or post-retold story. In addition, for each story pair, they select the life events in one story that are related to the life events in another story as the support evidence for explaining the event type. On the one hand, if the target story is post-retold story, the annotators were to classify the events into *Consistent*, *Inconsistent*, and *Additional*, and select related events from the pre-retold story. On the other hand, once the target story is the pre-retold story, the events need to be classified into *Forgotten* and *Unforgotten*, and the events in the post-story were selected as the support evidence. Taking Figure 3.2 as an example, event (his fiancée Ellen, host, it) in Sentence (4) is *Inconsistent* since the name of the brother's fiancée conflicts with the event (my brother, propose to, his fiancée Ellie) in Sentence (1), although it matches the event (She, hosted, party) in Sentence (2). In other words, to identify *Inconsistent* events, comparing the subtle differences in the descriptions of the pre-retold and post-retold stories is essential.

The examination of the annotation quality proceeds similarly to the method mentioned in Section 3.3. We randomly sampled 50 story pairs (2,113 events in total) and assigned them to each annotator. An annotator who majored in linguistics was selected as the supervisor. We measured the agreement of each annotator with the supervisor via the Cohen's kappa. The average event type agreement was a Cohen's kappa score of 0.95. Finally, we collected 1,260 story pairs, with an event type distribution of *Consistent*, *Inconsistent*, *Additional*, *Forgotten*, and *Unforgotten* events of 11,525, 226, 17,661, 18,773, and 12,704, respectively. The further analyses are described in Chapter 7.

Chapter 4 Methodology

4.1 Task Definition

4.1.1 Need Detection of Information Recall

To detect the need for information recall, we propose a novel task that is aimed at determining the event type by comparing a pair of pre-retold story $\mathcal{U}$ and post-retold story $\mathcal{V}$. This can be considered a multi-class classification. We regard one story as the reference story D and compare the event triple in another story (i.e., the target story) $D^{'}$ with all sentences in D to identify the event type. Formally, given a pair of $\mathcal{U}$ and $\mathcal{V}$, the task is to identify the life event type $y_i^{D^{'}}$ of the i-th event triple e_i in $D^{'}$, where $y_i \in \{$*Consistent*, *Inconsistent*, *Additional*, *Forgotten*, *Unforgotten*$\}$, and D' denotes $\mathcal{U}$ or $\mathcal{V}$. On the one hand, for the task of identifying *Consistent*, *Inconsistent*, and *Additional* events, $D = \mathcal{U}$ and $D^{'} = \mathcal{V}$. On the other hand, for the task of identifying *Forgotten* and *Unforgotten* events, $D = \mathcal{V}$ and $D^{'} = \mathcal{U}$.

4.1.2 Support Evidence Extraction

Apart from the event type identification, a interpretable model makes humans can readily understand the reasoning behind predictions and decisions made by the model.

doi:10.6342/NTU202202609

Thus, to remind the user which event is forgotten or confused in the proactive mode, providing an explanation is beneficial for memory recall. To this end, we also propose an explanation task to extract the events in D that are related to e_i in D' as evidence to explain the decision of event type. The extracted event triple can also help users recall their life experiences.

4.2 Structured Event Enhancement Network

Although the pre-trained language models have shown great success on various NLP tasks, some works (Tang et al., 2020; Wang et al., 2020; Xiao et al., 2021) also suggest that the structured information can enhance token representations. We construct an event graph based on life event triples. The event graph is incorporated into our model for capturing fine-grained information of life event relations within a document. Inspired by GreaseLM (Zhang et al., 2022), which incorporates the language model with the external knowledge graph, we initialize the node representations by using the language model. Specifically, we extract the hidden states from different encoder layers of the language model as the node representations. A GAT model is employed to propagate the structured information of the event graph. Then the updated node representations are used for enhancing the token representations in the language model by our fusion mechanism. Figure 4.1 shows an overview of our proposed structured event enhancement network (SEEN). In addition, since we take the event type identification as a sequence pair task, we further investigated whether our proposal can be benefited from pre-training on NLI dataset. The details are described as follows.

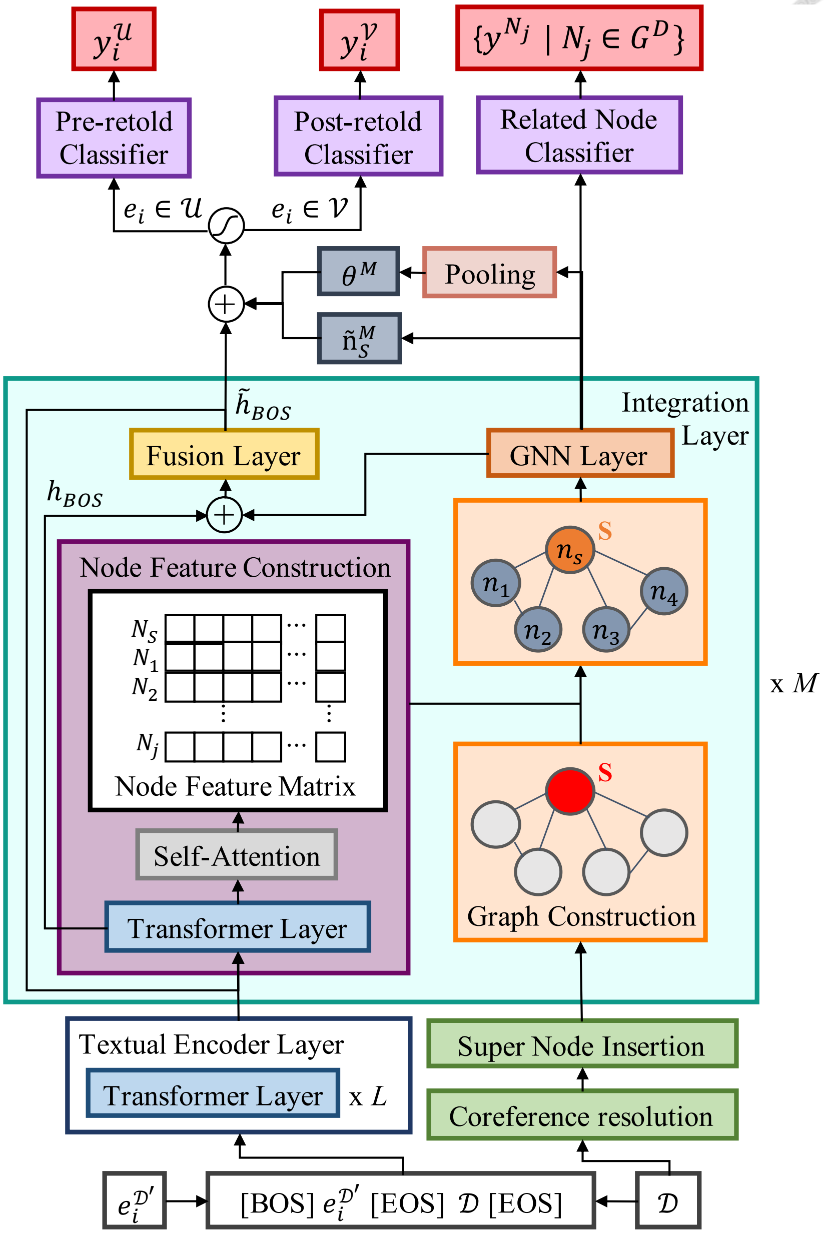
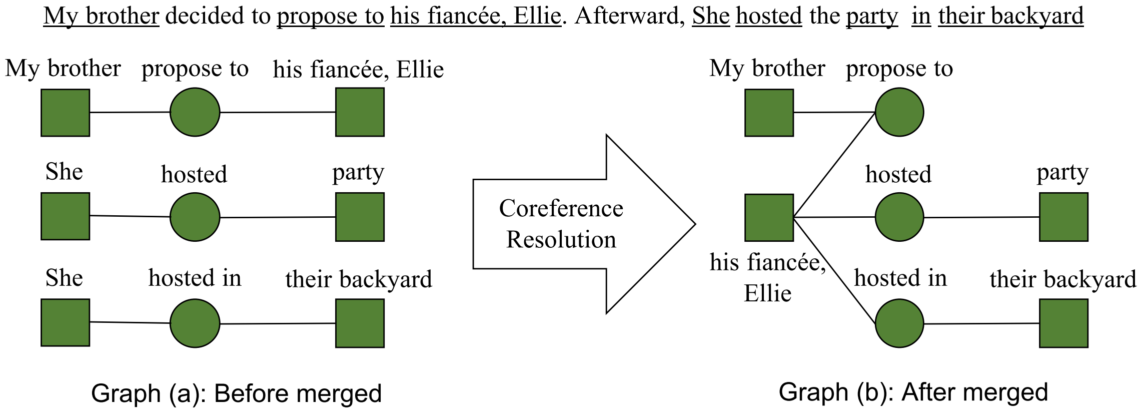


Figure 4.1: Overview of SEEN.

4.2.1 Event Graph Construction

To construct an event graph G^D, we regard subjects, predicates, and objects of all events in reference story D as the nodes. Since some subjects or objects may refer to other nodes, the nodes which are connected with the coreference links are merged as one node. Here, the coreference links are obtained by utilizing the coreference resolution model (Lee et al., 2018). Then, for each life event triple, we connect the predicate nodes to the subject and object nodes to create G^D. Take Figure 4.2 as an example, the words with the underlines represent the subjects, predicates, and objects of the event triples, and Graph (a) and (b) are regarded as the graphs merged before and after. According to the semantics and the result of coreference resolution, we can infer that *She* is the pronoun of *his fiancée, Ellie*. Since the subjects of the second and third event triple (i.e., *She*) and the object of the first event triple(i.e., *his fiancée, Ellie*) are referred to the same entity, we merged these three nodes as one node and take *his fiancée, Ellie* as the text span of the node. In addition, since we construct the graph based on the event triples and the result of the coreference resolution, most nodes only have a few edges connected to the other nodes which makes the connectivity of the graph poor. To enhance the connectivity of G^D, we insert a *Super Node* $\mathcal{S}$ into the graph, and connect it to all the other nodes.

Figure 4.2: An example of the coreference resolution.



4.2.2 Textual Encoder Layer

To encode textual features of reference story D and the i-th event triple e_i in target story D', we concatenate D and $e_i^{D'}$ with the special tokens [BOS] and [EOS]. The format is [BOS] $e_i^{D'}$ [EOS] D [EOS]. For example, if the goal is to identify the type of i-th life event in $\mathcal{V}$, the input sequence is [BOS] $e_i^{\mathcal{V}}$ [EOS] $\mathcal{U}$ [EOS], where $e_i^{\mathcal{V}}$ is the concatenation of the components in the event triple. The output of l-th layer is the hidden states $H^l = \{h^l_{BOS}, h^l_1, ..., h^l_i\}$, where $l = 1, ..., L$. L is a hyperparameter that denotes the number of transformer layers stacked in the textual encoder layer. l = 0 is the initial embedding of the tokens.

4.2.3 Integration Layer

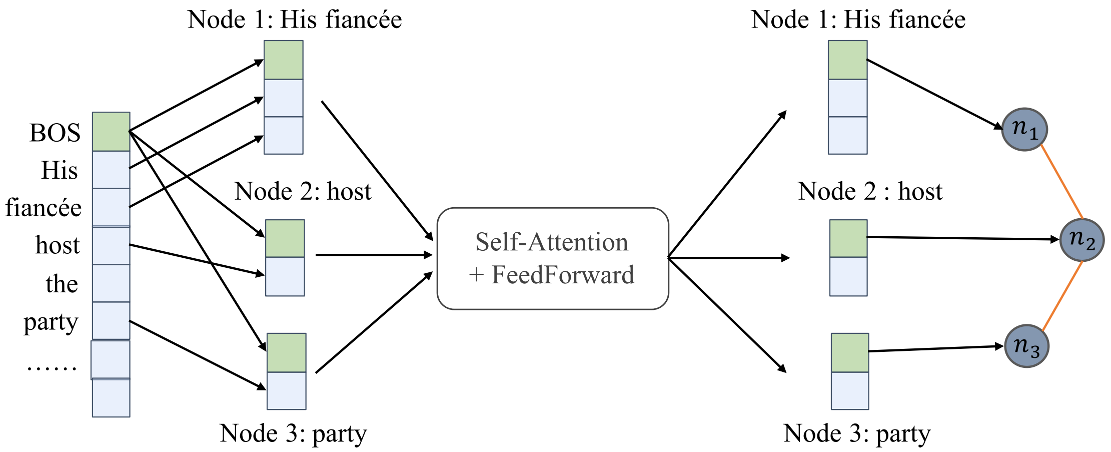
To introduce the structured information of event graph into our language model, we stack M integration layers on the textual encoder layer, where M is a hyperparameter.

Node Feature Construction: Since different layers in the encoder capture different linguistic information for language understanding (Hoover et al., 2020), we initialize the node representations in G^D by using the hidden states of different encoder layers. Each node in G^D is a component in the event triple. Hence, a node can be a text span of the given D. To construct the node feature matrix, we first input H^L into the transformer layer in the integration layer. Then, we construct the initial feature matrix of all nodes in D. Specifically, we extract the hidden states of the `[BOS]` token and the tokens belong to each node. For example, the feature of the j-th node in the m-th integration layer is $[h^m_{BOS}; \|_{t \in T_j} h^m_t]$, where T_j is the token set of the j-th node. Afterward, we concatenate the initial features of each node as the initial feature matrix, and fed the matrix into a self-attention layer. Finally, we take the hidden state of the `[BOS]` token from the self-attention layer's output as the feature of j-th node, which is denote as n^m_j.

$$[n^m_j; \dots] = \text{Attn}([h^m_{BOS}; \|_{t \in T_j} h^m_t]) \cdot W \tag{4.1}$$

Graph Encoder: After initializing the node features, we exploit the GAT layer to encode

Figure 4.3: An example of node feature construction.



the event graph. To learn the representation $\hat{n}_j^m$ of the j-th node N_j from the m-th GAT layer, N_j receives the messages from its neighbor nodes $\mathcal{R}_j$ and computed its feature as Equation 4.2, where $\alpha_{j,j}^m$ and $\alpha_{j,r}^m$ denote the weights of the j-th node and the r-th neighbor node in m-th GAT layer, respectively. And the attention weight is computed by Equation 4.3, where $\alpha_{s,d}^m$ denotes the attention weight of the message between the s-th node and the d-th node. The score $x_{s,d}$ is computed by Equation 4.4. The encoded graph is denoted as $\hat{G}^{D,m} = \{\hat{n}_1^m, ..., \hat{n}_j^m\}$, where $m = 1, ..., M$.

$$\hat{n}_j^m = \alpha_{j,j}^m n_j^m + \sum_{r \in \mathcal{R}_j} \alpha_{j,r}^m n_r^m \tag{4.2}$$

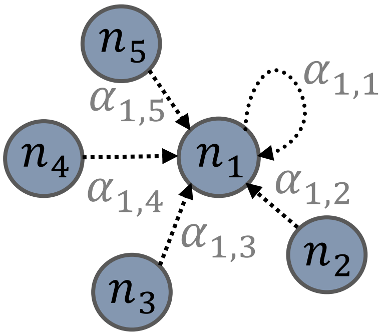
$$\alpha_{s,d}^m = \frac{x_{s,d}^m}{\sum_{k \in \mathcal{N}_s \bigcup \{s\}} x_{s,k}^m} \tag{4.3}$$

$$x_{a,b}^m = \exp(W_\tau \text{LeakyRelu}(W_\kappa \cdot [n_a^m; n_b^m]) \tag{4.4}$$

Fusion Layer: To enhance the language model with the structured information from the event graph, we fuse the hidden state of `[BOS]` token $h_{BOS}^m \in \mathbb{R}^\epsilon$ of the m-th transformer

doi:10.6342/NTU202202609

Figure 4.4: Overview of GAT.



layer and the feature of *Super Node* $\hat{n}_{\mathcal{S}}^{m} \in \mathbb{R}^{\delta}$ in $\hat{G}^{D,m}$, where ϵ and δ are the hidden size of h_{BOS}^{m} and $\hat{n}_{\mathcal{S}}^{m}$, respectively. We concatenate and feed the result into a feedforward network to obtain the integrated feature $z \in \mathbb{R}^{\epsilon+\delta}$. Hence, z is a feature after the fusion of textual and structured information. Afterward, we split z into two parts as the updated features $\tilde{h}_{BOS}^{m} \in \mathbb{R}^{\epsilon}$ and $\tilde{n}_{\mathcal{S}}^{m} \in \mathbb{R}^{\delta}$ of the `[BOS]` token and *Super Node*, respectively.

$$z = \text{GeLu}(W([h_{BOS}^{m}; \hat{n}_{\mathcal{S}}^{m}]) + b) \tag{4.5}$$

4.2.4 Event Type Classifier

After updating the features through M integration layers, the super node's feature $\tilde{n}_{\mathcal{S}}^{M}$ and the mean pooling result θ^{M} of graph $G^{D,M}$ are concatenated with the hidden state of `[BOS]` token to obtain the feature h for the event type identification. We use different classifiers to identify the event type of the event triple from different stories. For the events in $\mathcal{U}$, we use the sigmoid function following a feedforward network ϕ to identify whether it is *Forgotten* or *Unforgotten*. And the loss is denoted as $\lambda_{\mathcal{U}}$. Otherwise, we apply the softmax function following another feedforward network ψ to determine whether the event in $\mathcal{V}$ is *Consistent*, *Inconsistent*, or *Additional*. And the loss is denoted as $\lambda_{\mathcal{V}}$.

$$h = \tilde{h}^{M}_{BOS} \oplus \tilde{n}^{M}_{\mathcal{S}} \oplus \theta^{M} \tag{4.6}$$

$$y^{D'}_{i} = \begin{cases} \text{Sigmoid}(W_{\phi}h + b_{\phi}) & D' = \mathcal{U} \\ \text{Softmax}(W_{\psi}h + b_{\psi}) & D' = \mathcal{V} \end{cases} \tag{4.7}$$

4.2.5 Related Node Classifier

To extract the support evidence, we identify whether the node N_j in G^D is related to $e^{D'}_{i}$. Thus, each node feature is fed into a feedforward network following a sigmoid layer to perform binary classification. Note that *Forgotten* and *Additional* are the events only occurring in D'. The related nodes cannot be found in D. Thus, we exclude these two events to train the related node classifier, and the loss is denoted as $\lambda_{\mathcal{G}}$. Finally, we compute the weighted sum of three losses as shown in Equation 4.9 to update the model, where α and β are 0.5 after tuning by the validation set.

$$y^{N_j} = \text{Sigmoid}(\hat{n}^{M}_{j} \cdot W) \tag{4.8}$$

$$\lambda = \alpha \cdot (\lambda_{\mathcal{U}} + \lambda_{\mathcal{V}}) + \beta \cdot \lambda_{\mathcal{G}} \tag{4.9}$$

4.2.6 Integration with Natural Language Inference

To identify event types, we propose a pilot model to determine the relations between the event in the target story and the reference story. This is different from simply compar-

doi:10.6342/NTU202202609

ing the relation between two sentences in a natural language inference (NLI) task (Bowman et al., 2015; Camburu et al., 2018; Williams et al., 2018). Identifying the event types in narratives involves two main challenges. Firstly, the event type of the event in the target story must be determined by identifying the event pair relations with all life events in the reference story, since the discourse structures in the target story and the reference story are often different. Secondly, the granularity of event descriptions between the stories in a pair can differ, e.g., the number of events that happened, the order of activities, the name of the friend, or the object appearance description. Hence, to determine the event type, we must infer the relevant details of the described events in both stories.

To investigate the difference between NLI and event type identification in information recall assistance, we experiment with the impact of introducing the NLI task into our model. We find that pre-training the language model on the NLI task and fine-tuning the model on our task will improve the performance. However, the label definitions in the NLI task are different from our task. The details are discussed in Section 6.2.

doi:10.6342/NTU202202609

Chapter 5 Experiments

5.1 Baseline Models

Since the stories in our dataset are lengthy, we exploit the models that are capable of encoding the whole story as our baseline models. Below baseline models are trained with the same structure of the event type classifier we mentioned in Section 4.2.4 to fit our task.

XLNet (Yang et al., 2019): XLNet is a sequence-to-sequence autoregressive model that pre-trains with the permutation language modeling task instead of the masked language model task in BERT (Devlin et al., 2018). To determine the event type, XLNet and the following baseline models equipped with the autoregressive decoder, GPT-2, and BART, use the hidden state of the last `[EOS]` token as the input of event type classifiers.

GPT-2 (Radford et al., 2019): In addition to the model equipped the autoencoder, we fine-tune an autoregressive model–GPT-2 on our dataset for event type identification.

BART (Lewis et al., 2019): BART is a sequence-to-sequence model that can encode lengthy documents. Compared with our model only containing the autoencoder, BART consists of an autoencoder and autoregressive decoder.

Longformer (Beltagy et al., 2020): Since the number of the story tokens exceeds 512, we utilize Longformer which is a common language model used to encode long-lengthy documents and consists of an autoencoder.

Longformer with GATs: To further compare SEEN with a model capable of encoding a graph, we simply stack M layers of GAT into Longformer. The final hidden states of Longformer and the GAT layer are concatenated and input to the classifiers to identify the event type. The related node classifier is included.

5.2 Experiment Setup

The story pairs in our dataset are randomly split into training, validation, and test sets by the ratio 8:1:1. In other words, the training, validation, and test sets consist of 1,002, 48,413, and 129 pairs of stories, and 48,413, 5,913, and 6,563 events, respectively. In our experiments, we exploit several language models to evaluate the performance of encoding textual data. To load the pre-trained weight of the language model into our proposed model–SEEN, the sum of L and M is the same as the number of the transformer layers in the original language model. In SEEN, we have experimented with M ranging from 3 to 8. The detail of the comparison is described in Section 6.3. And the contributions of different layers are reported in Section 6.4. Finally, we report the results of setting M as 5 in the following sections. To ensure reliability, we train each model three times with different seeds and report the average performance.

For each hyperparameter trial, we evaluate it on the validation set, and the one with the highest score on the event type identification task will be chosen. Apart from the hyperparameters, we evaluate our methods on the validation set 10 times in each epoch. The one with the highest score will be treated as the final checkpoint and reported its test set performance. The hyperparameters of each model are reported in Table 5.1. In addition, we use eight V100 GPUs to train our models and report the average training time in Table 5.2.

Table 5.1: Hyperparameter of each model.

Hyperparameter	SEEN (BART-large)	SEEN (Longformer-base)	SEEN (Longformer-large)
parameter-size	529M	221M	556M
Number of Integration Layers	5	5	5
Number of attention heads in GNN	16	16	16
Dimension of node feature in GNN	1024	768	1024
Dropout rate in GNN	0.2	0.2	0.2
Learning rate	1e-5	1e-5	1e-5
Number of epoch	5	5	5
Optimizer	AdamW	AdamW	AdamW

Table 5.2: Time consumption to train the models.

Model	Average training time (hr/epoch)
SEEN (BART-large)	0.47
SEEN (Longformer-base)	0.36
SEEN (Longformer-large)	0.87

5.3 Experiment Results

The performance of each model on overall event types is shown in Table 5.3 and the evaluation metrics are reported in macro-averaged. We also report the results of each event type in Table 5.4. The F-score is adopted as the evaluation metric. We calculate McNemar's statistical significance test on the baselines and our models. To verify the effectiveness of the integration layer, we compare the performances of the following three combinations: (1) “BART-large” and “SEEN (BART-large)”. (2) “Longformer-base” and “SEEN (Longformer-base)”. (3) “Longformer-large” and “SEEN (Longformer-large)”. The performances of SEEN in the three combinations outperform the baseline models at $p < 0.01$, $p < 0.05$, and $p < 0.01$, respectively. The results show the adaptability of the integration layer to different language models.

Table 5.3: The overall performance of detecting information recall needs.

Model	F-score	Precision	Recall
XLNet	0.6062	0.6164	0.6082
DistilGPT2	0.5313	0.5361	0.5350
GPT2	0.5857	0.5870	0.5885
GPT2-Medium	0.6024	0.6022	0.6037
GPT2-large	0.6025	0.6020	0.6034
Bart-base	0.6065	0.6057	0.6143
Bart-large	0.6369	0.6517	0.6385
Longformer	0.6183	0.6181	0.6192
Longformer-large	0.6334	0.6331	0.6362
Longformer-large w/ GATs	0.6531	0.6628	0.6497
SEEN (Bart-large)	0.6384	0.6363	0.6412
SEEN (Longformer-base)	0.6403	0.6414	0.6477
SEEN (Longformer-large)	**0.6654**	**0.6781**	**0.6607**

Table 5.4: Experimental results of each event type of detecting information recall needs.

Model	Consistent	Inconsistent	Additional	Forgotten	Unforgotten
XLNet	0.7076	0.0238	0.7911	0.7925	0.7159
DistilGPT2	0.6115	0.0000	0.6918	0.7380	0.6153
GPT2	0.6775	0.0000	0.7710	0.7819	0.6981
GPT2-Medium	0.6997	0.0000	0.8038	0.7963	0.7124
GPT2-large	0.6999	0.0000	0.8063	0.7955	0.7107
Bart-base	0.7168	0.0128	0.7832	0.7851	0.7346
Bart-large	0.7582	0.0247	0.8324	0.8135	0.7555
Longformer	0.7340	0.0000	0.8256	0.8081	0.7237
Longformer-large	0.7462	0.0142	0.8315	0.8221	0.7529
Longformer-large w/ GATs	0.7472	0.1095	0.8337	0.8158	0.7591
SEEN (Bart-large)	0.7623	0.0000	0.8385	**0.8268**	0.7641
SEEN (Longformer-base)	0.7379	0.0550	0.8183	0.8120	0.7471
SEEN (Longformer-large)	**0.7633**	**0.1313**	**0.8411**	0.8262	**0.7653**

We find that "Longformer-large w/ GATs" significantly outperforms all the other baselines. That means incorporating the event graph is able to encode event relations to improve the performance. In addition, training the task of support evidence extraction simultaneously benefits the performance of event type identification. Moreover, "SEEN (Longofrmer-large)" outperforms "Longformer-large w/ GATs", suggesting that our proposed fusion mechanism introduces structured information effectively to enhance the language model. Comparing the last three rows, the Longformer-based encoder is better than the BART-based, and "SEEN (Longofrmer-large)" achieves the highest overall performance. The reason may be that the integration layers are built on the encoder layer. Identifying the event types by exploiting the output of the hidden states from the integration layer connected with the autoencoder is more suitable for our task. While the prediction of "SEEN (BART-large)" is based on the hidden states output from the autoregressive decoder. Note that all the models achieve relatively lower scores on *Inconsistent* type because the number of this event is sparse in NIR. Besides, we find that "SEEN (Longofrmer-large)" usually identifies *Inconsistent* as *Additional*. The further error analysis is shown in Section 6.6.

Table 5.5: Results of support evidence extraction task.

Model	F-score	Precision	Recall
Longformer-large w/ GATs	0.7289	0.7360	0.8304
SEEN (Longformer-large)	**0.7888**	**0.7775**	**0.8883**

To verify the impact of the integration layer on the support evidence extraction task, we compare our proposed model SEEN with "Longformer-large w/ GATs". The evaluation metric is macro-averaged F-score. As mentioned in Section 4.2.5, we only extract the related nodes of *Unforgotten*, *Consistent*, and *Inconsistent* events. In Table 5.5, SEEN outperforms "Longformer-large w/ GATs". That means fusing the textual and structured information improves the related node selection.

Chapter 6 Analysis and Discussion

6.1 Ablation Study

In this section, we perform an ablation study to analyze the impact of SEEN with different settings.

w/o pre-training on NLI: We introduce the NLI task to strengthen the ability of our language model on capturing semantic features to infer the consistency of event descriptions. Hence, we investigate the influence of pre-training the language model on the Multi-NLI (Williams et al., 2018) dataset.

w/o Concat: Instead of concatenating the final hidden state of the super node and the average of node representations, we only use the hidden state of `[BOS]` as the input of the event type classifier to evaluate the importance of structured features.

w/o Support Evidence Extraction: To analyze the impact of extracting support evidence toward the event type identification, we construct a classifier to extract related nodes in the event graph as evidence for explaining the event type predictions.

w/o Event Graph: To investigate whether the structured event information is beneficial for capturing the fine-grained relations between life events, we analyze the impact of with or without event graphs on the task of detecting information recall needs. Specifically, "SEEN w/o Event Graph" is the alias of the baseline model "Longformer", which does not encode the event graph.

Table 6.1: Ablation study of SEEN.

Model	F-score	Precision	Recall
SEEN (Longformer-large)	**0.6654**	**0.6781**	**0.6607**
w/o pre-training on NLI	0.6488	0.6532	0.6465
w/o Concat	0.6408	0.6574	0.6420
w/o Support Evidence Extraction	0.6349	0.6415	0.6335
w/o Event Graph	0.6334	0.6331	0.6362

The ablation study results are shown in Table 6.1. We find that the performance degrades the most when the event graph is excluded, suggesting that enhancing the language model with the structured event information benefits the event type identification results. In addition, introducing a graph-related task, support evidence extraction, into SEEN strengthens the ability of the graph neural network as well as assists the model in detecting information recall needs. Furthermore, pre-training on the Multi-NLI dataset and fine-tuning on our NIR dataset is also beneficial for identifying the semantic relatedness between the event and the reference story. We further perform an experiment to analyze the relevance between the NLI task and the task of detecting information recall needs in Section 6.2.

6.2 Impact of Pre-training Task

To further compare the event type identification task with the NLI task, we experiment the different pre-training task settings. In other words, we note that the labels are different between the Multi-NLI dataset and our NIR dataset. Therefore, we align the *entailment*, *contradiction*, and *neutral* in Multi-NLI with *Consistent*, *Inconsistent*, and

Additional in post-retold of NIR, respectively. We report the overall macro-averaged F-score and F-score of each individual label. The first two columns denote whether the methods are trained on the Multi-NLI dataset and fine-tuning on the NIR dataset to identify the event types in the post-retold stories. As shown in Table 6.2, the method only trained on the Multi-NLI dataset does not work well in detecting information recall needs. That means the label definitions between NLI and NIR are marginal different, especially the *Consistent* events. SEEN trained on both datasets achieves the highest performance. It means pre-training on the NLI task helps the model better capture semantic relatedness between two descriptions.

Table 6.2: Results of different pre-training task settings.

Pre-training on Multi-NLI	Fine-tuning on Post-Retold Events	Overall	Consistent	Inconsistent	Additional
v		0.4075	0.4715	0.0397	0.7113
	v	0.5480	**0.7556**	0.0625	0.8260
v	v	**0.5572**	0.7512	**0.0837**	**0.8367**

6.3 Number of Integration Layers

We further compare the performance of SEEN with the different numbers of the integration layers. Experimental results shown in Table 6.3. We find that SEEN with five integration layers ($M = 5$) achieves the highest performance, which is the same as the result of GreaseLM. However, different from GreaseLM, there is no consistency in performance changes while M decreases or increases. We think the reason is that the way SEEN fuses textual and structured features are by iteratively initializing the node representations with the updated token representations in each integration layer (The process is described in Section 4.2.3). While GreaseLM utilizes additional node embeddings as node representations, and concatenates the parts of hidden states from the language model and the node embeddings without re-initializing the node representations. To compare SEEN with GreaseLM, we re-implement GreaseLM with the best setting, which is referred to as GreaseLM-like (Longformer-large). The slight difference is that node representations are built from language models. Since most of the nodes in our event graph are text spans, we cannot leverage existing node embeddings. Table 6.4 reports the comparison of SEEN and the GreaseLM-like model. The result shows that SEEN outperforms the GreaseLM-like model. That means the robustness of our proposed integration layer.

Table 6.3: Performance of different number of the integration layer.

# of Integration layer(M)	F-score	Precision	Recall
M = 3	0.6559	0.6601	0.6563
M = 4	0.6470	0.6482	0.6481
M = 5	**0.6616**	**0.6781**	0.6607
M = 6	0.6568	0.6725	0.6556
M = 7	0.6414	0.6682	0.6410
M = 8	0.6597	0.6633	**0.6659**

Table 6.4: Comparison of the GreaseLM-like model and SEEN.

Model	F-score	Precision	Recall
GreaseLM-like(Longformer-large)	0.6417	0.6500	0.6417
SEEN (Longformer-large)	**0.6654**	**0.6781**	**0.6607**

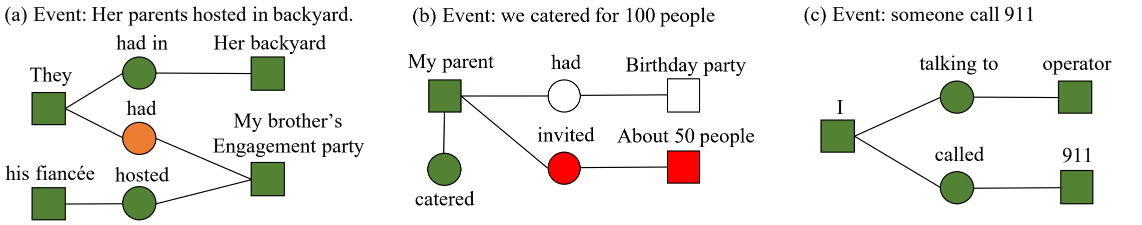
6.4 Contribution of Different Fusion Layers

To investigate the contribution of each fusion layer in SEEN, we compute the distribution of the edge weights between nodes in the GAT layer. We denote the edges connecting to the related node and the unrelated node as E_{RN_+} and E_{RN_-}, respectively. To show the difference between the edge weights, we tell whether the edge weights are higher than the threshold (0.5). If the edge weight is higher than the threshold, the edge is denoted as the positive case as "triggered edges". In the first GAT layer, the distributions of edge weights are relatively average. That leads to none of the edges is triggered edge. This might be that the first GAT layer attempts to capture structured information of the whole event graph by gathering the messages from the neighbor nodes. In contrast, 6.74% edges are triggered edges in the last GAT layer, which is much more than those in the first layer. We further compare the triggered edge distribution of E_{RN_+} and E_{RN_-}, which are 14.93% and 4.92% in the last GAT layer, respectively. That is, compared with the first GAT layer, the last GAT layer in the integration layer aims to focus on the information related to the event $e_i^{D'}$.

Table 6.5: The distribution of the triggered edge.

Layer	Distibution of the trigged edges	The trigged edges distibution of E_{RN_+}	The trigged edges distibution of E_{RN_-}
m=1	0.0000	0.0000	0.0000
m=5	**0.0674**	**0.1493**	**0.0492**

6.5 Case Study of Support Evidence Extraction


(a) Event: Her parents hosted in backyard.
had in
Her backyard
They
had
My brother's Engagement party
his fiancée
hosted
(b) Event: we catered for 100 people
My parent
had
Birthday party
invited
About 50 people
catered
(c) Event: someone call 911
talking to
operator
I
called
911

Label: **Inconsistent**; Predict: **Inconsistent**

Label: **Inconsistent**; Predict: **Additional**

Label: **Inconsistent**; Predict: **Consistent**

Figure 6.1: The examples of the support evidence extraction task.

To investigate the result of the support evidence extraction task, we perform the case study and plot the selected nodes as shown in Figure 6.1. The nodes in the circle and square shapes are predicates, and entities (i.e., subjects or objects), respectively. The green nodes are the correct selections, the red nodes are ground truth but not selected, and the orange nodes are selected nodes but not ground truth. Case (a) is an *Inconsistent* event since the host of the party described in the event sequence and the event graph (constructed from the reference story) are different. In this case, most of the selected nodes are correct, which are related to the described event and can explain why the event is inconsistent. In contrast to case (a), case (b) fails to select the related nodes and the prediction of the event type is also incorrect. Although the model selects all related nodes in case (c) correctly, the prediction of the event type is wrong. Here, the caller of 911 is the author, not the others, while SEEN classifies the *Inconsistent* event as *Additional*. Note that even though case (c) shows the event type identification is incorrect, SEEN is still capable of reminding the user that the event is forgotten or confused by providing the related nodes. In this way, SEEN can proactively provide information recall assistance.

6.6 Error Analysis

To investigate the performance of SEEN on each event type, Figure 6.2 shows the confusion matrix of our model in predicting *Consistent*, *Inconsistent*, and *Additional*. We find that SEEN predicts most *Inconsistent* events as *Additional* events. Firstly, although people often mix their experiences, we tend to avoid unclear events while writing, which results in the rareness of the *Inconsistent* event in our datasets. Apart from the problem of limited training data, this may be because determining that the described event conflicts with established facts require further reasoning on details such as the number of events that occurred, the order of activities, the friend's name, or the object description. Furthermore, since both *Inconsistent* and *Additional* cannot be found in the story context, it is more difficult to classify the event between these two types, which may cause misclassifying *Inconsistent* event as *Additional* event.

doi:10.6342/NTU202202609

In addition, there are also some *Additional* events being identified as *Consistent* events. Since definition of an *Additional* event is an event containing the extra information not mentioned in the reference story, including the fine-grained information, it leads to difficulty identifying the *Additional* event. For instance, there is an event "I go to the hospital with my friends and my mom." and a description in the reference story "I go to the hospital with my mom." The event type of the event is *Additional* since the author only mentioned that he/she went to the hospital with her/his mom but not his friends. That is, the model will fail to identify the *Additional* event once the model can not capture the additional information such as "his friends" in the above example.

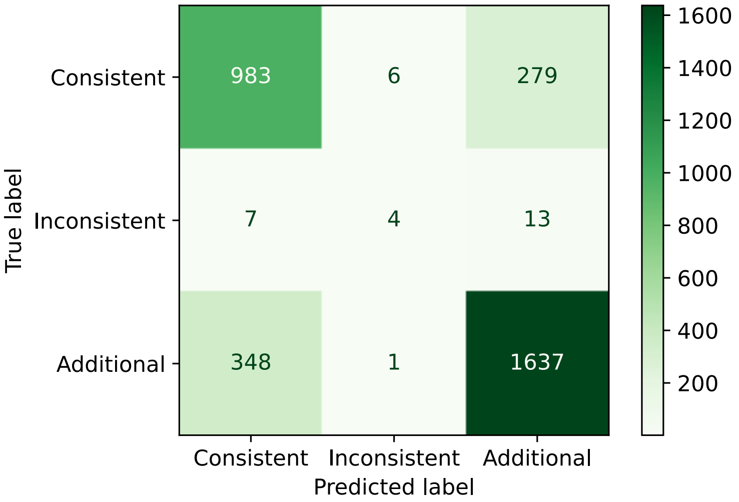


Figure 6.2: Confusion matrix of "SEEN (Longformer-large)" for event type identification.

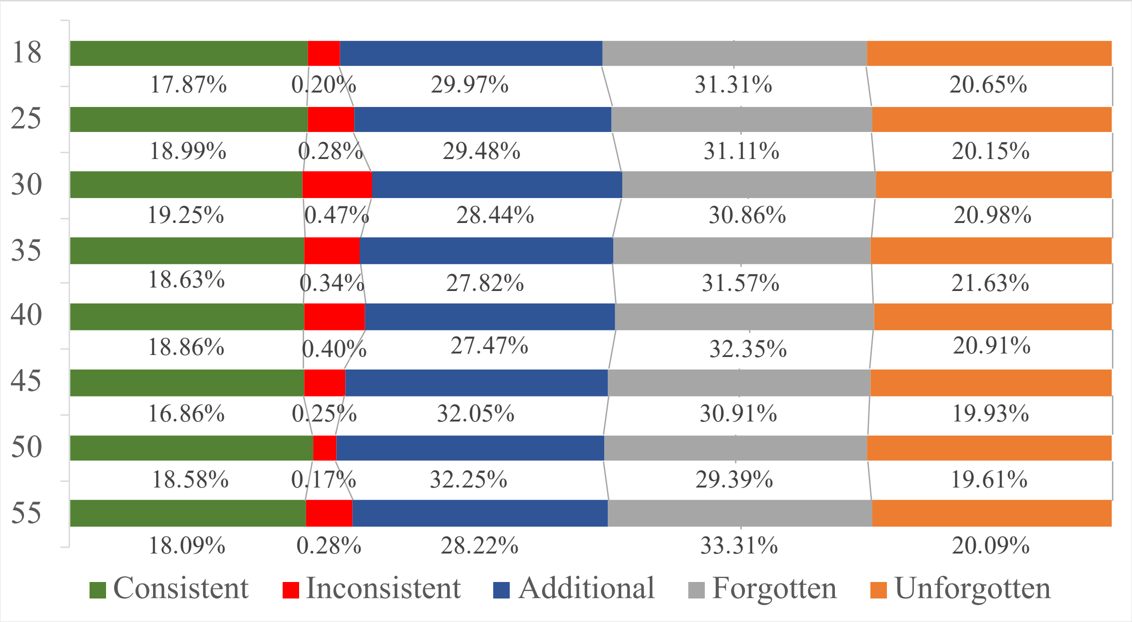
doi:10.6342/NTU202202609

Chapter 7 Data Analysis

7.1 Event Type Analysis on Age

In general, older people are assumed to need more memory assistance because of physical aging which makes people consider that they are more likely to forget things than younger people. To examine whether elders are indeed more likely to forget or confuse their past experiences, we calculated the average ratio of the five event types in each story pair over eight age groups. In Hippocorpus, 82, 214, 281, 208, 133, 117, 83, and 133 crowd-workers were 18, 25, 30, 35, 40, 45, 50, and 55 years old, respectively. The ratio of each event type in each age group is shown in Figure 7.1. For better visualization, the bars are presented using smoothed and normalized ratios, whereas the numbers under the bars are the average distribution of each event type. The ratio of the *Forgotten* events is similar across all age groups, suggesting that both older people and younger people require information recall support. Hereafter, we view people over or equal to 50-years old as the 50-and-above group; those younger than 50-years old are the below-50 group. Comparing the ratio of *Inconsistent* events between the 50-and-above group and the below-50 group, those in the latter group were more likely to confuse life events, where the difference was statistically significant (t-test, $p < 0.05$). This suggests that when younger people recall past experiences, they often confuse details. However, when writing post-retold stories, people in the 50-and-above group preferred not to mention events of which they had only vague impressions.

Figure 7.1: The ratio of each event type in each age group.



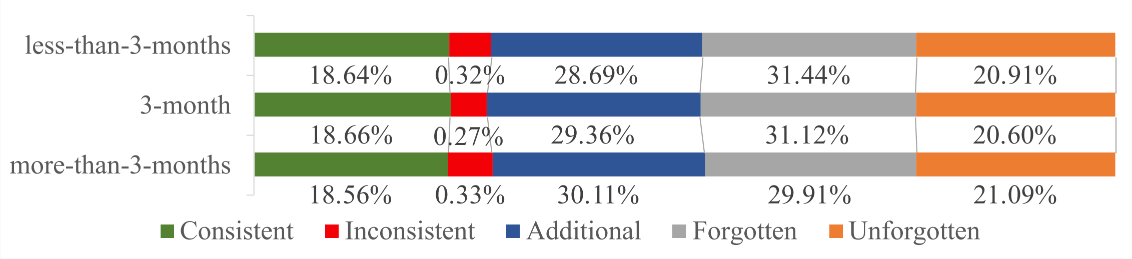
7.2 Event Type Analysis on Time Interval

Typically, people think the most influential factor of memory is time. For instance, we usually consider that people forget or mix up detail as time pass. To examine whether time influence people to recall life experience, we compare the event type distribution of different time interval between the two writing. Since Hippocorpus collected the post-retold story 2-3 months later after they wrote the pre-retold story, we separate each story pair into three groups, time interval less than three months as “less-than-3-months”, the time interval equal three months as “3-months”, and the time interval more than three months as “more-than-3-months”. In Hippocorpus, 616, 115, and 256 story pairs belong to “less-than-3-months”, “3-months”, and “more-than-3-months”, respectively. The ratio of each event type in each time interval group is shown in Figure 7.2. Surprisingly, the

doi:10.6342/NTU202202609

"more-than-3-months" group's *Forgotten* and *Consistent* event ratios are relatively lower than other groups. For this phenomenon, we thought it might relate to the increase of *Additional* events since its *Additional* event ratio is the highest among others. That is, we tend to write the additional event, which makes the ratio of *Additional* events higher.

Figure 7.2: The ratio of each event type in time interval group.

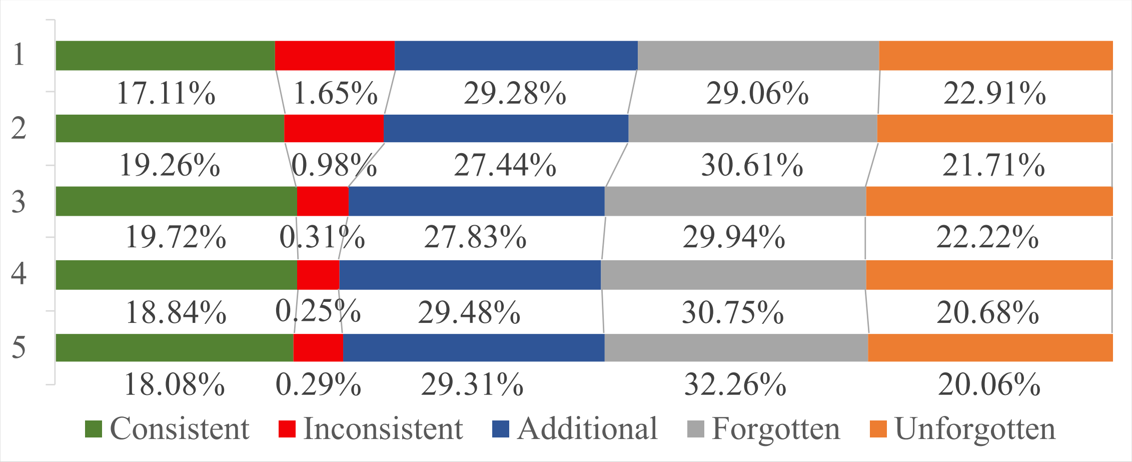


7.3 Event Type Analysis on importance

Apart from the age and time, the importance of the event also influences us when recalling it. In general, important events are assumed to be impressive, making us remember longer. For instance, people usually remember more details of impressive events, such as the wedding decoration and the process of giving birth. Otherwise, the less important events, such as the meeting in the office and the routine, might be forgotten quickly, or we are often confused about the detail when recalling them. To this end, we utilize the importance score the Hippocorpus's workers provided, and the 5-point Likert was used. In Hippocorpus, 14, 57, 147, 356, and 686 stories scored 1, 2, 3, 4, and 5, respectively. The event type ratio of different score stories is shown in Figure 7.3. It is evident that the ratio of *Inconsistent* events is highly related to the importance score. That is, when people recall non-important events, we usually mix them up with other similar experiences, which is consistent with our assumption.

doi:10.6342/NTU202202609

Figure 7.3: The ratio of each event type in time interval group.



7.4 Event Type Analysis on Ownership

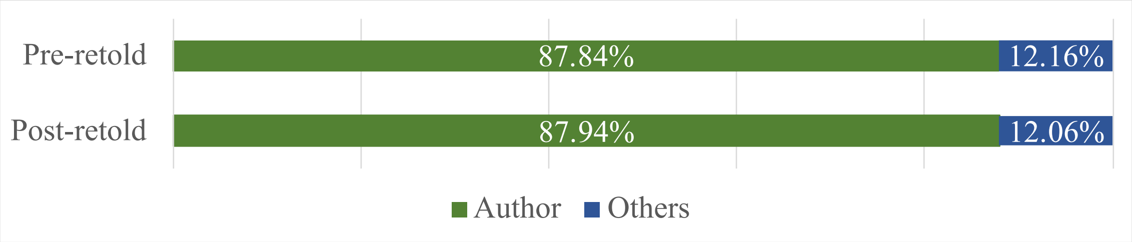
Note that people recall not only their life events but also events involving family, friends, and acquaintances. We further investigated whether people tended to remember their own experiences better than those of others. At the current stage, as events are not labeled to indicate to whom the event belongs, we classified events that do not contain the words “I”, “me”, “we”, or “us” in the subject or object as life events belonging to others. Otherwise, the event was taken to be a life event of the author.

Firstly, we report the ownership ratio of different story types in Figure 7.4. It is obvious that the ratios are similar, and the events of the authors are significantly more than those belonged to the others. In addition, we assume that people tend to write down the events they certainly remembered instead of the unclear one. Thus, we can infer that people usually recalled the events or the experience related to themselves instead of the others, whether they wrote the story at the first time or the second time.

Secondly, we analyze the distribution of each event type on people recalling their own and others’ events. The result is shown in Figure 7.5. We find that the ratio of *Inconsistent*

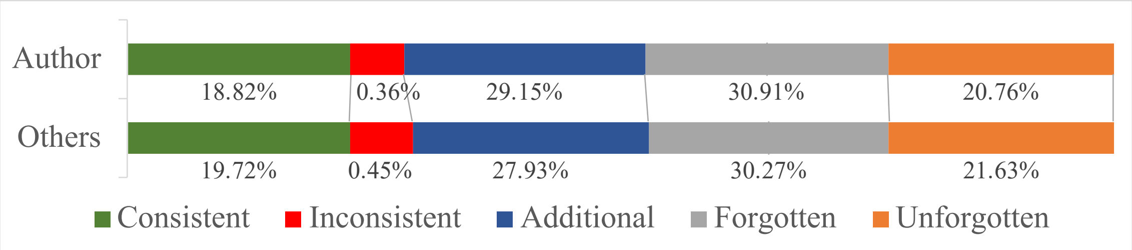
doi:10.6342/NTU202202609

Figure 7.4: The ratio of people recalling their own and others' events of different story types.



events in recalling other people's life events is higher than that when recalling their own life events. The ratio of *Additional* events is also lower when recalling other people's life events: when people write a retold story, they describe only those life events of others that they remember. Hence, when people describe life events again in a post-retold story, they rarely mention new life events about others. However, as people do not remember the life events of others as clearly as their own, they are more prone to confusing such life events.

Figure 7.5: The ratio of event type on people recalling their own and others' events.



doi:10.6342/NTU202202609

Chapter 8 Conclusion

Information recall has attracted much attention in recent years. In contrast to previous studies, we present the task of proactive information recall support and construct NIR, the first human-annotated dataset, to investigate the need for information recall. In this work, we seek to detect event relations between life experiences retold at different times, and identify five event types to determine the time to trigger information recall. To identify the event types for information recall assistance, a pilot model–structured event enhancement network (SEEN) is proposed. We construct an integration layer to fuse the structured information from the event graph into textual representations. In addition, SEEN provides the support evidence to the events by selecting the related nodes in the event graph. Users can consult the explanation to recall their past experiences.

However, the number of *Inconsistent* events is relatively lower in our dataset due to the human writing habit of avoiding uncertain events. In other words, when writing a diary, we always write the ones we exactly remember, which leads to difficulty collecting *Inconsistent* events and training the model to identify *Inconsistent* events. In addition, consulting only one document that describes personal life experiences is not enough to identify the need for information recall assistance in the real-world application. However, the dataset that can be applied to investigate the issue of detecting information recall needs is hard to collect. On the other hand, although our NIR dataset provides two versions of stories of the same events written at different times, we still cannot confirm which story, the previous one or the latter, is correct when contradictory; These limitations are left as future work. We also plan to investigate the time information of the life event and construct an end-to-end system to extract life events in narratives and provide proactive information recall support.

doi:10.6342/NTU202202609

References

Adrian Aiordachioae and Radu-Daniel Vatavu. 2019. Life-tags: a smartglasses-based system for recording and abstracting life with tag clouds. Proceedings of the ACM on human-computer interaction, 3(EICS):1–22.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. arXiv:2004.05150.

Yoshua Bengio, R. Janda, Y. W. Yu, Daphne Ippolito, Max Jarvie, D. Pilat, Brooke Struck, Sekoul Krastev, and A. Sharma. 2020. The need for privacy with public digital contact tracing during the covid-19 pandemic. The Lancet. Digital Health, 2:e342 – e344.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.

Rajeshree Bora-Kathariya and Yashodhara Haribhakta. 2018. Natural language inference as an evaluation measure for abstractive summarization. In 2018 4th International Conference for Convergence in Technology (I2CT), pages 1–4.

Karsten M Borgwardt, Cheng Soon Ong, Stefan Schönauer, SVN Vishwanathan, Alex J Smola, and Hans-Peter Kriegel. 2005. Protein function prediction via graph kernels. Bioinformatics, 21(suppl_1):i47–i56.

doi:10.6342/NTU202202609

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics.

Shaked Brody, Uri Alon, and Eran Yahav. 2022. How attentive are graph attention networks? In International Conference on Learning Representations.

Vannevar Bush et al. 1945. As we may think. The atlantic monthly, 176(1):101–108.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems 31, pages 9539–9549. Curran Associates, Inc.

Tai-Te Chu, Yi-Ting Liu, Chia-Chung Chang, An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2020. Nlp301 at the ntcir-15 micro-activity retrieval task: incorporating region of interest features into supervised encoder. In Proceedings of the NTCIR-15 Conference.

Tzu-Hsuan Chu, Hen-Hsen Huang, and Hsin-Hsi Chen. 2019. Image recall on image-text intertwined lifelogs. In 2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI), pages 398–402. IEEE.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics.

doi:10.6342/NTU202202609

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In Machine learning challenges workshop, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Aiden R Doherty, Niamh Caprani, Vaiva Kalnikaite, Cathal Gurrin, Alan F Smeaton, Noel E O'Connor, et al. 2011. Passively recognising human activities through lifelogging. Computers in Human Behavior, 27(5):1948–1958.

Charles J Fillmore, Miriam RL Petruck, Josef Ruppenhofer, and Abby Wright. 2003. Framenet in action: The case of attaching. International Journal of Lexicography, 16(3):297–332.

Jim Gemmell, Lyndsay Williams, Ken Wood, Roger Lueder, and Gordon Bell. 2004. Passive capture and ensuing issues for a personal lifetime store. In Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences, pages 48–55.

Zheng Gong, Kun Zhou, Wayne Xin Zhao, Jing Sha, Shijin Wang, and Ji-Rong Wen. 2022. Continual pre-training of language models for math problem understanding with syntax-aware memory network. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 5923–5933.

Cathal Gurrin, Hideo Joho, Frank Hopfgartner, Liting Zhou, and Rami Albatal. 2016. NTCIR Lifelog: The first test collection for lifelog research. In Proceedings of the 39th

doi:10.6342/NTU202202609

International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 705–708.

Cathal Gurrin, Hideo Joho, Frank Hopfgartner, Liting Zhou, Rashmi Gupta, Rami Albatal, and Duc Tien Dang Nguyen. 2017. Overview of ntcir-13 Lifelog-2 task.

Cathal Gurrin, Hideo Joho, Frank Hopfgartner, Liting Zhou, V-T Ninh, T-K Le, Rami Albatal, D-T Dang-Nguyen, and Graham Healy. 2019. Overview of the ntcir-14 Lifelog-3 task. In Proceedings of the 14th NTCIR Conference, pages 14–26. NII.

Cathal Gurrin, Tu-Khiem Le, Van-Tu Ninh, Duc-Tien Dang-Nguyen, Björn Þór Jónsson, Jakub Lokoš, Wolfgang Hürst, Minh-Triet Tran, and Klaus Schoeffmann. 2020. Introduction to the Third Annual Lifelog Search Challenge (lsc'20). In Proceedings of the 2020 International Conference on Multimedia Retrieval, pages 584–585.

Cathal Gurrin, Alan F Smeaton, Aiden R Doherty, et al. 2014. Lifelogging: Personal big data. Foundations and Trends® in information retrieval, 8(1):1–125.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. Advances in neural information processing systems, 30.

Sanda Harabagiu and Andrew Hickl. 2006. Using scenario knowledge in automatic question answering. In Proceedings of the Workshop on Task-Focused Summarization and Question Answering, pages 32–39, Sydney, Australia. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778.

Benjamin Hoover, Hendrik Strobelt, and Sebastian Gehrmann. 2020. exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 187–196, Online. Association for Computational Linguistics.

Pei-Wei Kao, An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2021. Convlogminer: A real-time conversational lifelog miner. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, pages 4992–4995. International Joint Conferences on Artificial Intelligence Organization. Demo Track.

Jerrold J Katz and Jerry A Fodor. 1963. The structure of a semantic theory. language, 39(2):170–210.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Amel Ksibi, Ala Saleh D Alluhaidan, Amina Salhi, and Sahar A El-Rahman. 2021. Overview of lifelogging: current challenges and advances. IEEE Access, 9:62630–62641.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), pages 687–692, New Orleans, Louisiana. Association for Computational Linguistics.

Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data.

doi:10.6342/NTU202202609

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 521–528, Manchester, UK. Coling 2008 Organizing Committee.

Takuya Maekawa. 2013. A sensor device for automatic food lifelogging that is embedded in home ceiling light: A preliminary investigation. In 2013 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops, pages 405–407. IEEE.

Guanglin Niu, Bo Li, Yongfei Zhang, and Shiliang Pu. 2022. Cake: A scalable commonsense-aware framework for multi-view knowledge graph completion. In ACL.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 779–788.

doi:10.6342/NTU202202609

Maarten Sap, Eric Horvitz, Yejin Choi, Noah A. Smith, and James Pennebaker. 2020. Recollection versus imagination: Exploring human memory and cognition via neural language models. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 1970–1978, Online. Association for Computational Linguistics.

Chao Shang, Guangtao Wang, Peng Qi, and Jing Huang. 2022. Improving time sensitivity for question answering over temporal knowledge graphs. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8017–8026, Dublin, Ireland. Association for Computational Linguistics.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, and Claire Cardie. 2022. Improving machine reading comprehension with contextualized commonsense knowledge. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8736–8747, Dublin, Ireland. Association for Computational Linguistics.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6578–6588, Online. Association for Computational Linguistics.

Johan Van Benthem. 2008. A brief history of natural logic.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. International Conference on Learning Representations.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 3229–3238, Online. Association for Computational Linguistics.

Yu-Wun Wang, Hen-Hsen Huang, Kuan-Yu Chen, and Hsin-Hsi Chen. 2018. Discourse marker detection for hesitation events on mandarin conversation. In Interspeech, pages 1721–1725.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1112–1122. Association for Computational Linguistics.

Zeguan Xiao, Jiarun Wu, Qingliang Chen, and Congjian Deng. 2021. BERT4GCN: Using BERT intermediate layers to augment GCN for aspect-based sentiment classification. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9193–9200, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep contextualized entity representations with entity-aware self-attention. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. Advances in neural information processing systems, 32.

doi:10.6342/NTU202202609

An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2019. Personal knowledge base construction from text-based lifelogs. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 185–194.

An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2020. Multimodal joint learning for personal knowledge base construction from twitter-based lifelogs. Information Processing & Management, 57(6):102148.

An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2021a. Ten questions in lifelog mining and information recall. In Proceedings of the 2021 International Conference on Multimedia Retrieval, pages 511–518.

An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2021b. Unanswerable question correction in question answering over personal knowledge base. In Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21).

Adams Wei Yu, David Dohan, Thang Luong, Rui Zhao, Kai Chen, and Quoc Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. GreaseLM: Graph REASoning enhanced language models. In International Conference on Learning Representations.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. 2020. Sg-net: Syntax-guided machine reading comprehension. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 9636–9643.