A SEARCH ENGINE USING GRAPH-BASED STRUCTURES FOR LIFELOG RETRIEVAL

Manh-Duy Nguyen, B.Sc.

A Dissertation submitted in fulfillment of the

requirements for the award of

Doctor of Philosophy (Ph.D.)

to the



Dublin City University

Faculty of Engineering and Computing, School of Computing

Supervisor

Prof. Cathal Gurrin

July 2024

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Sign:  Student No.: 19214315 Date: 07/07/2024

(Manh-Duy Nguyen)

Acknowledgements

I would like to express my heartfelt gratitude to the many individuals and institutions that have contributed to the successful completion of this dissertation.

First and foremost, I am deeply indebted to my dissertation supervisors, Prof. Cathal Gurrin and Assoc. Prof. Binh Thanh Nguyen, whose unwavering support, guidance, and expertise were instrumental throughout this research journey. Their mentorship has been invaluable, and I am truly grateful for their dedication to my academic growth.

I am also thankful to Dublin City University for providing the resources and facilities necessary for conducting this research. The financial support with the grant number 18/CRT/6223, which I received from the SFI Centre for Research Training in Artificial Intelligence (CRT-AI), greatly facilitated the execution of my research project.

I extend heartfelt gratitude to my family for their unwavering support. Special thanks to my girlfriend, for her love, encouragement, and understanding. Their belief in me sustained me through the challenges.

I would like to acknowledge the participants of my study whose willingness to share their experiences and insights made this research possible. Their contributions are invaluable, and I am thankful for their time and cooperation.

Last but not least, but certainly not least, I want to express my deepest gratitude to my friends who stood by me during the ups and downs of this doctoral journey. Their encouragement, laughter, and occasional distractions were a welcome respite from the demands of research.

This dissertation reflects the collective support and encouragement of all these individuals, and I am genuinely thankful for their contributions to my academic journey.

Table of Contents

Table of Contents i

List of Figures iv

List of Tables vi

List of Abbreviations viii

1 Introduction 1
- 1.1 Overview . . . 1
  - 1.1.1 Lifelogging . . . 1
  - 1.1.2 Lifelog Retrieval . . . 4
  - 1.1.3 Image-Text Retrieval . . . 7
- 1.2 Research Motivation . . . 10
- 1.3 Hypothesis and Research Questions . . . 11
- 1.4 Research Contributions . . . 13
- 1.5 Thesis Outline . . . 15

2 Related Work and Background 18
- 2.1 Lifelog Retrieval . . . 18
  - 2.1.1 Early Lifelog Retrieval Systems . . . 18
  - 2.1.2 NTCIR-Lifelog Challenge . . . 24
  - 2.1.3 ImageCLEF-Lifelog Challenge . . . 29
  - 2.1.4 Lifelog Search Challenge . . . 33
- 2.2 Image-Text Retrieval . . . 40
  - 2.2.1 Independent Embedding Models . . . 40
  - 2.2.2 Cross-embedding Models . . . 42
  - 2.2.3 Fusion Models . . . 44
  - 2.2.4 Pre-training on large dataset . . . 46
- 2.3 Graph-based Retrieval Models . . . 48
- 2.4 Chapter Summary . . . 50

3 Research Methodology and Evaluation Methods 52
- 3.1 Research Methodology . . . 52
- 3.2 Operating Constraints . . . 55
- 3.3 Evaluation Methods . . . 56
  - 3.3.1 Evaluation metrics for a lifelog retrieval system . . . 56
  - 3.3.2 Recall@K metrics for an image-text retrieval model . . . 58
  - 3.3.3 Mean Reciprocal Rank used in the pilot study . . . 59

4 Baseline Lifelog Retrieval System **60**
4.1 Introduction . . . 60
4.2 MyScéal at LSC'20 . . . 61
4.2.1 Concept-based Search Engine . . . 62
4.2.2 User Interface . . . 68
4.2.3 LSC'20 Configuration . . . 71
4.2.4 Performance . . . 72
4.2.5 Learning Improvement . . . 76
4.3 MyScéal at ImageCLEF2020-Lifelog . . . 76
4.3.1 System Modification . . . 76
4.3.2 Performance . . . 79
4.3.3 Learning Improvement . . . 83
4.4 MyScéal 2.0 at LSC'21 . . . 84
4.4.1 Novice User Experiment for MyScéal at LSC'20 . . . 85
4.4.2 System Modification . . . 86
4.4.3 LSC'21 Configuration . . . 88
4.4.4 Performance . . . 88
4.5 Discussion and Conclusion . . . 91
4.6 Chapter Summary . . . 91

5 Graph-based Models in Image-Text Retrieval **93**
5.1 Introduction . . . 93
5.2 A Deep Local and Global Scene-Graph Matching Model . . . 94
5.2.1 Scene-graph Structure . . . 94
5.2.2 Methodology . . . 95
5.2.3 Experiment Configuration . . . 105
5.2.4 Results and Discussion . . . 106
5.3 HADA: A Graph-based Amalgamation Framework . . . 108
5.3.1 Methodology . . . 109
5.3.2 Experiment Configuration . . . 116
5.3.3 Results and Discussion . . . 119
5.4 Discussion and Conclusion . . . 121
5.5 Chapter Summary . . . 122

6 Graph-based Lifelog Retrieval System **123**
6.1 Introduction . . . 123
6.2 Pilot Study: Scene Graph Matching in Lifelog Retrieval . . . 124
6.2.1 Data Indexing . . . 125
6.2.2 Lifelog Image Retrieval . . . 128
6.2.3 Experiment . . . 129
6.2.4 Results and Discussion . . . 132
6.3 An Empirical Comparison of Performance of HADA and Other Approaches in Lifelog Retrieval . . . 133
6.3.1 HADA Modification . . . 134
6.3.2 Datasets . . . 135
6.3.3 Non-Interactive Experiment . . . 137
6.3.4 Interactive Experiment . . . 140

6.4 Conclusion 148
6.5 Chapter Summary 149

7 Conclusion **151**
7.1 Proving the Hypothesis 151
7.1.1 Research Question 1 151
7.1.2 Research Question 2 152
7.1.3 Research Question 3 154
7.2 Research Contributions 155
7.3 Limitations 156
7.4 Future Works 157
7.4.1 Lifelog Retrieval System 158
7.4.2 Graph-based Image-Text Retrieval Model 158
7.5 Publication List 159
7.5.1 Chapter 4 - RQ1 159
7.5.2 Chapter 5 - RQ2 160
7.5.3 Chapter 6 - RQ3 161
7.5.4 Others 161

A List of Lifelog Queries Used in Lifelog Experiments **163**
A.1 Novice User Experiment in Chapter 4 163
A.2 Experiments in Chapter 6 163
A.2.1 Pilot Study 163
A.2.2 Automatic and Interactive Manner Experiments 168

Bibliography **174**

List of Figures

1.1 Gordon Bell, who is a lifelogger and a lifelogging pioneer, at a conference in 2013 wearing his sensor devices such as audio recorder, Narrative Clip, and OMG Autographer [1]. 3
1.2 An example of a lifelog image. 5
1.3 Some correct lifelog images of the lifelog query: *"Show all moment I was looking at items in a toyshop where various toys are being examined"*. 6
1.4 An example shows that the interaction between concepts is important information. 8
1.5 An example image and its captions in an image-text retrieval dataset. 8

2.1 The interfaces of Microsoft's MyLifeBits 19
2.2 SenseCam Photo Viewer [2] 20
2.3 The image browser system developed by Lee et al. [3] 22
2.4 The first lifelog retrieval system using multifaceted search mechanism [4] 23
2.5 LEMoRe's web-based interface [5] 27
2.6 DCU team's interactive lifelog retrieval system in NTCIR14-Lifelog challenge [6] 29
2.7 LIFER system at the ImageCLEF2018-Lifelog Challenge [7] 32
2.8 The input interface of HCMUS in ImageCLEF2019-Lifelog challenge [8] 33
2.9 UU-DCU system in the virtual reality environment [9] 35
2.10 User interface of LifeXplore system in LSC'18 [10] 36
2.11 VIRET system in LSC'18 [11] 38
2.12 Exquisitor system in LSC'19 [12] 39
2.13 Two most commonly used pipelines of image-text retrieval models . . 41

3.1 Research methodology schema 53
3.2 The workflow of research questions and approaches 55

4.1 Cycle path of our baseline concept-based system 61
4.2 MyScéal system's pipeline 62
4.3 User Interface of MyScéal at LSC'20 68
4.4 Supplementary elements in MyScéal's interface 70
4.5 A semantic query and relevant lifelog images. The symbol " | " indicates hints in the query. 72
4.6 The precision and recall of some systems participated in LSC'20. The teams are ordered based on their official rank in the competition. . . 75
4.7 Number of times when a system was in the top three quickest systems submitting correct answers. 75
4.8 A modified UI of MyScéal for ImageCLEF2020-Lifelog 78
4.9 A modified pop-up window showing all scenes and events in a day . . 80

4.10 F1@10 scores of teams participating in ImageCLEF2020-Lifelog retrieval task 82
4.11 Updates on UI and the Map panel in MyScéal 2.0 87
4.12 Pipeline of MyScéal 2.0 87

5.1 An example of scene graphs generated from an image and its caption. The green and the red color indicate objects and their interactions, respectively. 96
5.2 Our proposed LGSGM pipeline 97
5.3 The workflow of an image in our LGSGM model. 98
5.4 The workflow of a sentence in our LGSGM model. The N and M indicate the number of words and the number of relations in the sentence. 101
5.5 The pipeline of HADA. The red borders indicate trainable components. The ITM and ITC infer the training tasks which are Image-Text Matching and Image-Text Contrastive Learning accordingly. They will be discussed in Section 5.3.1.4. 110
5.6 The pipeline of many SOTA models following the independent embedding approach. 111

6.1 The overview of our scene-graph indexing approach for lifelog data and lifelog queries for retrieval purposes. 125
6.2 Image to Graph Pipeline. 127
6.3 Sample results for the example queries of the two types of queries . . 130
6.4 A sample in the LLQA dataset with the description of an event . . . 136
6.5 User interface of the lifelog retrieval system E-MyScéal used in the user study experiment 143
6.6 The scene view pop-up window of E-MyScéal 144
6.7 The distribution of scores based on the order of performance solely (a) and with the systems (b) 148
6.8 The average score of users gained on each batch based on the system that they used 148

List of Tables

4.1 $F1@10$ scores of three users. The *dot* indicates the user could not find the answer. Numbers in **bold** are the highest number in that query among all users. 84
4.2 Experiment score of eight novice users compared to MyScéal team score in LSC'20 86
4.3 Performance of several teams in LSC'21 90

5.1 Performance of LGSGM and others on Flickr30K Dataset. R-Sum is the sum of all R@K metrics. The column **Total** is the sum of R-Sum of Image-to-Text and Text-to-Image. 107
5.2 Performance of models on Flickr30K Dataset. The symbol † indicates the results are originally reported in their research, while others are from my re-implementation using their public pre-trained checkpoints. The column △R shows the gap compared to ALBEF. 119
5.3 Performance of models on the test set in Flickr30K and MSCOCO datasets. The column △R shows the difference compared to BLIP in that dataset. 121

6.1 The assignment of query subsets and systems for each user in our experiment in which A, B, C, and D are our four runs (groups of five topics). 132
6.2 Mean Reciprocal Rank scores of two systems on each type of query and entire dataset. 133
6.3 Performance of embedding-based and concept-based search engines on 23 queries of LSC'21. The symbol † indicates that the results are originally reported in their research. The symbol ★ depicts that the model is fine-tuned on the LLQA dataset. The superscripts ^{1,2,3} depict the best values in each evaluation metric for hints 1, 2, and 3, accordingly. 141
6.4 The assignment of query batches and search engines for each user in the interactive experiment in which A, B, C, and D are 6-query batches. 145
6.5 Performance of each retrieval model on different query batches. WS is short of Wrong Submission. The bold values indicate the best results for that evaluation metric in each row. 147

A.1 List of queries used in the experiment of evaluating the performance of MyScéal operated by novice users. Symbol " | " indicates the hints in a query. 164
A.2 Query batches and their queries used in the pilot study 164

A.3 Query batches and their queries used in the experiments in Section 6.3. Symbol " | " indicates the hints in a query. Batch T contains testing queries that are used to guide searchers on how to use systems. LSC76 is the testing query in LSC'21 but is randomly chosen to be a query in batch B for the interactive manner experiment. 168

List of Abbreviations

SOTA	State-of-the-art
LSC	Lifelog Search Challenge
LMRT	Lifelog Moment Retrieval Task
LSAT	Lifelog Semantic Access Task
I2T	Text-to-Image
T2I	Image-to-Text
MRR	Mean Reciprocal Rank
NLP	Natural Language Processing
GCN	Graph Convolution Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
GRU	Gated Recurrent Unit
RCNN	Region Convolutional Neural Network
BRNN	Bidirectional Recurrent Neural Network
LSTM	Long Short Term Memory
BERT	Bidirectional Encoder Representations from Transformers
TF	Term Frequency
IDF	Inverse Document Frequency
UI	User Interface
GPU	Graphic Processing Unit
GPS	Global Positioning System
LGSGM	Local and Global Scene-graph Matching
HADA	Graph-based Amalgamation
ViT	Vision Transformer

Abstract

Manh-Duy Nguyen

A Search Engine Using Graph-based Structures for Lifelog Retrieval

Lifelog, a personal digital record of daily activities, is becoming popular and can operate as a form of a digital diary. Although having a wide range of applications, the basic underlying challenge of lifelogs is how to build a lifelog retrieval system that can retrieve a specific activity efficiently from large multimodal lifelog data. Recently, there have been many workshops organized to encourage researchers to solve that problem leading to the introduction of many lifelog retrieval systems with different approaches. In addition to the user interface, the backend search engine plays a critical role in the lifelog retrieval system. The fundamental approach for the lifelog search engine is the concept-based model which matches the objects in lifelog images with keywords in a semantic query. However, many state-of-the-art embedding multimodal retrieval models, such as CLIP, have been introduced recently where they encode both images and texts into the same vector space to perform the retrieval. This has opened the opportunity to apply this new generation search engine to the lifelog retrieval task. In this thesis, I propose applying embedding-based models to the lifelog retrieval system. Furthermore, I enhance these models by applying a graph-based structure to them. I also build an interactive lifelog retrieval system that follows the concept-based approach to serve as the baseline in my experiment. Through my interactive user studies, I observe that the graph-based enhanced retrieval model surpasses the conventional concept-based by a large margin in various evaluation metrics. In summary, this thesis primarily contributes to the field of creating an accurate lifelog search engine using graph neural network structures that can be employed to build an effective and efficient lifelog retrieval system to address lifelog retrieval tasks.

Chapter 1

Introduction

1.1 Overview

1.1.1 Lifelogging

While the concept of lifelogging has been around for a considerable period, there is still a lack of consensus on its precise definition [1]. There have been many works trying to clarify its definition [1, 13, 14, 15, 16]. Dodge and Kitchin's [14] explanation is frequently cited as a thorough one when discussing the definition of lifelogging. According to their definition, lifelogging can be considered as a *"form of pervasive computing, consisting of a unified digital record of the totality of an individual's experiences, captured multi-modally through digital sensors and stored permanently as a personal multimedia archive"*. This definition's central idea is that lifelogging should aim to capture all of a person's experiences. Nevertheless, it is currently not possible to record every personal experience due to hardware constraints. On the other hand, Gurrin [1] suggested a less strict definition of lifelogging. He refined the lifelogging meaning to adapt to the current limitation of wearable devices where lifelogging now was assumed as an attempt to passively collect, process, and reflect comprehensively on an individual's actions. In summary, ***lifelogging*** can be interpreted as a mode of recording and storing the personal data of an individual, who is called a ***lifelogger***. The lifelogging is conducted by lifeloggers themselves to record a detailed digital trace of their experiences on a daily basis. This multimodal personal data, which is referred to as

lifelog, is collected using many wearable sensors such as accelerometers, smartwatches, or cameras which makes lifelog varies in different types such as visual images, semantic texts, or biometric signals. It then can be stored on a computing device or on a cloud service.

One of the accessible ways to gather personal experience in daily life is to continuously capture ego-centric images through a wearable camera that is usually attached to the front of the lifelogger. Figure 1.1 shows Gordon Bell, who is a legendary lifelogger and also a lifelogging pioneer, wearing his sensors devices that could automatically record audio and images. He was obsessed with the main idea of the "Memex" concept (short for "memory extender") and wanted to store everything in a digital form [17]. The "Memex", which was a concept proposed by Vannevar Bush in 1945, was described as a "*device in which individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility*" [18]. Following that spirit, Bell and Gemmell introduced MyLifeBits [19, 20] which was a software that stored all personal data from the daily life of Bell himself such as documents, images, and videos as a multi-terabyte digital archive on his computers. This research project of Microsoft Research Lab, along with its introduction of a compact wearable camera called SenseCam [2] that could automatically capture a series of still images of a wearer during a day, made lifelogging become an active research topic although the concept of lifelogging had been introduced years before.

Regardless of how lifelogging is conceptualized, it is clear that it is an area of rising importance for both research and industry due to the rapid advances in wearable devices. There is a wide range of applications for lifelogging. Lifelogging can be applied to monitor dietary habits, allowing users to capture images of their meals. This enables them to track daily food consumption and effectively manage their dietary balance [21]. Additionally, the application of lifelogging extends to market analytics where a visual lifelogging device could help a market research company to evaluate how their advertising campaigns expose to targeted customers



Figure 1.1: Gordon Bell, who is a lifelogger and a lifelogging pioneer, at a conference in 2013 wearing his sensor devices such as audio recorder, Narrative Clip, and OMG Autographer [1].

using captured images and location metadata [22]. There was a recent experiment that used lifelog camera combined with other sensors measuring electrooculogram and heart rate to identify a person's micro-activities, for instance, writing an email, watching a video, or doing physical exercise [23]. Additionally, Kid'sCam which was a lifelogging-based methodology was applied in schools in Wellington, New Zealand to investigate students' behaviors in their daily life [24]. The most popular early application of lifelogging is in memory assistance, which can help lifeloggers or users to recall an event or a milestone they find necessary. An exploratory study showed that capturing personal images of daily activities through the use of a wearable camera may help individuals who suffer memory impairment improve their memory as part of a treatment plan [25]. Regardless of the wide range of use cases, a fundamental underlying challenge of lifelog remains as to how to support effective searching through multimodal lifelog data from what can be a massive archive. Particularly, a lifelogger captures approximately one million images in a

day which is roughly 479GB of disk space for storage requirement [1].

It is important to mention that lifelogging and quantified self[1] are relatively similar and there is a certain degree of vagueness regarding the intersection of these two terms. Quantified self can be defined as a self-tracking practice where a person uses technology to digitalize their own experience to keep track, measure, and analyze the acquired data for the targeted purposes. The key factor to differentiate quantified self and lifelogging is that the former term focuses on a specific domain to answer a pre-defined goal, whilst the latter is more indiscriminate where there is no targeted or unknown end use-case during the recording. Therefore, I consider that lifelogging is different from the quantified self and this assumption is held throughout the dissertation.

1.1.2 Lifelog Retrieval

Retrieving is one of the main benefits of lifelogging as discussed by Sellen and Whittaker [26]. Although there are many types of information that one can capture in lifelogging such as music history, biometric signal, or location history[19, 27], passive photos seem to be the most popular form [1] in which a lifelogger only needs to turn on the device once and leave it operational them during the day. Figure 1.2 shows an example of a still image captured by a lifelog camera worn by a lifelogger. As a consequence, this long period of capture creates a massive visual data archive posing the challenge of retrieving effectively from these vast archives. In fact, lifelog images are the most explored and analyzed data type compared to aforementioned forms in many lifelog fora and workshops [11, 28, 29, 30, 31, 32, 33, 34]. Therefore, in this thesis, we consider ***lifelog retrieval*** as an image retrieval task in which one or many specific lifelog images need to be retrieved within the huge multimodal lifelog database.

Lifelog retrieval is considered a fundamental underlying technology since it is easy for everyone to possess portable sensors and capture lifelog data in the current digital

[1] `https://quantifiedself.com/`



Figure 1.2: An example of a lifelog image.

era. The challenge will be to access relevant data at the appropriate time. Many workshops and competitions about lifelogging have been organized to encourage more research focusing on solving this lifelog retrieval challenge [11, 28, 35, 36]. Specifically, the NTCIR Lifelog and ImageCLEF Lifelog series [28, 29, 34] focus on summarising and analyzing a day of a lifelogger based on their lifelog images. On the other hand, the Lifelog Moment Retrieval Task (LMRT) in ImageCLEF Lifelog series [28, 29, 30, 31] and the Lifelog Search Challenge series [11] creates an evaluation benchmarking for lifelog retrieval systems where participating systems are required to retrieve specific lifelog images from the multimodal dataset given a textual query. Although having different platforms, all of them share the same purpose: to encourage the progress of creating a lifelog retrieval system that can efficiently assist a lifelogger in recalling a specific moment within an extensive archive. These lifelog fora have made lifelog retrieval to become a research field that gathers the attention of international researchers, especially when they also release public anonymized lifelog datasets with multimodal metadata given to the participants. The dataset includes thousands of visual lifelog photos taken by the lifeloggers. Each of them was synchronized with timestamp and location information indicating

when and where it was captured. Besides these primary metadata, a lifelog dataset can consist of other content music that the lifelogger was listening to or biometric data such as heart rate or galvanic skin response. Along with the dataset, these competitions also release a set of semantic queries that help participants to seek correct lifelog images. Each query that describes an event that the lifelogger wants to remember contains information relating to that moment. It also can include the time of the day when it happened or the location where it took place. Thus, based on a given query, participating retrieval systems need to build an efficient search engine that can seek the relevant moment or photos within a time limit. Figure 1.3 shows an example of a semantic query used in the LMRT task in ImageCLEF Lifelog 2020 and some of its relevant lifelog images.



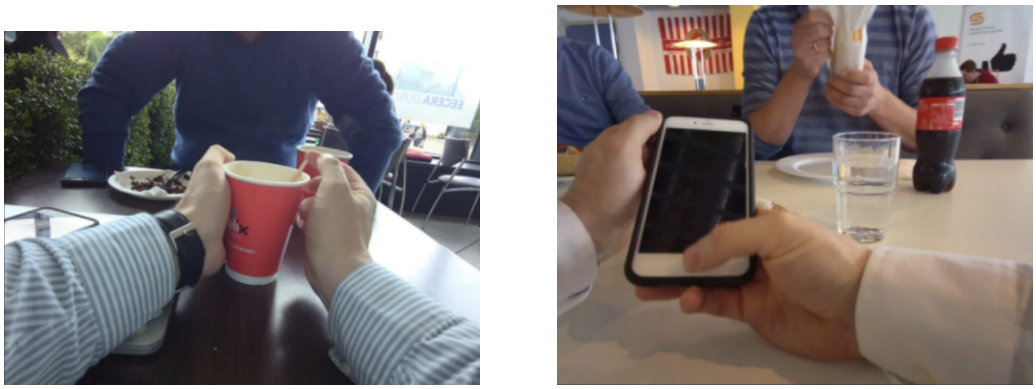
Figure 1.3: Some correct lifelog images of the lifelog query: *"Show all moment I was looking at items in a toyshop where various toys are being examined".*

Many lifelog retrieval solutions have been proposed through international lifelog challenges and competitions. The most common and also fundamental approach to address the lifelog retrieval problem is to create a concept-based search engine that can match keywords in semantic queries to concepts detected in lifelog images to perform the retrieval. Each lifelog image is annotated with textual concepts that are inferred from such as objects appearing in the image, the context of the image, or any words visible in the image. The process of lifelog retrieval is converted into text retrieval since each image can be seen as a semantic document. We can apply some similarity functions usually used in text retrieval, such as BM25 [37] or term frequency-inverse document frequency (TF-IDF) [38], to find the correct images that are relevant to the semantic queries. This simple yet efficient approach

has been applied in many lifelog retrieval systems participated in several lifelog search competitions and obtained good results [8, 9, 39, 40, 41, 42]. Nevertheless, this approach has some disadvantages. Firstly, it relies heavily on the concepts detected from images that do not reflect the real-world wide range of possible objects, hence likely to miss some important information from the images. It is because many object detection models are trained with available datasets whose labels are a limited pre-defined list of words [43, 44]. Secondly, this traditional approach cannot take into account the interaction between visual concepts in an image. Figure 1.4 shows an example that the relations contain valuable information. It is difficult to distinguish Figure 1.4a and Figure 1.4b if only based on concepts *"man"* and *"cup"* because both images all have these concepts. Last but not least, this concept-based approach requires searchers to have knowledge about the concepts that are attached to images in the database in order to have a good result. These problems can be addressed if we use an embedding-based search engine that can not only capture the interaction information but also other details such as visible text or dominant colors in images. The embedding-based approach can both encode images and texts into the same vector space where we can measure their similarity to perform the retrieval. Therefore, users can freely search with any text that is no longer limited to the pre-defined list of concepts annotated with images. Although many multimodal embedding models have been introduced in the image-text retrieval field, it is not until 2023 that this type of search engine has been commonly used in lifelog retrieval yet.

1.1.3 Image-Text Retrieval

Cross-modal image-text retrieval can be considered a broader field of lifelog retrieval. This task consists of a multimodal dataset including images and texts. Each image comes with textual captions describing the image. The ***image-text retrieval*** task can be inferred as finding a unimodal item that is relevant to the other modality item in the vision-and-language dataset. The image-text retrieval, therefore, is divided



(a) A man is holding a cup

(b) A man and a cup

Figure 1.4: An example shows that the interaction between concepts is important information.

into **two** sub-tasks which are ***image-to-text retrieval*** and ***text-to-image*** retrieval. In particular, the former sub-task is to find a caption that is relevant to a given image, while the latter sub-task is vice versa. Figure 1.5 shows an example of an image and its description in a dataset used in image-text retrieval.

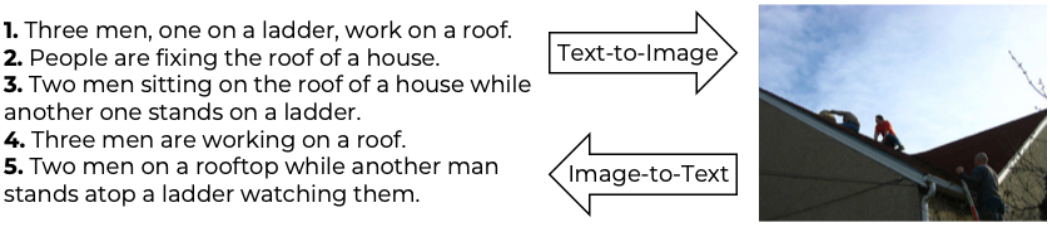


Figure 1.5: An example image and its captions in an image-text retrieval dataset.

Although computer vision and natural language processing are two of the most popular research fields, it was not until 2010 that the first dataset [45] which is appropriate for image-text retrieval was introduced. This preliminary dataset only consists of 1000 images and 5000 descriptions annotated by using Amazon's Mechanical Turk, which is a crowdsourcing-based framework for anonymous non-expert or novice users to perform annotation tasks. The dataset has grasped the interest of researchers with many solutions for this retrieval task. The first approach from Farhadi et al. [46] followed the simple concept-based method when they projected each image and each sentence into the same semantic triplet space

of object, action, and scene to calculate the similarity score between them. Regarding the deep learning models, the first multimodal embedding retrieval model for this task was DT-RNN proposed by Socher et al. [47] who encoded sentences and images by using Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN), respectively. Since technology develops with more powerful computation hardware such as graphics processing units or GPUs being introduced, there is a need for a bigger dataset to make the most of the processors and to obtain better results. The two most current popular datasets for image-text retrieval are MSCOCO [43] and Flickr30K [48] which includes 123,000 images and 31,000 images, accordingly. Along with that, the introduction of Transformer structure [49], especially BERT [50], has led to an increase in the number of state-of-the-art (SOTA) retrieval models in this joint vision-and-language research field. Particularly, the CLIP[2] model [51] is a game-changer not only in the image-text retrieval task but also in other vision-and-language tasks. This multimodal embedding model utilized SOTA encoders for each modality. Specifically, Vision Transformers [52] and Transformers were employed to encode images and texts, respectively. Nevertheless, the main reason that CLIP and other SOTA models obtained impressive results is not only because of using encoder network architecture with hundreds of millions of trainable parameters but also comes from pretraining on massive external datasets. This is a challenge to build a retrieval model that can surpass current SOTA models, especially for those who lack computing resources since it requires a decent amount of time to train a model on external datasets. For example, DT-RNN took at least 8 days to train, whilst CLIP needed 12 days with 256 GPUs to get the best performance.

Although lifelog retrieval contains many modalities rather than only visual and language, it is fundamentally the text-to-image retrieval sub-task because the main problem of lifelog retrieval is to find lifelog images that are relevant to the given

[2]CLIP: Contrastive Language-Image Pre-training

textual query, as shown in Figure 1.3. Therefore, it is possible to apply SOTA image-text retrieval models to lifelog retrieval, especially when there is still a limited number of works about applying them.

1.2 Research Motivation

The most popular approach in the lifelog retrieval challenge is the traditional concept-based method that annotates lifelog images with visual concepts detected within them. However, this fundamental approach has several limitations. Firstly, it cannot capture interactions or relationships between concepts, which is considered important information in the retrieval task as shown in Figure 1.4. Being able to learn this interaction information can improve the model performance. The second disadvantage of the traditional retrieval models is that they require searchers to have knowledge about the annotated concepts to get the best performance. It is because if users search for a concept that does not belong to any image's annotated concept, they cannot find the relevant lifelog images. It is a considerably difficult constraint since not all users are experts in this field. Last but not least, a retrieval model is not enough to tackle the lifelog retrieval task. The ultimate purpose of lifelog retrieval is to support lifelogger, or people in general, to recall a memorable milestone. It is necessary to create a lifelog retrieval system to facilitate obtaining that target.

The limitations of the traditional concept-based method can be addressed with the following solutions. Regarding the relationship between concepts, a graph-based structure is appropriate to describe the interactions between concepts since nodes of graphs represent the concepts and edges represent their relations. This structure has been applied to image-text retrieval recently and has shown promising results [53, 54]. On the other hand, a lifelog retrieval system needs to be ease-of-use and not be confusing to users. Furthermore, the system should operate in an interactive manner and return retrieved results as a ranked list to be appropriate for the huge

lifelog archives [1] For those reasons, it motivates me to develop an interactive lifelog retrieval system that uses a graph-based search engine and is easy to use.

1.3 Hypothesis and Research Questions

The research in this dissertation focuses on the building of an enhanced search engine using graph-based structures for an interactive lifelog retrieval system. The traditional concept-based retrieval models can work well and achieve competitive results in many lifelog search competitions. However, it confronts several limitations. This approach only captures some specific concepts and does not capture other information such as any activity happening in an image or the interactions between concepts although they can be really useful for retrieval. Meanwhile, graph-based structures can address the disadvantages of the concept-based method but this type of data structure has not been widely applied in the lifelog retrieval task. Therefore, I believe that an interactive lifelog retrieval system that employs a graph-based search engine can surpass one following the conventional concept-based method. To validate my conjecture, I define the following hypothesis for this Ph.D. research:

Hypothesis

Applying a graph-based structure to an interactive lifelog retrieval system can help to increase performance in terms of speed and accuracy compared to conventional concept-based interactive retrieval systems.

In order to either prove or disprove this hypothesis, a number of related research questions have been developed as follows:

- **Research Question 1 (RQ1). Conventional Concept-based Lifelog Retrieval System**

  What is the state-of-the-art of the interactive lifelog retrieval systems following the conventional concept-based approach?

  First of all, I need to develop a baseline interactive lifelog retrieval system in

order to prove or disapprove the hypothesis. This system should employ the concept-based retrieval model as the internal search engine. Additionally, the system also has to come with an ease-of-use user interface that is carefully designed for not only expert users but also novice users who are not familiar with this lifelog retrieval task. I will create the system based on the insight that I have learnt from other systems' features including their search engines and interfaces. The proposed system is evaluated by participating in many well-known lifelog search competitions to show that it is the SOTA concept-based lifelog retrieval system and is appropriate to be served as the baseline in my hypothesis.

- **Research Question 2 (RQ2). Graph-based Cross-modal Retrieval Models in Image-text Retrieval**

  Can using graph-based structures improve the retrieval accuracy of current state-of-the-art cross-modal retrieval models in the image-text retrieval task?

  This research question is proposed because graph-based structures have not been applied widely in the lifelog retrieval task as I have mentioned in the previous parts. Meanwhile, the image-text retrieval task, which is a broader research field compared to the lifelog retrieval task, has witnessed many retrieval models using this type of data structure and achieving good results. Therefore, it is better to approach this general retrieval field to develop an enhanced retrieval model. I try to improve the retrieval accuracy of the current SOTA model in this task by integrating graph-based structures into it. The term "*accuracy*" referred to *Recall at K* metric which is commonly used in the image-text retrieval task. The detail of this metric is described in Section 3.3.2. I evaluate the proposed model on several popular image-text retrieval datasets to find the answer to this research question.

- **Research Question 3 (RQ3). Graph-based Lifelog Retrieval System**

  To what extent can a graph-based interactive lifelog retrieval system improve

retrieval performance compared to the conventional concept-based system?

After developing a new SOTA graph-based retrieval model in the image-text retrieval field, I then apply it to the lifelog retrieval task. I first conduct a proof-of-concept experiment where I want to have a preliminary comparison between concept-based and graph-based lifelog retrieval systems. A small lifelog query bank, including my manually made queries and a subset of ordinary queries that are used in an official lifelog search challenge, is also created for the experiment. In this pilot study, I only focus on how accurate the search engines are based on the ranked list of retrieved lifelog images returned by them. The search time and the system's interface bias are not taken into account in this fundamental study. Nevertheless, to have a fair and unbias comparison, my final comprehensive experiment needs to consider these criteria. Both systems will have the same user interface with the same features and how to interact with it. I follow the format of the Lifelog Search Challenge and use their scoring function to assess both search time and the accuracy of the systems. A bigger lifelog query bank is also used for this comparison. The result of this user study will answer my last research question, leading to the conclusion of the hypothesis of my thesis.

1.4 Research Contributions

My thesis focuses on building a new generation of a lifelog search engine following the embedding-based retrieval model using a graph-based structure compared to the conventional concept-based approach. In this section, the key contributions made in this thesis are listed as follows:

- I review the core features of lifelog retrieval systems that participated in several lifelog competitions. I then develop a SOTA interactive lifelog retrieval system that followed the most popular concept-based approach with the core features that I find useful and suitable for lifelog retrieval. I evaluate the performance

of the proposed concept-based lifelog retrieval system by participating in many lifelog retrieval competitions and achieved top-performance results. The results show that the system using a conventional search engine can be considered as a SOTA lifelog retrieval system and appropriate to be used as a baseline in my experiments.

- I conduct a proof-of-concept experiment where I develop a graph-based lifelog retrieval model and compared it with the concept-based baseline model. The proposed model applies graph-based structures adopted from the image-text retrieval field to capture the interaction between concepts in images and texts while encoding them into matrices. I also create a small lifelog query bank for comparison. The list of the queries is described in Appendix A.2. The result shows the promise of applying SOTA graph-based techniques in image-text retrieval to lifelog retrieval.

- I review SOTA search models in the general image-text retrieval task, which is currently having many approaches more than the specific lifelog retrieval task, to gain more insight into retrieval models. Based on these models, I extend and improve their performance on the image-text retrieval datasets by integrating graph-based network architectures into the original models to overcome their limitations. I propose two graph-based SOTA cross-modal retrieval models. The first model addressed the technical disadvantages of the baseline to enhance the results. However, with the pre-training stage on massive datasets, the recent CLIP model surpassed all other retrieval models. My second model, therefore, proposed using graph-based structures to combine many SOTA models including CLIP to obtain better performance without using multiple GPUs to pre-train on huge datasets like CLIP.

- I apply my SOTA model proposed in the image-text retrieval task to the lifelog retrieval task by integrating the model into the aforementioned lifelog retrieval system. Additionally, I try fine-tuning the model on a lifelog

question and answer dataset. I evaluate the proposed model with other lifelog retrieval models including concept-based baseline search engine and CLIP in the automatic manner comparison. The results show that my model, which is built from the image-text retrieval task, obtains better performance than other models in the non-human-in-the-loop experiment.

- I conduct a user-study experiment following a human-in-the-loop configuration in which I ask eight volunteers to use two interactive lifelog retrieval systems to solve a set of official lifelog queries in a lifelog search challenge. One system employs the traditional concept-based retrieval model, while the remaining system utilizes my proposed graph-based model. Both systems have the same user interface, which is adapted from the SOTA interactive retrieval system, to avoid bias in how users interact with systems. The result shows that using the graph-based search engine could not only help users to find the correct lifelog images faster but also with higher accuracy.

1.5 Thesis Outline

This thesis is mainly about using a graph-based deep learning network to build a SOTA lifelog retrieval model. In this chapter, I have briefly provided an introduction to lifelogging and its fundamental application called lifelog retrieval as well as the image-text retrieval task. My motivations and the hypothesis were also discussed. Additionally, I proposed my research questions to prove the hypothesis and gave a summary of the research contributions. The remainder of this thesis is organized as follows:

- Chapter 2 presents the overview of the background of lifelog retrieval systems including their backend search engines and their frontend user interfaces. Many retrieval models in the image-text retrieval field are reviewed and discussed in their network architecture. Furthermore, some specific retrieval models using graph-based structures are summarized as well.

- Chapter 3 presents the research methodology and evaluation metrics to address three research questions in my research.

- Chapter 4 presents the work in creating the state-of-the-art interactive lifelog retrieval system called MyScéal that employed the conventional concept-based search engine. The performance of the system, which was evaluated through many lifelog search competitions that MyScéal participated in, is also analyzed. Along with that, I also briefly review the lifelog benchmarking challenges and their criteria.

- Chapter 5 presents my research in developing two proposed cross-modal retrieval models to address the image-text retrieval task. While the first model uses graph-based structures to capture the interaction between concepts in images and texts, the second model applies this kind of structure to combine any SOTA retrieval models that are publicly available to produce the better one. Their accuracies are also analyzed and discussed through their performance on the two most widely used datasets in the image-text retrieval challenge.

- Chapter 6 presents my attempt in applying the SOTA retrieval models built in Chapter 5 in the lifelog retrieval field and also the comparison between the proposed graph-based models and the conventional SOTA lifelog retrieval model created in Chapter 4. A preliminary proof-of-concept study is described to show the opportunity of using these graph-based retrieval models in the lifelog retrieval task. Another comprehensive user-study experiment, with two different manners including automatic and interactive settings, is presented in this chapter. Additionally, a short introduction about an annotated lifelog dataset, which was used to fine-tune my proposed graph-based models, is also discussed. A detailed experiment design and the results are discussed to prove my hypothesis.

- Chapter 7 presents a summary of three research questions and concludes the

hypothesis of this dissertation. Limitations of the research, future work, and the list of my publications are also discussed in this chapter.

Chapter 2

Related Work and Background

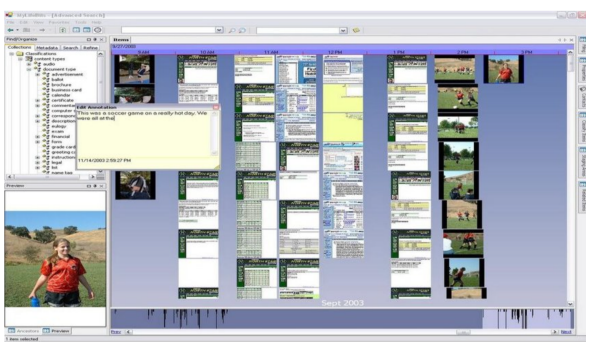
In this chapter, existing research that is related to my work is reviewed to help me resolve my research questions. Section 2.1 gives an overview of various lifelog retrieval systems including their search engines and user interfaces. Section 2.1 also briefly describes some popular lifelog retrieval competitions to highlight features of participating systems that were supposedly created for the challenges. The image-text retrieval task is discussed in Section 2.2 with a comprehensive outline of some state-of-the-art methods. Finally, Section 2.3 introduces some graph-based models that have been proposed in both retrieval research topics.

2.1 Lifelog Retrieval

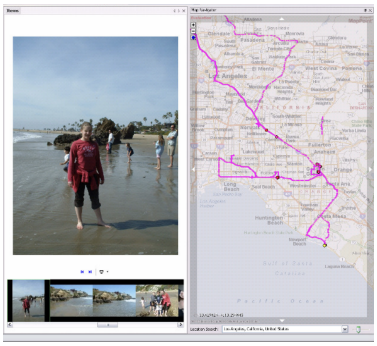
2.1.1 Early Lifelog Retrieval Systems

As was mentioned in Chapter 1, although the concept of lifelogging was proposed by Vannevar Bush [18] in 1945, it was not until 2001 that the first attempt to create software that can preserve the personal life experience was started by Gordon Bell at Microsoft [19, 20]. The software, called MyLifeBits, allowed the archiving of multimedia daily life data of Bell such as images, movies, emails, books, phone calls, or papers in the SQL database. MyLifeBits can be considered as the first lifelog retrieval system because it supported organizing, annotating, and searching its content. Its retrieval mechanism was based on the SQL database inquiries with the basic built-in full-text search and allowed users to find images with pre-defined annotations. Therefore, each non-text lifelog item, which could be an image or a

video, was able to be annotated with a semantic piece of information in order to be searched. For instance, an image could be attached with the text indicating the date it was captured or just a quick note describing what a lifelogger thought about it. Figure 2.1 shows the user interface of MyLifeBits that had the option to open a map interface. This map tracked the location of the lifelogger and might be attached to any lifelog item, for example, an image as illustrated in Figure 2.1b. The main interface of MyLifeBits, as depicted in Figure 2.1a, contained a large central area for showing photos that were organized in a timeline of a day from left to right. It was possible to directly refine an image's metadata such as its annotation with this interface and perform searching through the search bars at the top. However, although MyLifeBits supported users in finding specific data based on its textual annotation, the system worked as a database inquiry tool rather than a retrieval tool.



(a) MyLifeBits's main user interface [55]



(b) MyLifeBits's map panel [56]

Figure 2.1: The interfaces of Microsoft's MyLifeBits

While MyLifeBits was initially designed for general data gathered from many resources, Microsoft also developed an application called SenseCam Photo Viewer which was designed specifically for a wearable SenseCam camera [2]. This small lifelog camera could not only capture still images without intervention while being worn but also collect several sensors including temperature, accelerometer, or GPS signal. The SenseCam Photo Viewer application will automatically synchronize all images and sensor signals from SenseCam if we connect the camera to the computer.

The application then serves as a tool to manage and display images along with sensor signals as a playback video. Figure 2.2 shows the interface of the SenseCam Photo Viewer application which mainly visualized the sequence of images in the central area in the form of a video player. This basic viewer tool allowed users to remove unwanted images and create a bookmark which was an image that users found memorable as depicted at the left panel of the interface. They also could annotate the bookmarks with semantic annotations. Although this tool had limited functionality, it and MyLifeBits still laid a fundamental brick in developing a lifelog retrieval system.

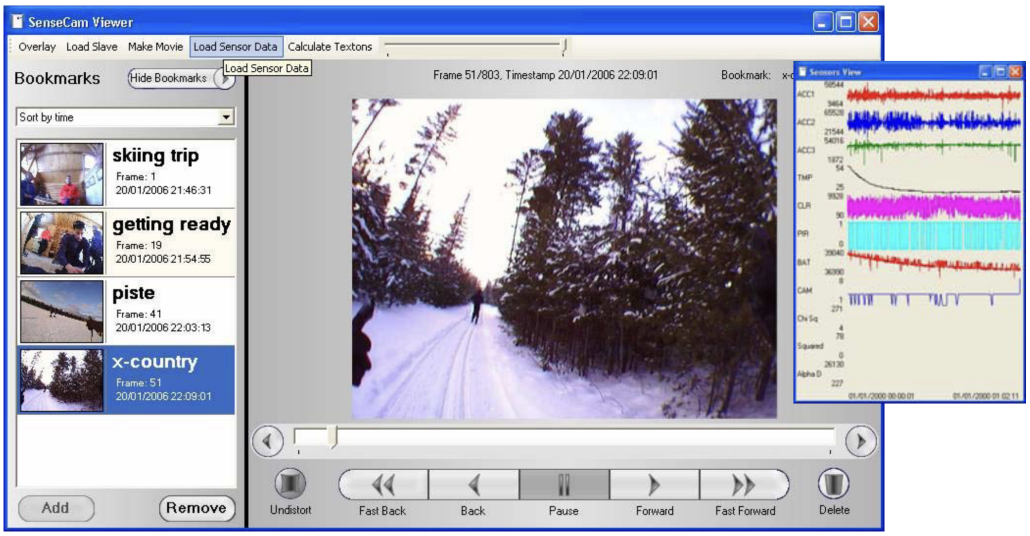


Figure 2.2: SenseCam Photo Viewer [2]

Inspired by the SenseCam Photo Viewer application, Lee et al. [3] made an effort to create a digital visual diary from images captured by the SenseCam camera. Since the camera consistently captured images every 50 seconds, which is configurable, it resulted in a substantial number of photos by the end of the day. Therefore, it was hard for users to go through all the photos quickly when they wanted to remember a moment. It would be even more difficult if the number of images increased. Lee addressed this issue by grouping photos into events, such as having breakfast or chatting with friends, using the previous lifelog segmentation work [57]. Images were allocated into events based on not only their visual content

but also their corresponding sensor signals. Regarding the content-based analysis, the image's color information and edge histogram was used to distinguish the difference between its adjacent groups of images from which an event segmentation algorithm [58] decided to allocate the image into the relevant group. Additionally, images were also segmented into events based on sensor data. A significant change in the signal might indicate a new event. For example, a gap in the temporal temperature signal could imply that the lifelogger moved from the indoor to the outdoor environment. After the segmentation, a single photo in each event was selected as a thumbnail to be visualized in the main user interface of the application as shown in Figure 2.3. The thumbnail image was chosen either from their color features or just a simple method that selected the middle image in the event. Moreover, as we can see in Figure 2.3, not all thumbnail images had the same size in the central panel, in which a bigger image indicated a more important event. An event was more important than others if it was more unique and more different compared to the ordinary events, for instance, the event of having a chat with friends was considered more important than the event of working in the office. The differences between events were determined by analyzing the representative visual features derived from the average characteristics of the images within each event. This method also helped the system allow users to find similar events from the selected one. Similar events were displayed in the right area of the system, while the left panel was utilized for the calendar widget as shown in Figure 2.3. There was a search bar below the widget that the lifelogger could use to search images containing relevant annotations. This digital visual diary system also supported the lifelogger to highlight their favorite moments allowing users to find them later easily. Overall, this system contained most of the important functions for an interactive lifelog retrieval system, such as finding similar images, grouping events, and showing events in a day, compared to MyLifeBits and SenseCam Photo Viewer.

Not a long time later, Doherty et al. still adapted the event segmentation idea

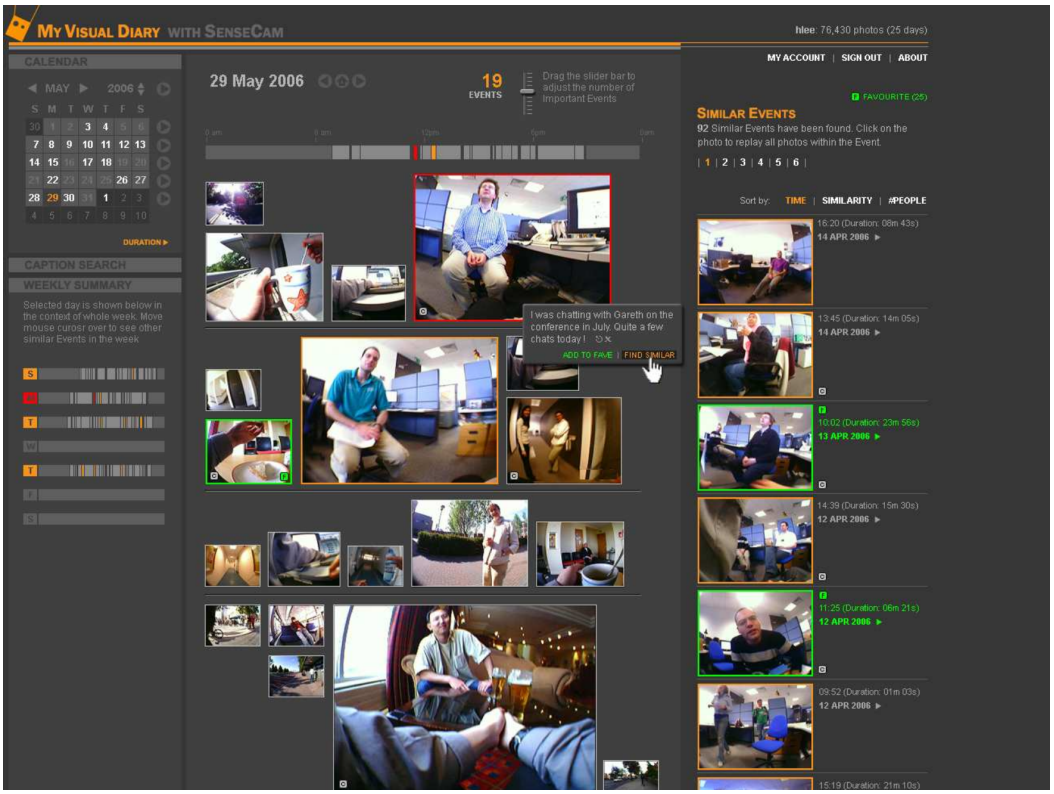


Figure 2.3: The image browser system developed by Lee et al. [3]

to build the first multifaceted lifelog retrieval system but was more conducive for searching than Lee's system [4]. The development of the system was informed by prior research findings indicating that lifeloggers could enhance their image retrieval efficiency by recalling the four "W" axes associated with the images: "*What*" happened in the image, "*Who*" was there, "*Where*" and "*When*" it occurred. Therefore, they created the lifelog retrieval system supporting faceted search with four fields which satisfied them. Figure 2.4 illustrated the interface of Doherty's system in which these faceted panels were located on the left of the window and highlighted in green color. Users could browse for the wanted images based on their memory of four "W" axes with corresponding checkboxes. Particularly, the "*Who*" axis indicated how many people were detected in the images, and the "*Where*" axis was for the location information such as longitude and latitude. Meanwhile, the "*When*" axis allowed users to filter the temporal metadata including year, month, day, season, or time in a day. The "*What*" axis

was about annotated semantic visual concepts of the image. An experiment comparing using this multifaceted system and Lee's system was conducted with one lifelogger as a user. The result showed that the user significantly reduced the median search time for an event from 774 seconds with the old system to 127 seconds using the faceted search.

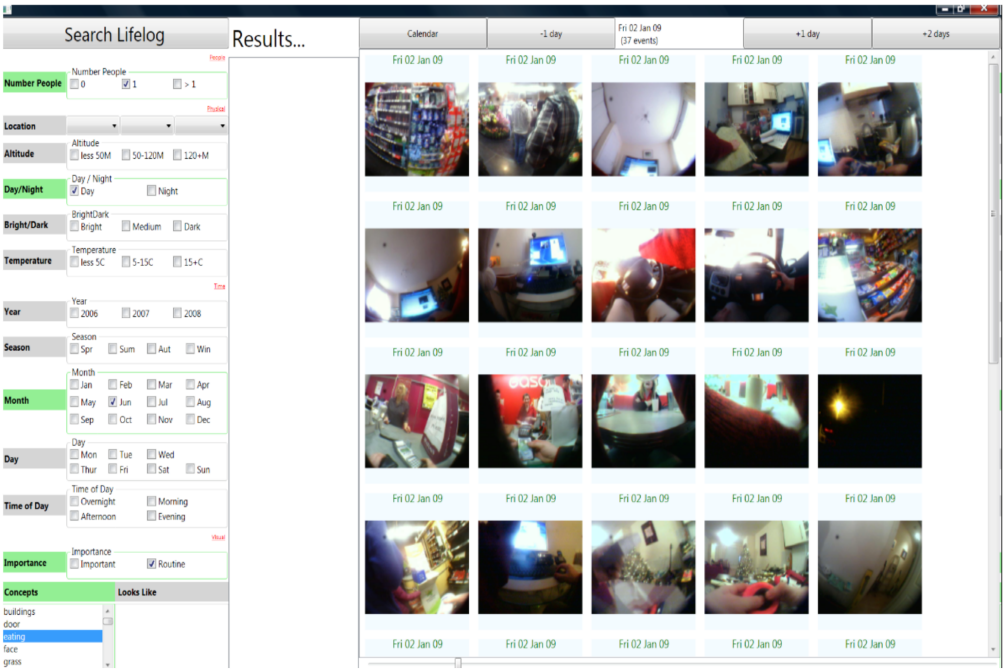


Figure 2.4: The first lifelog retrieval system using multifaceted search mechanism [4]

There were other lifelog retrieval systems with different interfaces and platforms [59, 60, 61] were proposed after Microsoft introduced MyLifeBits which made lifelog retrieval became an active research field. However, there was no clear comparison between them or any benchmarking to evaluate the performance of the systems. Although Doherty made the first experiment to evaluate the performance of two systems [4], it still had several limitations. Its design could be further improved since there was only one user. Additionally, the number of queries and the set of queries used for each system were also different. Therefore, it rose a need to have a comprehensive evaluation mechanism with a dedicated design and appropriate metrics to encourage researchers to create new systems with novel approaches. There are currently three popular lifelog benchmarking efforts which are organized to fulfill

it including NTCIR-Lifelog[1], ImageCLEF-Lifelog[2], and Lifelog Search Challenge[3].

2.1.2 NTCIR-Lifelog Challenge

NTCIR-Lifelog series [32, 33, 34, 62] was the first rigorous comparative evaluation platform designed for the lifelog field. It aimed to encourage researchers to have further analysis of lifelog data, especially to develop state-of-the-art lifelog retrieval systems. Although this series organized many sub-tasks with different targets on lifelog, only the retrieval task which was called Lifelog Semantic Access Task (LSAT) was considered in the scope of this dissertation. The LSAT task required participants to retrieve a specific moment in the dataset from a given textual description, which was similar to a known-item search task. A moment is considered an activity or an event that occurred in the dataset, which might happen more than once (e.g. "*Find the moment(s) when I was looking at Tower Bridge in London*"). Participating teams needed to find all events that were relevant to the query regardless of automatic or interactive configuration. The interactive manner allowed teams to freely refine their input during the search process. Meanwhile, the automatic (non-interactive) run allowed users to only change the input for the query before doing the retrieval, hence they played no role during the search process. The evaluation metrics used in this challenge were precision and recall which will be discussed in detail in Chapter 3.

To organize this retrieval task, NTCIR12-Lifelog which was the first run of the NTCIR-Lifelog challenge series published the first available collection of multimodal lifelog data ever. The dataset consisted of 88,124 images gathered from three lifeloggers [35]. It came with metadata which consisted of the time, semantic location of the lifeloggers (e.g. home, work, restaurant, etc.), and physical activities (e.g. driving, walking, transport, etc.) attached to each image. Additionally, an image was also annotated with a list of visual concepts detected by running a

[1] `http://lifelogsearch.org/ntcir-lifelog/`
[2] `https://www.imageclef.org/2020/lifelog`
[3] `http://lifelogsearch.org/lsc`

CNN-based object detector [63]. All images with potentially offensive information and identifiable faces were removed to guarantee privacy protection. Five teams participated in the NTCIR12-Lifelog task. There were four non-interactive systems including LIG-MRM [64], III&CYUT [65], VTIR [66], and QUT [67], and one interactive system called LEMoRe [5]. The interactive system with humans involved in solving queries acquired better results compared to non-interactive systems. Regarding the non-interactive methods, the LIG-MRM system was the best team among the four of them. For each image, LIG-MRM concatenated its embedding features from three different convolutional neural network (CNN) models that were mostly used at that time including AlexNet [68], GoogleNet [69], and VGG [70]. The feature was then normalized and fed into 346 separated factorized multi-SVM [71] models to predict if the lifelog image contained each of the 346 visual concepts defined in the TRECVid 2013 data [72] with the corresponding probability score. To perform the retrieval, the team manually selected concepts in a query. These concepts were in the list of 346 pre-defined concepts predicted for each image, and their cumulative score was determined by summing their respective probability values. The retrieved images were ranked based on their final scores and then were filtered based on other information about location and time to make the final result. On the other hand, the III&CYUT team approached this challenge by using a natural language processing (NLP) embedding Word2Vec model [73]. They measured the similarity between a query and an image by computing the cosine distance between the encoded concepts in the query and the encoded concepts in the image that came from the metadata. This team also extended the concepts in the query with similar words based on the Word2Vec model to improve the performance. Regarding the only interactive system, LEMoRe was the first interactive lifelog retrieval system that participated in the official lifelog search competitions. This system had the same idea as other teams since it assumed that an event can be retrieved based on the annotated concepts of lifelog images. LEMoRe enlarged the number of visual concepts by using the LSDA

object detection model [74] to increase the number of concepts of an image to roughly 4,300. Each lifelog image was attached with its detected concepts and their detection probability scores. LEMoRe employed the same similarity function with the III&CYUT team to perform the retrieval. Nevertheless, its interface as depicted in Figure 2.5 displayed to users the similarity heatmap in the main view. This heatmap indicated which day and what time of the day contained images that were relevant to the concepts from the query. For instance, the heatmap depicted in Figure 2.5 provided valuable insights. It revealed that images with the "laptop" concept were typically taken between 8 A.M. and 8 P.M. Additionally, there were specific days when the appearance of the "laptop" concept was notably infrequent in the lifelog images. Based on the heatmap, users could further investigate to find the answer to the query. In addition, the LEMoRe system introduced the concept similarity functionality which help them to mitigate the issue of a novice user who was not familiar with using the system and did not know the list of concepts. For each concept that the user inputted into the search bar, LEMoRe suggested similar concepts that were available in the annotation by using WordNet [75], which could help the user to find the answer easily. In order to find all moments as required by the NTCIR-Lifelog task, LEMoRe allowed users to search similar images based on their color features or histogram of oriented gradients using the LIRE engine [76]. LEMoRe simulated a real-world use case by asking a novice user to use the system to solve queries and compared the result with that of an expert user.

NTCIR13-Lifelog challenge [33] was the second iteration in the series. It followed its previous run's guidelines to gather and publish the lifelog dataset to participating teams. The dataset used in this challenge contained the lifelog data from two lifeloggers who collected through 90 days and comprised 114,547 images with metadata. The metadata still included the timestamp, location, semantic physical activities, and visual concepts produced by using Microsoft's computer vision API[4]. Nevertheless, the multimodal dataset from the NTCIR13-Lifelog

[4]https://azure.microsoft.com/en-us/products/cognitive-services/vision-services

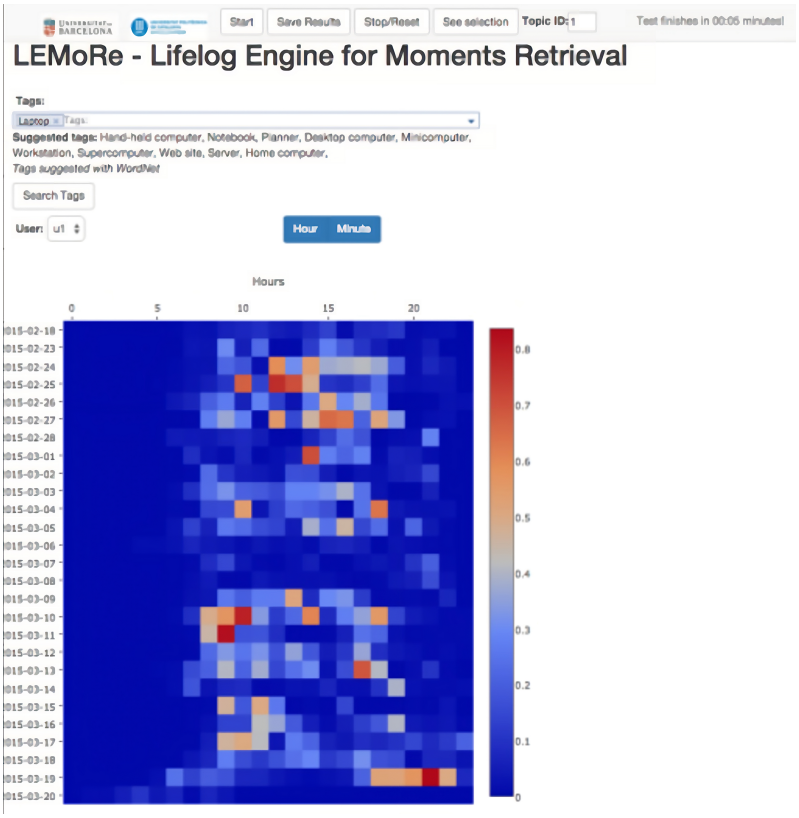


Figure 2.5: LEMoRe's web-based interface [5]

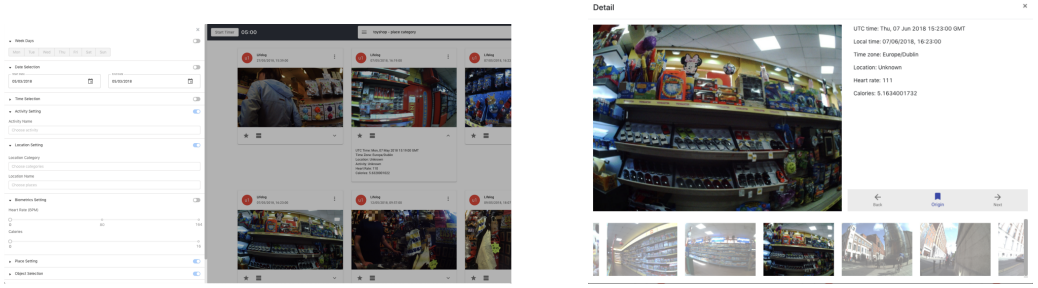
challenge also had biometric signals such as heart rate, galvanic skin response, blood pressure, etc. There were three teams participating in the LSAT sub-task including the baseline from the organizer, and all of them followed the non-interactive approach. The baseline search engine [77] adopted the basic retrieval filtering mechanism. Images were retrieved and sorted based on how many concepts from their metadata matched with the concepts in a query. A tree-based clustering technique [78] was then applied to the results to group similar images into one, hence leading to improve precision and recall. Although this concept-based search engine used a simple method, an expert user still managed to get a competitive result (second place in the challenge) since they could convert words in the query to match concepts attached to images using their domain knowledge. Meanwhile, the VCI2R team [79] expanded the number of concepts by running multiple CNN-based object detection models. Instead of faceted searching

as the baseline engine, this team applied the probabilistic Conditional Random Field [80] to find the appropriate weights for each concept and computed the weighted sum to output the ranked list of images.

The third NTCIR14-Lifelog challenge [34] witnessed an increase in the number of teams developing interactive systems. Three out of four teams used an interactive approach and obtained better scores compared to the remaining team. The dataset had the same structure as previous years but only contained 81,474 images collected in 43 days from two lifeloggers. The winning system was the HCMUS[5] team [81] following the general pipeline of an ordinary approach that employed deep learning models to enrich images with visual concepts. Each concept in a query was converted to those that appeared in the images' annotation by using their synonym words. Images were ranked based on whether they contained the concepts in the query or not. The critical component which made this team obtain the highest result was building a customized detector for lifelogger's habit. For example, they manually annotated some foods in images like "*ice cream*", "*hamburger*", or "*chip*", and then trained detectors from these annotated data. The NTU[6] team [82] also ran multiple object detectors to enlarge the list of concepts for annotation. The team did not follow the filtering mechanism which was to return relevant images if they contained the input concepts. Instead, they used the BM25 scoring function [83] to get the ranked list of retrieved images. This ranking function was originally used for document retrieval. It used term frequency and document length to measure the relevance of a document to a given query. The pre-processing stage of the NTU team was to calibrate the camera's egocentric view and remove some uninformative images with blurriness. The baseline system was developed by DCU[7] [6] had the same idea as the NTU team which was also to convert images into text based on their concept and rank the retrieved result by a similar function. However, their interactive

[5]HCMUS: Ho Chi Minh City - University of Science, Vietnam
[6]NTU: National Taiwan University, Taiwan
[7]DCU: Dublin City University, Ireland

system supported many facet filters to eliminate non-relevant images easily which was shown in Figure 2.6a. Users could either use conventional filtering panels at the left of the UI or input the text in the search bar at the top but the latter was limited to search only for visual concepts, physical activities, and location information. The input texts were expanded to include more relevant concepts through a thesaurus-lookup approach. For example, the concept "*seaside*" would be expanded with the following words: "*shore*", "*coast*", "*sands*", "*sea*", etc. The relevant images were then shown in the main center area and each of them was clickable to open the detail popup window as depicted in Figure 2.6b. This detail view displayed the information of the selected image with other images which were adjacent to it in the temporal order.



(a) The facet panels

(b) The detail view of an image

Figure 2.6: DCU team's interactive lifelog retrieval system in NTCIR14-Lifelog challenge [6]

2.1.3 ImageCLEF-Lifelog Challenge

From the success of the NTCIR-Lifelog challenge as well as to connect lifelog researchers to the image retrieval field, the ImageCLEF-Lifelog challenge series had been organized since 2017 [28, 29, 30, 31]. It used a similar dataset to the NTCIR-Lifelog task with the same types of sub-tasks which included lifelog retrieval and analyzing lifelog data. This challenge matched with the nature of ImageCLEF which was about applying deep learning models to address multimodal issues to which lifelog retrieval belongs. Since both ImageCLEF-Lifelog

and NTCIR-Lifelog shared similar sub-tasks and employed the same dataset, some teams participated in both challenges using the same systems and approaches. For instance, team VCI2R [79, 84] adopted a fundamental method that involved enhancing visual concepts extracted from images through state-of-the-art object detectors. Then the concepts in a query would be translated to annotated concepts from images by using synonyms from WordNet or based on their embedding vectors obtained from the Word2Vec. These concepts were compared to each other to measure the similarity between images and the query. Some teams used faceted searching, others used the BM25 function from the document retrieval field. The initial ranked list of images underwent further refinement process by considering additional query information, including factors such as location, physical activities, or time. In this ImageCLEF-Lifelog challenge, our attention is directed solely toward systems that introduce novel features or alternative approaches, along with distinctive user interface designs, different from those elucidated in the preceding section.

Although the UPB[8] team [85] attended the first ImageCLEF2017-Lifelog challenge with the basic approach, they introduced a new way to match concepts from a query to those from images. Since WordNet's structure was to link words into semantic relations such as synonyms, the team proposed that the similarity between two words could be computed based on how long the path was to connect them, e.g. how many edges one word needed to pass to get to the remaining word. Therefore, they employed tree-based Wu-Palmer distance [86] to measure the similarity between concepts. The AILab-GTI team [87] was the best-performing team in the second iteration of the ImageCLEF-Lifelog series in 2018. Their strategy was different from other teams since they considered this challenge as a classification problem. For each query, each image that passed the filtering process about time and locations was fed into a CNN-based binary classification model to predict whether it was relevant to the query or not. However, it required manually

[8]UPB: Politehnica University of Bucharest, Romania

selecting a small number of relevant images in order to train the classifier. There were two teams applying recurrent neural network (RNN) in the challenge [88, 89]. While Regim[9] team [88] employed the LSTM model to encode and match concepts between queries and images annotation, the HCMUS team [89] used this structure to generate descriptions for images from which they could extract the activities and enrich visual annotation. LIFER [7] was the interactive lifelog retrieval system that served as the baseline of the ImageCLEF2018-Lifelog challenge. Basically, this baseline with a faceted searching interface had the same approach as LEMoRe, but its interface was similar to Doherty's system [4]. Figure 2.7 shows that LIFER allowed users to filter more fields than LEMoRe, and the system also clearly displayed retrieved images rather than just the heatmap like LEMoRe. Users could select the concepts and other information such as time, biometric data, or location at drop-down lists put at the top of the interface. Each relevant image, which was measured by using an AND boolean operation or a filtering mechanism, is displayed below with its annotation data.

The third run of the ImageCLEF-Lifelog sequence had the most number of teams participating with nine teams submitting their solutions. The third-place BIDAL team [90] addressed the challenge by using the interactive approach that consisted of two main stages. Their first step was to group images into atomic clusters based on annotated concepts. The second step was about retrieving images in an interactive manner. First, every concept in the query was expanded using a Google Search API [10] instead of WordNet or Word2Vec models like previous approaches to get the Bag-of-Words (BoW) features which would be manually refined to get all relevant images. Then, relevant images were retrieved based on their BoW obtained from the visual concepts and those of the query. Noted that users could interactively choose the relevant images after the similarity scores were computed. The atomic clusters containing chosen images were then hidden and users kept refining the BoW features of the query to explore more images in different clusters to expand the result. While

[9]Regim: Research Groups in Intelligent Machines, University of Sfax, Tunisia
[10]https://github.com/abenassi/Google-Search-API

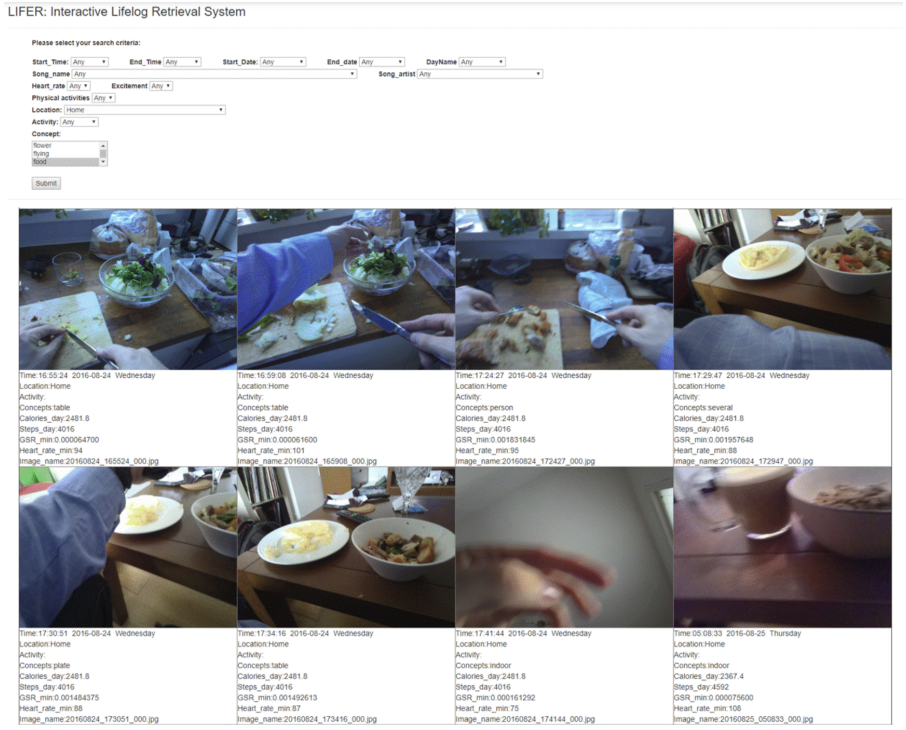


Figure 2.7: LIFER system at the ImageCLEF2018-Lifelog Challenge [7]

the second-best team ZJUTCVR [91] followed the classification solution of AILab-GTI in the ImageCLEF2018-Lifelog challenge, the best team HCMUS [8] was the previous version of them that participated in the NTCIR14-Lifelog challenge [81]. They manually extracted and annotated visual concepts from images that matched the lifelogger's habits and activities. An object detection model then was applied to train on that hand-made data to make the prediction for entire images in the dataset. Additionally, they also inferred an object's color based on its detected bounding box in an image by using K-Means clustering [92]. While previous lifelog interactive retrieval systems required users to manually select concepts based on a query description, the retrieval system from team HCMUS had an interface that could automatically transform the free-text input query into concepts that were annotated for images as illustrated in Figure 2.8. Users still could also manually refine the concepts for better results. This feature mitigated the problem of novice users who did not know what concepts were annotated in the database.

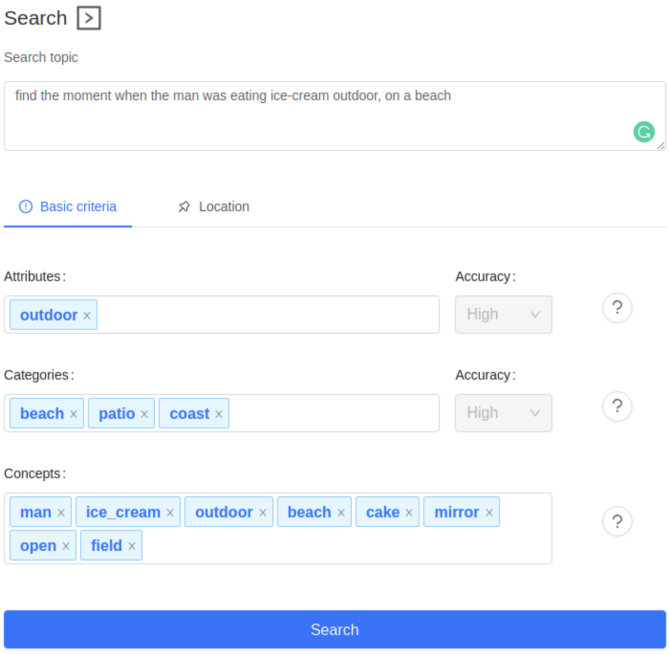


Figure 2.8: The input interface of HCMUS in ImageCLEF2019-Lifelog challenge [8]

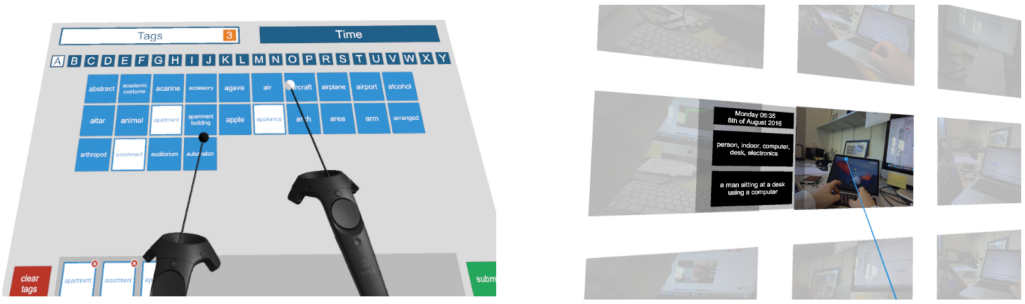
2.1.4 Lifelog Search Challenge

While the NTCIR-Lifelog and ImageCLEF-Lifelog challenges successfully contributed to lifelog research, they were not suited for assessing interactive lifelog retrieval systems. In contrast to the other two lifelog competitions, the Lifelog Search Challenge (LSC) [93, 94, 95, 96, 97] was organized in 2018 to provide a comparative evaluation of interactive lifelog retrieval systems. It featured an open, real-time, and metrics-driven assessment, requiring all systems to compete simultaneously in solving the same set of queries. Participating systems were expected to be both accurate and easy to use to be able to give a correct answer as soon as possible within a limited time. The evaluation metric used in LSC takes these criteria into account and will be discussed in detail in Chapter 3. Each system was expected to be operated in two different sessions by an expert user who developed the system and a novice user who has no knowledge about it. This was based on an assumption that if a system does not possess a user-friendly UI, the novice user will find it difficult to operate and hence get a lower score. The lifelog

dataset used in LSC was quite similar to the other two challenges. The data consists of anonymized lifelog images captured by only one lifelogger during one month (LSC'18 and LSC'19) or three months (LSC'20, LSC'21, LSC'22). The images were synchronized with timestamps and locations indicating when and where they were taken. Along with images, a set of concept annotations of each image were released. In addition, lifelogger's biometric data and physical activity such as walking or driving were also recorded. In general, most of the interactive systems in the LSC had the same approach as the LeMore system in which images were annotated with their textual, and visual concepts to enrich the information, including winners of the LSC challenges during the first four years of organizing [9, 39, 41, 42, 98]. After the CLIP [51] model was introduced and became dominant in vision-and-language challenges, many teams integrated this multimodal retrieval model into their systems and achieved competitive results in LSC'22, including the winning system [99]. In this section, I only briefly describe noticeable systems with their search engines and user interfaces in LSC'18 and LSC'19 which is before I participate in the challenge in subsequent years. I will further discuss teams that attended LSC'20, LSC'21, and LSC'22 later in Chapter 4.

The first iteration of the LSC witnessed six interactive retrieval systems participating. The most common approach was the concept-based method which was enhancing the images with their semantic visual concepts. This approach has consistently delivered positive results across various lifelog search competitions [9, 39, 42]. The top-performing UU-DCU system [9] of the LSC'18 also used this approach with the interface built in the virtual environment. Figure 2.9 shows its interface and how to interact with it. A list of pre-defined concepts was sorted in alphabet order and directly shown to searchers who could use their two controllers to select as depicted in Figure 2.9a. It was noted that the author did not use external visual concepts such as those extracted by running deep learning models but only used concepts given by the organizer in the metadata. Users could also perform filtering with the temporal information from the query. The relevant

images were then shown based on the chosen concepts with the filtering operation and users could see the metadata of these images by using the controllers for further exploration as illustrated in Figure 2.9b. The result of LSC'18 showed that this gesture-based design was efficient for both novice and expert users since they obtained relatively high scores compared to other teams [11]. Particularly, the UU-DCU system obtained 94.5 points on average, whilst the second team [10] got 92.9 points. The details of the scoring function is described in Section 3.3.1.1.

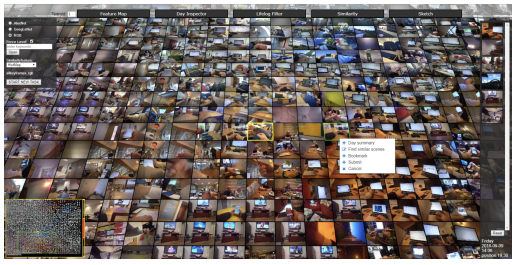


(a) Faceted interface with a list of concepts (b) Result browser

Figure 2.9: UU-DCU system in the virtual reality environment [9]

The lifeXplore system [10] obtained the second rank in LSC'18. It was the modified version of the video retrieval system diveXplore[100] participating in the annual Video Browser Showdown competition (VBS) [101] to match with the lifelog retrieval. Because it was originally built for video retrieval, lifeXplore treated a sequence of lifelog images in a day as a video having a low frame-rate playback speed. Therefore, all lifelog images in 27 days in the dataset of LSC'18 were converted to 27 videos. Then the authors applied the same techniques they used in their video retrieval system to these videos to participate in LSC'18. For each video, the pipeline started with the event segmentation step to group adjacent images into scenes by using their custom shot detection algorithm [100]. A scene was analyzed to choose one representative image, called a keyframe, to represent the scene and to be displayed in the user interface. This keyframe selection step could not only reduce the number of subsequent images which might not contain extra valuable information but also maintain the interface with less number of

images to be visualized without losing much significant information. Additionally, reducing from roughly 1,500 images to 300 keyframes in one day in the dataset could help users to find answers faster. Despite grouping images into scenes to reduce the dataset size, all visual analyses, including feature extraction using various CNN-based detectors or color histograms, were performed on every image within a scene. The features of the scene now were the summary of those from corresponding images. Based on the features, keyframes of the scenes were arranged using the hierarchical self-organizing map [102] for efficient navigation as shown in Figure 2.10a. This map of visualization was then also applied by other systems in the following LSC [103, 104]. The search engine of lifeXplore still followed the fundamental concept-based filtering approach. However, it supported an additional sketch search functionality where images were retrieved based on their color histograms to compute their similarity with a hand-drawing sketch from users. This color histogram also played a role in finding images that were similar to a selected image along with CNN-based extracted features. Users could select a keyframe in the feature map to find similar scenes or to inspect a day containing that keyframe as depicted in Figure 2.10b. This day inspector view had a grid showing all scenes of the day and scenes were represented by their representative keyframe. It showed the metadata of the scene containing food logs or visual concepts with the timestamps. Moreover, users had the option to see all images in the day in the form of video playback.



(a) Feature maps displayed by using self-organizing map



(b) Day inspector view

Figure 2.10: User interface of LifeXplore system in LSC'18 [10]

The VIRET system [105] achieved third place in LSC'18. Similar to lifeXplore, this system was also initially developed to address the video retrieval challenge and it was the winner of the VBS'18 competition. Hence, it had a quite similar approach to the competitor which was to convert lifelog images into low frame-rate video for each day in the dataset. It was followed by the event segmentation stage and representative keyframe selection which was to select the middle image of the event. They retrained the GoogleNet on their own ImageNet dataset possessing 1,390 labels to enhance the visual annotation for each image. The feature vectors obtained from the retrained GoogleNet [69] were utilized for similar image searching like lifeXplore's functionality. VIRET and lifeXplore both allowed users to search by sketching. Although they supported full-text search, VIRET applied WordNet to modify the input concepts into synonym concepts that were in the annotation. Moreover, VIRET measured similarity not only based on AND operation but also on OR operation depending on users' preferences. Unlike other systems, the VIRET system did not take the location or biometric data into account since their main purpose of participating in LSC was to investigate the performance of a retrieval tool specifically designed for video retrieval on lifelog data. Figure 2.11 shows the interface of VIRET in LSC'18 in which the input area was at the left and the right was for visualizing the result. This area also allowed users to sketch a color palette to find photos having similar color structures. The bottom left area was to show the selected frame in a bigger view and each photo was attached with the time information.

Three remaining systems that participated in LSC'18 were: the LIFER system [98], the UPC-DCU system [106], and the VNU-HCM system [107]. The LIFER and the VNU-HCM systems were the original frameworks that their inherited systems [7, 89] participated in ImageCLEF2018-Lifelog Challenge that we have described in the previous section. The UPC-DCU system was developed by Universitat Politecnica de Catalunya (UPC) and Dublin City University (DCU). The system considered this lifelog retrieval challenge as a document retrieval problem. Its main feature was



Figure 2.11: VIRET system in LSC'18 [11]

the ability of free-text searching. It meant that they used the conventional TF-IDF [38] scoring function that is normally used in the document retrieval field to match concepts of images to the texts input by searchers.

LSC'19 which was the second run in the series witnessed nine teams participate, including several systems updated from previous LSC [108, 109, 110]. The VIRET system [109] with the extended faceted options and the inclusion of lifelog metadata including locations and biometric data managed to claim first place in LSC'19. The second-best team in LSC'19 was Vitrivr [41]. Vitrivr was a content-based multimedia retrieval tool supporting different types of data including video, images, and audio. The system also participated in the VBS challenge and obtained the highest score in the video retrieval competition in 2017 and 2019. They ran multiple deep learning models on each image to obtain visual concepts, activities, captions, and text that appeared in the image (OCR). With these annotations, the system used a similar solution to VIRET since they applied the filter with AND/OR boolean function to find images based on the concepts parsed from the given query. Additionally, Vitrivr allowed the query-by-sketch like the competitor system. The novel system in LSC'19 was the Exquisitor [12] which proposed a searching mechanism distinct

from others. The system could analyze the relevance feedback from users from which the system can retrieve relevant images without any input query by using an interactive learning approach. Each image was embedded into a vector based on their visual concepts and semantic activity descriptions [111]. At each iterative process, a linear SVM classifier model was trained with the vectors of relevant and non-relevant images decided by searchers to produce a list of new suggestion images and shown to users, as shown in Figure 2.12. Starting with several initial random images, users of Exquisitor would make a judgment of which images were relevant and which were not relevant to the given query by clicking icons at the bottom-right or bottom-left of the image, accordingly. A trained linear SVM model then suggested a list of new images and showed them to users in order to start a new iteration. This process kept repeating until users found the correct answer. This interactive learning approach with relevance feedback was proven to be scalable for large datasets since each new suggestion only costs less than a second for a database with 100 million items [112] with a single CPU. Particularly, it took only 20 milliseconds to train a model and produce new images for the LSC'19 dataset containing less than 42,000 images.



Figure 2.12: Exquisitor system in LSC'19 [12]

Regarding the remaining teams in LSC'19, while the HCMUS team [110] inherited the core features from its previous system in LSC'18, LifeSeeker[113] was the new system in the LSC. They had competitive scores compared to each other and obtained third and fourth places, respectively. In fact, both systems and the NTU system [114] were their previous versions that participated in the NTCIR14-Lifelog Challenge we discussed in the previous section [6, 81, 82]. The VieLens system [115] followed the basic concept-based filtering approach similar to an ordinary system. Nevertheless, they employed the BERT model [50] instead of using WordNet or Word2Vec embedding model to convert the input concepts with those in the database to perform the retrieval.

2.2 Image-Text Retrieval

Compared to lifelog retrieval, image-text retrieval is a more popular challenge with a wide range of various approaches has been proposed. Nevertheless, they all have to transform items from different modalities, including images and texts, into a common vector space in order to be able to measure their similarities to perform the retrieval. Since CNN and RNN models were introduced along with more datasets were released [43, 48], the image-text retrieval challenge became more attractive to researchers, leading to more state-of-the-art results have been achieved. Although many retrieval models had different approaches to the challenge, they could be classified into two groups: the independent embedding approach, and the cross-embedding approach. Figure 2.13 shows the difference in the pipeline of the two most common categories for image-text retrieval.

2.2.1 Independent Embedding Models

An independent embedding method is an approach in which each modality is embedded into vectors by a separate encoder. Particularly, an image encoder and a text encoder will be used to encode an image and a caption into vectors from which

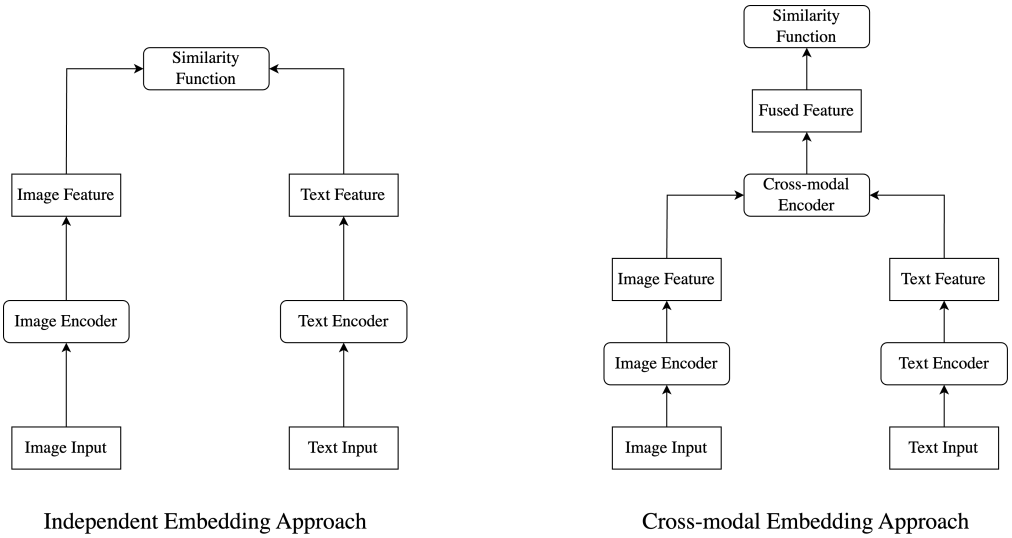


Figure 2.13: Two most commonly used pipelines of image-text retrieval models

a similarity function, such as inner product, can be used to measure their similarity score. DVSA[11] [116] was one of the first models using deep models as encoders in their method following this approach. Their assumption was that an image was considered relevant to a caption if the image contained objects which were mentioned in the sentence. They applied the Region Convolutional Neural Network (RCNN) [117] to detect object regions that appeared in an image. The top predicted regions, which had a high prediction score, with the whole image were then fed into an ordinary CNN to embed into vectors. Bidirectional Recurrent Neural Network (BRNN) [118] was employed to transform each word in a sentence into a vector. The image was relevant to the description if the sum of the dot product between encoded regions and encoded words was higher than other pairs of image-text in the dataset. They trained their model using the triplet loss in contrastive learning. This loss function helped optimize the embedding space by minimizing the distance between relevant image-text pairs while maximizing the distance between irrelevant pairs. In contrast, VSE[12] [119] encoded a whole image into a vector using VGG or ResNet [120], and did the same to a caption with the

[11]DVSA: Deep visual-semantic alignments
[12]VSE: Visual Semantic Embedding

Gated Recurrent Unit (GRU) model [121]. They also used the triplet loss function to train their model, however, they proposed the hardest negative mechanism integrated into the loss. Instead of using all negative (irrelevant) pairs of image-text in the loss function, they only chose the most irrelevant pair in the mini-batch to use as a negative sample in the triplet loss. It not only helped to enhance the accuracy but also improved the computational efficiency. This hardest negative triplet loss function has shown its advantages and is still adopted in many SOTA retrieval models today. PFAN[13] [122] was another model starting with detecting objects in images. This model employed Faster-RCNN [123] to detect regions and then updated their features with ResNet. They introduced the position-focused attention mechanism where they concatenated region features with their position features obtained from the attention network. Then all concatenated visual features were projected into a unified vector representing the whole image. Regarding the text branch, a bi-directional GRU [124] was used to get the textual feature of a caption. The training protocol of PFAN was similar to VSE and they adopted the latter's proposed loss function. DPC[14] [125] was a retrieval model that embedded an image or a sentence into a single vector by proposing a dual-path CNN method in which two separated CNN models were used to serve as visual and textual encoders. Regarding the training stage, they introduced a new instance loss, which was based on the basic classification loss, in addition to the triplet loss for training the model.

2.2.2 Cross-embedding Models

Compared to the independent embedding pipeline, the cross-embedding approach takes the interaction between images and descriptions into account, hence exploiting important features for image-text matching to perform the retrieval. This pipeline not only focuses on salient image features and text features separately but also the cross-modal information. This pipeline has become popular

[13]PFAN: Position focused attention network
[14]DPC: Dual-path convolutional

since the attention mechanism and the transformer structure were introduced [49]. This was because the attention model was appropriate to fuse the information from the image and the information from the sentence. SCAN[15] [126] was one of the first models following this approach although it had the same idea as DVSA when they both calculated the similarity between the image and text sentence based on visual regions and words. While DVSA computed the corresponding score of a region and a word solely on themselves, SCAN included all regions and words in the process with their learnable weights using the attention procedure. On the contrary, CAMP[16] [127] leveraged attention networks to augment both image and text features bidirectionally. This model viewed image-text retrieval as a straightforward classification problem, merging the enhanced features from both modalities and inputting them into a linear layer to predict the relevance of an image-text pair. To train the model, a hardest negative binary cross-entropy loss function was employed. Despite these advancements, the aforementioned models retained a certain limitation. Although they updated an image region feature with regard to every single word in a sentence through an attention process, it underlooked the intra-information between regions themselves. The same was true for the other way. UNITER[17] [128] addressed the drawback by using a transformer structure. All regions in the image and all words in the sentence were considered tokens in one unified sequence fed into the transformer network. The transformer model would learn the cross-modal embedding features from both visual and semantic features. Its outcome *[CLS]* token feature was a fused representative feature of both modalities containing the inter and intra information between regions of the image and words of the description. Moreover, UNITER could be applied to other vision-and-language tasks such as visual question answering, visual commonsense reasoning, or referring expression comprehension. This was because UNITER was pre-trained with multiple loss functions on external datasets and

[15]SCAN: Stacked cross attention network
[16]CAMP: Cross-modal adaptive message passing
[17]UNITER: Universal image-text representation

these functions could help to mitigate the differences in learned features for different downstream tasks. One of the loss functions that UNITER employed to solve the image-text retrieval task was the hardest negative binary cross-entropy where the representative *[CLS]* feature for a pair of image-text would be classified as a relevant or irrelevant pair. OSCAR[18] [129] had a similar idea as UNITER when they both utilized the transformer on the joint sequence of visual regions in an image and a sentence. Nevertheless, while UNITER concatenated only visual regions and words in the sentence, OSCAR extended it with the inclusion of semantic tags of the objects in the regions. In specific, a pair of image-text was transformed into a sequence of embedded tokens, including embedded words in the caption, embedded words of the object tags, and the visual features of regions, before being fed into the transformer model to get the *[CLS]* feature for other tasks. Although using fewer loss functions in the training stage, OSCAR obtained better accuracy than UNITER and was considered to be the SOTA cross-modal retrieval model at that time.

2.2.3 Fusion Models

The cross-embedding models have demonstrated their superiority in cross-modality learning, outperforming models that follow an independent embedding pipeline. However, a persistent challenge is their inability to be applied in real-time. For example, UNITER, with approximately 303 million parameters, requires a considerable amount of time—more than 12 seconds for each query on a dataset containing 30,000 images [130] to execute the retrieval process. This is because the cross-modality encoder needs to process every pair of image-text in the database whenever a query is provided. Meanwhile, the independent embedding models can encode images and descriptions in the database offline. The encoder model only needs to process the query (an image or a sentence) once and significantly reduces the retrieval time. Some works have been proposed to overcome this

[18]OSCAR: Object-semantics aligned pre-training

time-consuming issue [130, 131]. They had the same solution which was still using the cross-modality transformer structure. However, they did not use the final *[CLS]* feature for the retrieval yet. Instead, the embedded features of each modality, which were already updated through the cross-attention procedure, would be used to make the preliminary retrieved result. This initial result was then re-ranked by the *[CLS]* feature obtained from the cross-modality transformer network, which was learned in the training stage, to produce the final ranked list for the image-text retrieval task.

Miech et al. [131] introduced the *Fast and Slow* framework that combined the advantages of the independent embedding and cross-embedding groups. They used two separate encoders to get representative features for an image and a sentence (ResNet as an image encoder, and BERT as a text encoder). They selected the VirTex model [132] as their cross-modality encoder. Additionally, they introduced a novel loss function to facilitate knowledge distillation from the joint encoder to the dual encoders, aiming to enhance the modality representative features. The knowledge distillation technique helps the model transfer knowledge from a larger, more complex model to a smaller model, enhancing the student's performance and generalization capabilities. In the Fast and Slow model structure, it serves to refine and improve the modality representative features learned by the dual encoders. These features were used to get the initial retrieved ranked list of relevant items which was a subset of the large database before being refined by the VirTex network later to get the final output. LightningDOT [130], on the other hand, did not employ the multimodal transformer component, but introduced three loss functions that took the *[CLS]* features from both modalities as input to encourage the model to learn the fused information between them. Their experiment showed that LightningDOT with the UNITER as a post-processing component achieved higher accuracy than the original UNITER while being faster than 500 times.

2.2.4 Pre-training on large dataset

Although the fusion method did improve the performance, it was not until the CLIP model was released that a new generation of multimodal retrieval models has been introduced. CLIP followed the independent embedding pipeline when it integrated Vision-Transformer or ResNet as its image encoder, and the ordinary transformer as a text encoder. The features obtained from both encoders were then projected into the joint-embedding space using a simple linear layer. CLIP was trained with the contrastive learning procedure to ensure feature vectors of a relevant image-text pair would be close to each other whilst irrelevant pairs remained further away. However, the vital point that made CLIP become a game-changer in the vision-and-language field was the pre-training stage. While previous image-text retrieval models were trained on specific datasets for the retrieval task or were pre-trained on external datasets containing roughly 10 million image-text pairs [128, 129], CLIP was pre-trained on the massive newly created dataset of 400 million pairs which was considered to sufficiently cover a general set of visual concepts in the real world use cases. This huge and diverse dataset helped CLIP improve its generality and robustness. CLIP with the zero-shot version, which was not finetuned on downstream tasks, was able to obtain competitive or even better results in 30 different existing computer vision datasets. The author of the ALIGN model [133] argued that the dataset used in the pre-training stage of CLIP required costly work on data curation and data cleaning. This challenge could pose a hurdle to constraining the dataset size, consequently limiting the achievable accuracy. Therefore, ALIGN introduced a larger dataset for the pre-training stage. This dataset contained more than 1.8 billion pairs of image-text. However, they avoided the complex and heavy task of data cleaning and made the dataset noisier than the one used in CLIP. The ALIGN model leveraged this noise to obtain better representations of images and texts, hence achieving the new SOTA results. This model followed the independent embedding pipeline as CLIP did but ALIGN used EfficientNet [134] to encode images and BERT

to encode texts. These encoders were learned using a softmax loss in which relevant pairs should have the dot products of their features higher than those of irrelevant pairs. Their experiments showed that ALIGN obtained slightly better results than CLIP on the clean dataset such as Flickr30K. Nevertheless, regarding the noisier dataset like MSCOCO [43], ALIGN achieved an increase in accuracy compared to CLIP due to its pre-training stage on the noisy dataset.

Despite CLIP and ALIGN achieving state-of-the-art results in both image-text retrieval tasks and other vision-and-language challenges through pre-training on a large, noisy dataset, they did not harness the advantages of the cross-embedding transformer component. Additionally, their model did not consider any technique to confront the harmfulness of a noisy dataset, which can be a case of an image containing an entity that is not mentioned in the text or vice versa. ALBEF [135] proposed a novel method, called Momentum Distillation, to improve the learning capability of the model under noisy supervision. This technique prevented the model from being penalized if it produced a reasonable prediction that is not a ground truth in the noisy dataset. Moreover, ALBEF adopted the fusion approach when it used three transformer-based architectures in which a Vision-Transformer was the image encoder, a BERT was the text encoder, and another BERT was the multimodal encoder at the end. The fusion structure with the momentum technique helped ALBEF to achieve a competitive result compared to ALIGN although being pre-trained on a much smaller external dataset containing 14 million pairs. BLIP [136] was developed by the same authors of ALBEF, and can be considered to be the upgraded version of ALBEF. It introduced another technique to address the noise issue in the large dataset for the pre-training stage. They proposed the caption bootstrapping method in the pre-training process in which a text decoder was used to keep generating a synthetic caption for an image and a text encoder would classify whether a synthetic or a ground truth caption was a noise or not to perform the filtering and remove it out of the dataset. This encoder and decoder were trained along with their ordinary image-text retrieval model to enhance the quality of the

noisy external dataset without a need for a costly data cleaning procedure. BLIP managed to gain the new state-of-the-art results on most of the vision-and-language tasks by either pre-training on a medium (14 million pairs) or large (129 million pairs) dataset.

2.3 Graph-based Retrieval Models

Many graph-based retrieval models have been introduced for the image-text retrieval task. Shi et al. [54] applied this structure to expand the detected objects in images to get more relevant concepts from external knowledge databases. This method could help to deal with the issue of missing visual concepts, which appeared in an image, in the sentence query. However, it could be a problem if the expanded concepts were not matched to the sentence, leading to a lower performance of the model. Therefore, the authors included the sentence to this expansion stage to constraint generating noisy concepts. This model, thus, can be categorized as a cross-embedding approach. Cosine similarity was used to measure the similarity between visual features extracted from the concepts and the semantic features produced by applying a recurrent neural network to captions. Cosine similarity is a metric commonly used to calculate the cosine of the angle between two vectors. In the context of a retrieval task, it measures how similar the visual features extracted from concepts are to the semantic features produced by the recurrent neural network applied to captions. A cosine similarity value of 1 indicates perfect similarity, while 0 denotes no similarity. Therefore, the cosine similarity score was applied in many models to perform the retrieval. A graph structure can be employed to capture robust information about actions occurring within the data, including the interactions between entities/objects in images and descriptive queries. The scene graph structure [137] was proposed to represent such information, thus offering more in-depth insights into visual data as well as textual queries and enhanced image retrieval. This pioneering work introduced a manually

annotated dataset with scene graph structures describing interactions in images and used this graph-based data as a query to perform the retrieval. This preliminary model used the conditional random field and the maximum a posteriori as a similarity function to find relevant images. Extending that research, the SGM[19] model [53] generated scene graphs from both images and captions as input using any pre-trained scene graph generation model [138, 139]. Following the independent embedding pipeline, the model then extracted the graph features by using their own designed encoders for each modality. Regarding the image encoder, SGM used Faster-RCNN to obtain the objects' features and the feature of their union regions representing their interactions in the image. The semantic tags of the objects were also encoded with the fine-tuned Word2Vec. A Graph Convolution Network (GCN) [140] was applied to fuse the semantic tags and the visual object features to obtain the unified features for the visual graph. On the other hand, they employed Bi-directional GRU to encode the whole sentence and each of the triplet of interactions in the sentence to get the textual graph. Then, the similarity score was calculated between the two graphs by matching each part of the image's graph with every part of the caption's graph. Meanwhile, instead of using a scene graph, the GSMN[20] [141] constructed a dense graph for an image in which all concepts were connected to each other. Similar to the SGM, the visual graph was then compared to the textual graph by matching pair-wise nodes between two graphs. Nevertheless, GSMN integrated the cross-attention mechanism in this matching stage to enhance the corresponding information between modalities. Recently, the SGRAF[21] model [142] also utilized a graph structure along with a cross-attention network to learn the similarity between visual concepts in an image and words in a textual query as well as their interactions. In addition, another similarity cross-attention module was designed to assess the meaning of this information and refine the prediction. GSMN and SGRAF enhanced the

[19]SGM: Scene Graph Matching
[20]GSMN: Graph Structured Matching Network
[21]SGRAF: Similarity Graph Reasoning and Attention Filtration

performance of graph-based retrieval models in the image-text retrieval challenge through the cross-embedding approach. Nevertheless, this approach poses limitations on their effective use in real-time scenarios.

Although there have been many attempts to integrate graphs into image-text retrieval, this structure has not yet been commonly applied to the lifelog retrieval field. The LifeGraph system [143] in LSC'20 applied a knowledge graph structure with the nodes representing detected objects or scenes recognized in images. These entities can be linked with corresponding images and external sources [144, 145] to expand the information with frequent activities and relevant objects. This mechanism helped the system to find more images related to the concepts mentioned in a textual query. Meanwhile, another graph-based lifelog retrieval system in LSC'20 was the NTUnlp system [146]. This system applied the scene graph structure to capture the relative position of objects in lifelog images as those proposed by Johnson et al. [137]. However, it was just an additional option in the system in which the scene graph was used as a part of a scoring function and not described in detail. The performance of these systems in LSC'20, as discussed in Section 4.2.4.2, indicates that the full potential of the graph-based structure was not fully realized, leading to considerably low scores.

2.4 Chapter Summary

This chapter has presented the literature review for our research work in this thesis, particularly an overview of many noticeable lifelog retrieval systems. The user interface of lifelog retrieval systems has been revised through time to match the ultimate goal of helping lifeloggers to facilitate recalling a specific moment or event in their life captured through images. Although, some novel retrieval models have been proposed with the systems including relevant user feedback or sketching, the most commonly used solution is still the traditional concept-based method. This type of search engine, albeit simple, has obtained good results in many lifelog

retrieval challenges. However, it overlooks the detailed information besides the concepts such as the interaction between objects or the activity that happened in the images. Many SOTA models in the image-text retrieval task, which is a broader challenge than the lifelog retrieval, are also reviewed in this chapter. Therefore, there are many methods with various approaches have been introduced in this task compared to lifelog including those using graph-based structures to capture the interactions information in images. Nevertheless, the introduction of several retrieval models with their pre-training strategy on massive data on the internet led to a new milestone in the image-text retrieval task. Although obtaining remarkable accuracy, these models required considerable resources (GPUs) to train them on huge datasets. Consequently, addressing the highlighted challenges, I proposed a novel solution that leverages graph-based structures to formulate an advanced state-of-the-art cross-modal retrieval model in image-text tasks, with the potential applicability in the context of lifelog retrieval. Detailed exploration and implementation of this solution will be presented in the subsequent chapters.

Chapter 3

Research Methodology and Evaluation Methods

The two previous chapters gave an introduction to what lifelog retrieval and image-text retrieval is. I also summarized several works in these retrieval fields and their current limitations from which my hypothesis and research questions were proposed. This chapter describes the research methodology along with the research workflow that I employ. Additionally, the evaluation methods including the evaluation metrics that will be used in the research questions are also discussed.

3.1 Research Methodology

Research can be considered to be a logical and systematic investigation of seeking a novel and useful solution for a question on a particular topic [147]. This investigation is a process of a comprehensive sequence of steps that are diligently undertaken to acquire, gather, and meticulously analyze a multitude of information sources, with the ultimate goal of improving our knowledge in a particular topic or issue [148]. There are many different approaches to forming a design for this process that were recapped in Creswell's work [148]. Based on their finding, we propose and follow the research paradigm as shown in Figure 3.1 to ensure the systematic and logical investigation. The paradigm provides a structured framework that facilitates the orderly execution of the research process, from the indentification of research questions to the analysis of gathered information, ultimately contributing to a robust

and well-founded exploration of the chosen topic.

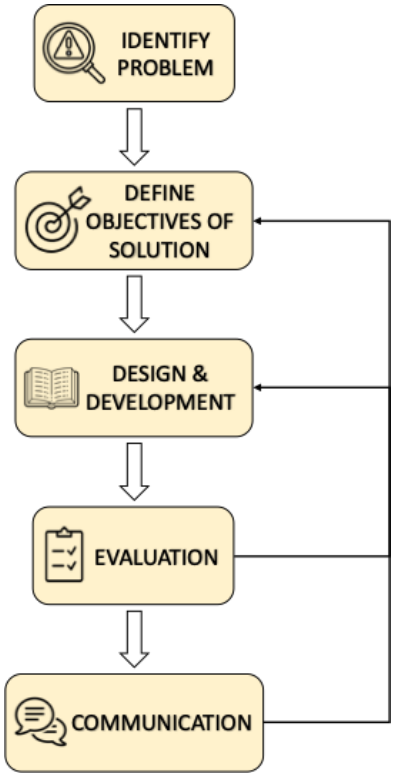


Figure 3.1: Research methodology schema

I start my research with problem identification in the main research domain, which is lifelog retrieval. I recognize that most of the current solutions for the lifelog retrieval task are developing a lifelog retrieval model following a conventional concept-based approach. Although obtaining good results in many lifelog retrieval competitions, this fundamental type of search engine suffers from the problem of underestimating other information besides only visual concepts. After conducting a literature review in related research domains, I realize that newly introduced embedding-based techniques have been adapted to the image-text retrieval task and achieved outstanding accuracy, including those using graph-based structures to capture relations between concepts in images. Therefore, I construct the hypothesis which is "*Applying graph-based structure into a lifelog interactive retrieval system can help to increase the performance in terms of speed and accuracy compared to conventional concept-based interactive retrieval systems*".

Three research questions are formed to either prove or disprove our hypothesis, as mentioned in Chapter 1. Each research question is systematically explored, and the proposed solution is rigorously evaluated through experiments, seeking to ascertain its ability to achieve the anticipated outcomes. The results will shed light on the validation of a portion of the hypothesis. In case the answer cannot be reached, I will repeat the whole process with the revised research questions or even the modified hypothesis. Subsequently, I will either prove or disprove the hypothesis by addressing the research questions through one or several cycles of the process.

Specifically, I carry out the following experiments with the workflow illustrated in Figure 3.2 to resolve the research questions:

- Given that the most common search engine used in many lifelog retrieval systems is matching concepts between images and semantic queries, I decided to develop an interactive lifelog retrieval system following this approach to serve as the baseline in my hypothesis. This system served as the baseline for my hypothesis and is assessed in various lifelog search competitions. Detailed insights into this system and its contribution to answering RQ1 are provided in Chapter 4.

- Since there are not many graph-based retrieval models developed for lifelog retrieval, it is a challenge to have a SOTA graph-based lifelog retrieval model when there are no reasonable systems to be a baseline. Therefore, I propose that constructing a graph-based model for the broader image-text retrieval task would be a more fitting approach. I introduce two retrieval models using graph-based structures. While the former model utilizes this data structure to capture the interaction between concepts, the latter model applies it to create a unified model by fusing any available models, leading to obtaining better performance. Both proposed models are compared to the current SOTA models at the time being introduced by evaluating them on the most popular dataset in the image-text retrieval challenge, including Flickr30K[48] and MSCOCO[43].

The detail of the implementation and results are described in Chapter 5 which is the answer for RQ2.

- I implement state-of-the-art graph-based models in the context of lifelog retrieval. First, I conduct a pilot study in which I only focus on the accuracy of retrieval models, and other factors are not taken into account. Then I design a meticulous user study in which I integrate my SOTA graph-based model into the system developed in RQ1 and compared it to the original concept-based system. It ensures that the interface of a lifelog retrieval system plays no role in the comparison. Eight volunteers participate in the user study to evaluate two systems in terms of accuracy and retrieval time. The experiment's configuration is designed to eliminate the learning bias from users. The detail of my experiments is discussed in Chapter 6. The result of the experiments also resolves the last RQ3 and fulfills my answer to the main hypothesis.

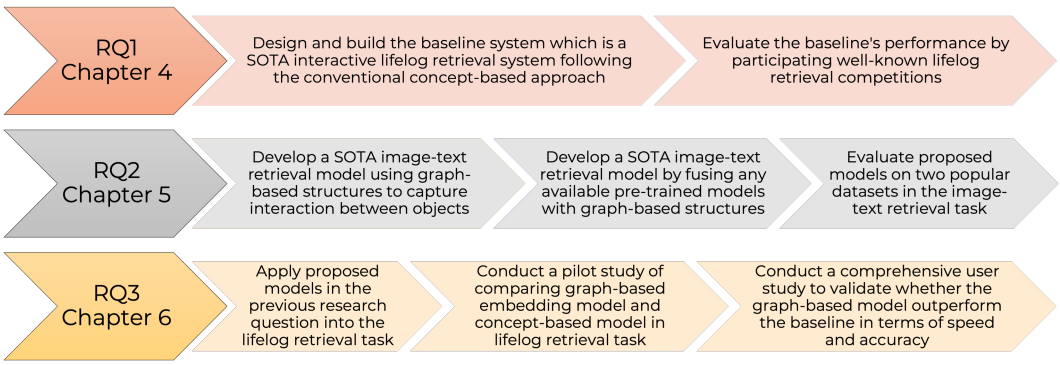


Figure 3.2: The workflow of research questions and approaches

3.2 Operating Constraints

Whenever a new topic of research is approached, some constraints remain. Therefore, I here identify these constraints in my research as follows to design and conduct the experiment properly:

- My research domain is heavily dependent on lifelog data, which is automatically captured from a lifelogger for a long period of time. Thus, the lifelog data is assumed to uphold the privacy policy and adhere to the prevailing data governance regulations. It is crucial to ensure that ethical approval, granted by the Ethical Research Committee of the respective research institution, is obtained in order to conduct research using the lifelog dataset.

- Any lifelog query from lifelog data should be generated by the lifelogger, or by those who are authorized and trained by the lifelogger.

- The baseline system developed in RQ1 and used in the comparison in RQ3 should be a reasonable SOTA lifelog retrieval system that is proven through its performance in many lifelog search challenges.

3.3 Evaluation Methods

I employ different evaluation metrics specialized for each research question, which are described as follows:

3.3.1 Evaluation metrics for a lifelog retrieval system

A lifelog retrieval system using the fundamental concept-based search engine is developed to address RQ1. Therefore, it needs to have appropriate metrics to evaluate the system to prove it is the SOTA system in the lifelog retrieval task. I validate that by taking part in several lifelog search challenges, including LSC and ImageCLEF-Lifelog.

3.3.1.1 Scoring function in LSC

Regarding the LSC, this benchmarking is designed for interactive lifelog retrieval systems and assesses participating systems based on two criteria which are **accuracy** and **retrieval time**. While the accuracy criterion aims to evaluate how good a search engine in a system's backend is, the latter criterion ensures that a system should have

a user-friendly interface that facilitates finding an answer in a time-limited manner. The scoring function used in the LSC is illustrated in Equation (3.1). It shows a point awarded to a system for each query if its user successfully solved the query in which ω is the number of incorrect submissions, T is the maximum allowed searching time, and τ is the retrieving time of the user.

$$S = max(0, 100.\frac{T.0.9^{\omega} - 0.5.\tau}{T}) \tag{3.1}$$

The scoring function indicates that the maximum point that the user can get is 100 points for each query. However, it will linearly decrease to half (50 points) over the allocated searching time if the user does not make any wrong submission ($\omega = 0$). Furthermore, the maximum point possible will also decrease 10% for each incorrect submission. There are two distinct evaluation sessions in the LSC that are for expert users and for novice users. The allocated retrieving time T, thus is different between sessions. Expert users are expected to use less searching time ($T = 180$ seconds) than novice users ($T = 300$) due to the gap in their domain knowledge about lifelog retrieval and the systems. I also apply this metric in the final user study experiment in RQ3.

In addition to the scoring function, the **Precision** and **Recall** metrics are also usually used in the LSC challenge for a deeper analysis. Particularly, the *Precision* is defined as the percentage of the total submissions submitted by a system that is correct. Meanwhile, the *Recall* of a system is calculated by the number of solved queries divided by the total number of queries in the competition. The precision and recall metrics used in LSC are quite different from the metrics described in the following section. Hence, I will mention the definition of the metrics again in each circumstance for better clarification.

3.3.1.2 Metrics in ImageCLEF-Lifelog

The proposed interactive lifelog retrieval system, although being designed for finding only one single correct lifelog image as fast as possible, also participated in the ImageCLEF-Lifelog task in order to evaluate how the system performs on finding all relevant images. Since the lifelog camera can capture an image every 30 seconds, most of the images that are captured in the temporal adjacency are visually similar to each other. It is redundant for the ImageCLEF-Lifelog task's organizer to score only based on adjacent images, especially when the task requires a retrieval system to seek all images that match a given query. Instead, the evaluation is conducted based on the number of clusters identified by the systems. A cluster, defined as a group of temporally adjacent images, is the focal point of assessment. ImageCLEF-Lifelog retrieval challenge introduces **Cluster Recall at K (CR@K)** which is a metric that assesses how many distinct clusters in the ground truth are found among the top K retrieved images. Additionally, **Precision at K (P@K)** measuring how many retrieved images in the top K result are relevant to the query is also used in this task. Last but not least, **F1-measure at K (F1@K)** is the harmonic mean of them. These metrics ensure that a good system needs to find not only relevant images (high P@K) but also their generality since they have to belong to different clusters (high CR@K).

3.3.2 Recall@K metrics for an image-text retrieval model

Regarding the image-text retrieval task, the most commonly used metric is the **Recall at K (R@K)** which is all used in other SOTA models in this domain [51, 128, 129, 133, 136]. This metric assesses the percentage of queries for which a retrieval model correctly identifies the relevant output in the top K of the retrieved ranked list:

$$R@K = \frac{1}{N_q} \sum_{q=1}^{N_q} \mathbf{1}(q, K) \tag{3.2}$$

where N_q is the number of queries and $\mathbf{1}(q, K)$ is a binary function returning 1 if the model finds the correct answer of the query q in the top K of the retrieved output. It is noted that since an image contains five relevant descriptions in the dataset, the binary function $\mathbf{1}(q, K)$ still returns 1 if only one correct text appears in the top K result and there is no need to have all of five ground truth in the result. This configuration is applied in all image-text retrieval research. Similar to other research, I use R@1, R@5, R@10, and R-Sum, which was the sum of them, in the experiment.

3.3.3 Mean Reciprocal Rank used in the pilot study

Mean Reciprocal Rank (MRR) is an evaluation metric that is usually used in information retrieval. We choose this metric for our pilot study in RQ3 because it is sensitive to the ranking position, which is also the main criterion in our assessment of a retrieval system's accuracy. A high value of MRR indicates that a correct image is found at the top of a retrieved result. The MRR is calculated as the following equation:

$$MRR = \frac{1}{N_q} \sum_{q=1}^{N_q} \frac{1}{\mathbf{rank}_q} \tag{3.3}$$

where N_q is the number of queries, and $\mathbf{rank}_q$ is the rank of the first relevant image in the returned ranked list, regardless of other relevant images in the list. The term $\frac{1}{\mathbf{rank}_q}$ is called reciprocal rank and is considered to be 0 if there is no relevant item in the list. My pilot study applies MRR over the top 100 images returned by the participating systems.

Chapter 4

Baseline Lifelog Retrieval System

4.1 Introduction

This chapter will give the answer for my RQ1, which is to find out what is the state-of-the-art interactive lifelog retrieval system following the conventional concept-based approach. Furthermore, my findings in this chapter will be used as the baseline system in the experiment which will be discussed in Chapter 6. In this chapter, an interactive lifelog retrieval system, called MyScéal (from the Gaelic word *Scéal* meaning *Story*), is developed and uses the concept-based search engine as the backend with a user-friendly user interface. MyScéal is evaluated by participating in many well-known lifelog retrieval challenges. Through the performances in these competitions, some system's disadvantages and advantages are analyzed to update the system not only the interface but also the retrieval model to enhance the performance for the next iteration of the challenges and ensure it is still a SOTA concept-based lifelog retrieval system. Figure 4.1 depicts the versions of MyScéal and the challenges it participated in.

It is noted that the MyScéal system and its updated versions are the results of the collaborative work between my colleague and me with each of us making equal contributions [39, 40, 42, 149]. Therefore, I will refer to "we" in this Chapter since I am working as part of a team to build this baseline lifelog retrieval system. Specifically, I play a key role in workflow and user interface design, employing deep learning models and cloud services for extracting visual concepts and features. I also contribute to query expansion using a rule-based part-of-speech tagging algorithm

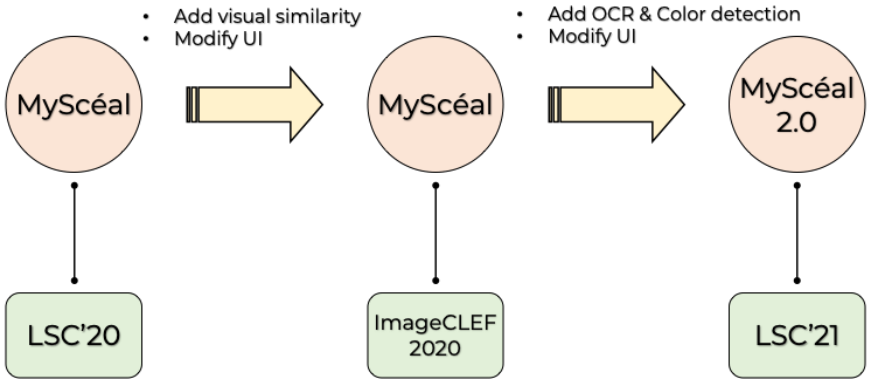


Figure 4.1: Cycle path of our baseline concept-based system

and perform data indexing with ElasticSearch.

4.2 MyScéal at LSC'20

As shown in Figure 4.1, MyScéal was initially designed for participating in the LSC challenge which is an annual comparative benchmarking activity for comparing proposed solutions to interactive retrieval from multimodal lifelog data. Our main target is the LSC because the interactive manner is the appropriate approach for addressing the retrieval task on a large dataset like lifelog [1]. After reviewing systems in the previous LSC series and considering the challenge's evaluation criteria, we chose to design MyScéal as a non-faceted interactive lifelog retrieval system, featuring a concept-based search engine and a simple UI. This design aims to uphold three essential principles related to the scoring function detailed in Chapter 3: speed, accuracy, and user-friendliness.

Figure 4.2 illustrates the schema of MyScéal which is also a typical pipeline of a basic lifelog search engine [150]. Our concept-based search engine was similar to other methods which required annotating each image with its semantic tags in order to perform the retrieval. Initially, we extracted textual features from images in the dataset, which consist of semantic locations, activities, times, and visual concepts obtained by using deep learning descriptors, to get their annotations. This process is described in detail in Section 4.2.1.1. We then created an inverted index in the

underlying Elasticsearch engine [151]. Moreover, the GPS data from the locations and the time information in the metadata were also stored in this database. Following previous research, we also extended these annotations to enrich the indexed data. Users can use a simple interface to perform interactive retrieval and the results of any query are displayed in a format of a ranked list from which users can select and find the correct answer.

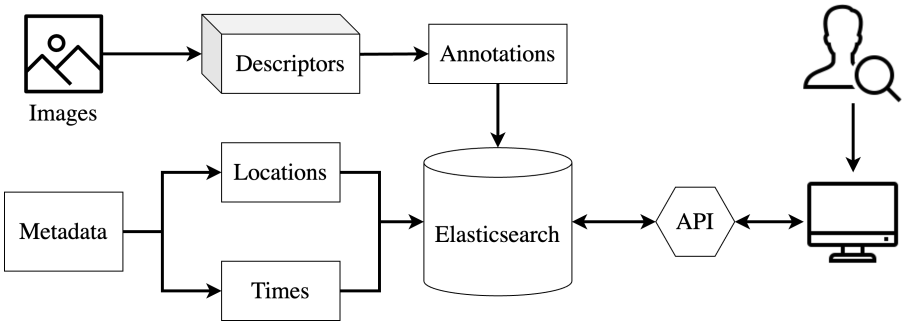


Figure 4.2: MyScéal system's pipeline

4.2.1 Concept-based Search Engine

4.2.1.1 Feature Extraction

The features we extracted from the metadata can be categorized into four groups, including location, time, activities, and visual concepts. We considered biometric data, for example, heart rate or calories, to be an inessential factor since they were not used in previous LSC series. Regarding visual concepts, we used the available concepts given by the organizer which consist of place attributes (for example "indoor" or "outdoor"), place categories ("home theater" or "television room"), and visible objects ("desk", "wall", or "television") in images. Furthermore, we enriched this type of feature by employing an additional object detector. We applied a semantic segmentation model called DeepLabv3+ [152], which was retrained on the ADE20k dataset [153], to not only detect visible objects but also their segmentation maps. We redefined the labels based on the similarities between some objects to remove some redundant names, for instance, "lamp" and "light source".

We could infer areas of detected objects in an image through its segmentation map by counting the number of pixels which were classified as corresponding objects. These area statistics were used to calculate a modified version of the term frequency-inverse document frequency statistic (TF-IDF [38]), which reflected the relative importance of each annotated object and is described in Section 4.2.1.4. This annotation, along with time, location, and activities, was stored in our database through an indexing mechanism which is described in Section 4.2.1.2.

4.2.1.2 Data Indexing and Enhancement

For MyScéal and its other versions, we indexed the annotation using Elasticsearch software, which is an open-source search and analytics engine. Although it is a full-text search library, it can be applied for retrieving other types of data such as images [154]. Regarding times data, we indexed it in a date format in Elasticsearch. Elasticsearch can support any time-related queries for data in date format. The GPS of location annotation in the metadata was stored in the geospatial format which contained latitude and longitude information. This GPS could be retrieved by using the map area of our interface when users select an area on the map. The times and the GPS data were expanded to enrich the database. We created the weekday texts (Monday, Tuesday, ...) from the times data and this weekday could be searched in an ordinary text format supported by Elasticsearch. This configuration helped searchers can perform retrieval without a need for faceted filters, which could be unwieldy in some user interfaces if do not have a good design. The location data was extended with the name of areas where the GPS points were located and were indexed in the text format in the Elasticsearch engine. They could be the names of cities or countries that the latitudes and longitudes belonged to. We used Google Maps API's Reverse Geocoding [1] to obtain the corresponding country and city from the latitudes and longitudes of a GPS point. This enrichment facilitated location-related queries as a normal free-text retrieval, especially when users did not know

[1] `https://developers.google.com/maps/documentation/geocoding/requests-reverse-geocoding`

where a queried area was located on the map. On the other hand, the detected objects created in Section 4.2.1.1 were indexed in original text format. We did not use other data such as music or biometric data because this information did not contribute much to the previous LSC events.

4.2.1.3 Query Expansion

Because MyScéal was designed to be a non-faceted retrieval system, users could query by using only text or by interacting with the map. Thus, an obstacle that we needed to address was to match words from a user's input with corresponding fields in the Elasticsearch database. A simple rule-based method using the part-of-speech tagging from the natural language toolkit library [155] was created to overcome this issue. For instance, from the query "*having coffee at the helix at 9 am on 23 August*", the system with our ad hoc regular expression pattern recognizes that there is an object (*coffee*), location (*the helix*), time (*9 am*), and date (*23 August*) and circulates them to the database for retrieving mechanism. However, the extracted keywords from the query are not always the same as those we indexed in the database. After we extracted the necessary fields of information in the part-of-speech tagging, the terms underwent another step in which they were mapped into a specific set of keywords that resulted from the feature extraction process. These terms then went to the mapping stage in which they were mapped into a pre-defined list of keywords obtained from Section 4.2.1.1. In this stage, we calculated the similarities between the words input by users and the keywords by a Word2Vec [73] model and WordNet [75] which is an extensive lexical database of English. WordNet provides relationships among words such as synonyms (the user may enter "*TV*" and this is mapped into the existing keyword "*television*"), hypernyms (a "*child*" is a "*person*"), and hyponyms (a "*fruit*" may be an "*apple*"). The Word2Vec model is used to learn the co-occurrence of words in one image. It was trained on the COCO detection and captioning dataset [43] as well as the LSC'20 dataset. This model was used to expand the entered word into semantically related concepts based on what it had learned

from the training dataset. For example, the word "*office*" suggests "*computer*" and "*chair*".

4.2.1.4 Information Retrieval

After matching all inputs from users to keywords indexed in the database, we could perform retrieval by measuring the similarity scores of surrogates, and these keywords expanded from the query. There are two distinct scores that affect the ranking of images. The first function is based on the filtering mechanism. From the query, MyScéal automatically extracts keywords that belong to filterable fields based on the tagging method described in Section 4.2.1.3. The list of all possible filters and their examples was defined as follows:

- *location:* an exact name of the location (Dublin City University) or the region (Oslo, Norway).
- *activity:* driving, walking, airplane, and transport.
- *time range:* before 6 A.M., around 12:30 P.M., between 4 P.M. and 6 P.M.
- *time of day:* morning, lunch, late afternoon, etc.
- *date or day of week:* a specific date (May 15), Monday, etc.
- *GPS coordinates*: using the map filtering which will be described in Section 4.2.2.

The expanded set of visual concepts resulting from the query expansion stage was also used for filtering. We used the ElasticSearch *term sets* query[2] in which an image will have a higher score if its annotation match more keywords expanded from the query. We denoted this score S_E.

Nevertheless, these visual concepts were used in not only the filtering process but also in the other scoring function. This is because the visual concepts were the

[2]`https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-terms-set-query.html`

result of the object detector, which was not sufficiently correct. The expansion stage also suffered the same issue. Therefore, using only the filtering function can lead to errors due to the accuracy of the detection model and the expansion models. In this case, we introduced a scoring algorithm, which was based on the ubiquitous TF-IDF [38] term weighting approach, to measure the importance of a visual concept in an image. We called this function **area TF-IDF** (**aTF-IDF**). The idea behind aTF-IDF is the assumption that bigger concepts (objects) contain more informative factors and hence are more important than others in an image. A bigger concept meant it had a larger area compared to other concepts in the image. The area of a concept was the number of pixels in its segmentation obtained from the DeepLabv3 [152] model. We denoted the set of lifelog images in the dataset by I, and the collection of visual concepts from Section 4.2.1.1 by C. Then the area of a concept $c \in C$ in an image $i \in I$ was expressed as $d_{o,i}$. The area of the concept was obtained directly from the segmentation model or from its bounding box output from the object detection model. The aTF-IDF of the concept c in the image i was calculated as following functions:

$$\text{aTF}(c, i) = 1 + log(d_{c,i}) \tag{4.1}$$

$$\text{aIDF}(c) = log\left(\frac{N}{|\{i \in I : d_{c,i} > \epsilon)\}|}\right) \tag{4.2}$$

$$\text{aTF-IDF}(c, i) = \text{aTF}(c, i) * \text{aIDF}(c) \tag{4.3}$$

where N was the total number of lifelog images in the dataset, and ϵ was a constant threshold of the area that verifies if a detected concept was a noise. The object detection model usually makes incorrect predictions for small objects. Thus, we added the ϵ threshold to eliminate the uncertain predictions. Finally, the similarity score between an image i and an expanded input query $q = [c_0, c_1, ..., c_m], c_j \in C$ was calculated as:

$$S(q, i) = w_1 S_E + w_2 \sum_{j=0}^{m} \text{aTF-IDF}(c_j, i) \tag{4.4}$$

where w_1, and w_2 were the hyperparameters that were used for balancing the weights and required tuning. This scoring function was the weighted average of both Elasticsearch function and our proposed aTF-IDF. We manually tried different values for w_1 and w_2 by solving several LSC queries to obtain the most appropriate values.

4.2.1.5 Temporal Retrieval

A special factor of lifelog data that distinguish it from other data is naturally sequential and continuous in nature. A lifelogger usually has a tendency to recall a moment with temporal relations. In fact, it also reflects in some LSC queries when they contain temporal information, such as "*I bought the bottle of wine after driving to the shop before I went to the party*". We needed to organize data in logically related sequences of images and metadata in order to make MyScéal support this type of temporal retrieval. We grouped redundant consecutive lifelog images that were visually similar into *scenes*. A difference in two adjacent scenes can indicate an introduction of a new action happening in a sequence of images or a change in the point of view of the lifelogger wearing the camera. For example, this difference may occur when the lifelogger transitions from sitting to standing and moving around, or when they begin using a blood pressure monitor after having breakfast. Images were categorized into scenes by leveraging their visual features, which were extracted using both the VGG model [70] and SIFT features [156]. The similarity between images was determined using the cosine distance on these visual features to ascertain whether they belonged to the same scene. Pertinent details about a scene, including its starting and ending time, were then appended to the image's annotation. With the introduction of the scenes, MyScéal can facilitate temporal retrieval which is a query containing up to three consecutive scenes. Regarding the query about shopping in the previous paragraph, the system firstly searches for the main scene, which is about buying a bottle of wine. The time information from the initial retrieved results is used as additional input for the two remaining scenes,

including driving to the shop and attending a party. In the final step, MyScéal groups all corresponding results and ranks them based on their total score. It is worth mentioning that MyScéal is one of the first systems that support the temporal retrieval mechanism.

4.2.2 User Interface

The LSC challenge evaluates participating interactive retrieval systems based on not only the accuracy of the retrieved results but also on how fast the systems are in finding the answers. Moreover, LSC also benchmarks systems on their performance with novice users who are first-time users of the systems. Therefore, a system need to have an appropriate interface to assist users, especially novice users, to find the correct images as quickly as possible. Figure 4.3 depicts the main UI of MyScéal participated in LSC'20. Because we aimed to develop MyScéal as a non-faceted lifelog retrieval system to facilitate a searching experience for novice users, we removed all filter selection panels and only kept the free-text search bars at the top of the interface. This design not only shows all functionalities of MyScéal to users but also helps users to minimize different steps in the search process.

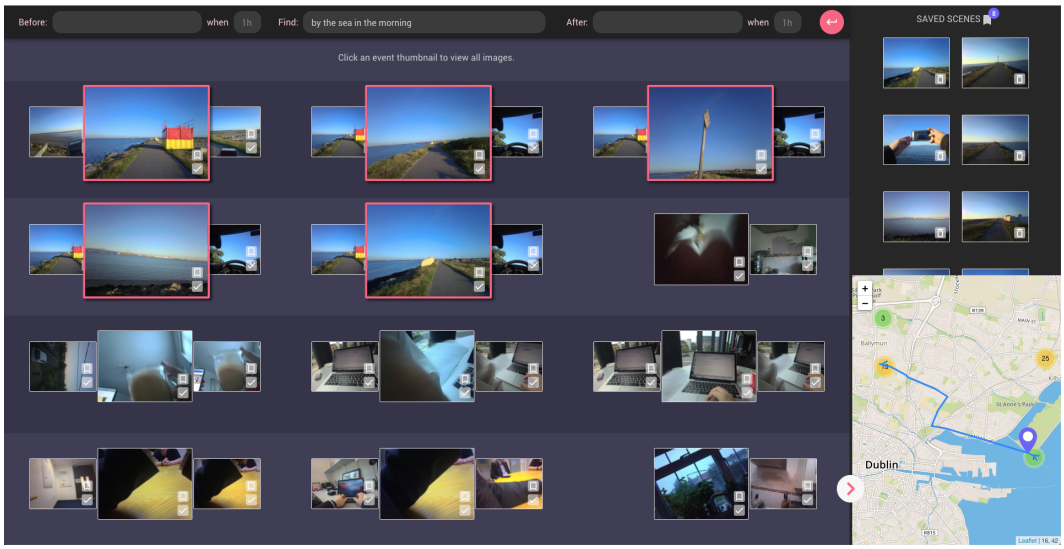


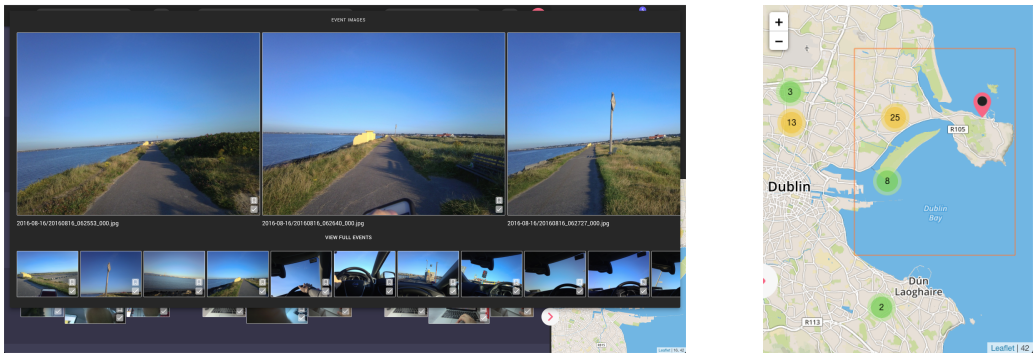
Figure 4.3: User Interface of MyScéal at LSC'20

After receiving a query, the user can start solving it by typing the keywords,

which can be identical to those in the query, in the search bars at the top of the screen. MyScéal employs part-of-speech tagging to automatically parse the input, organizing it into relevant fields in Elasticsearch to execute the retrieval. Part-of-speech tagging is a linguistic analysis technique that assigns grammatical categories, such as nouns, verbs, or adjectives, to each word in the user's input. However, the rule-based tagging stage only can capture information in one single scene. Therefore, our interface has three search bars that are for the scene that happened before, the main scene, and the scene that happened after to address temporal queries containing more than one scene. There are small boxes next to the search bars where the user can specify the relative time information of the two auxiliary scenes. The search bars area has a simple and straightforward design so that the user can quickly know how to use it and which bar should be used after the query is given. Furthermore, MyScéal is the first system whose UI contains three explicit search bars for temporal queries.

After the user enters words in either of the three search bars, MyScéal returns the relevant images and displays them in a vertically-scrollable panel which is in the main center area of the interface. The results are returned in the ranking order in which the most relevant result is at the top-left area, descending from the left to right, and from the top to the bottom. Each entry of results is shown in the way how the user enters search bars. The main scene will be represented by the relevant image in the middle, and the scenes before and after are shown by two smaller corresponding images on the left and right, accordingly. Although each scene is represented by a photo, there can be multiple images in the scene. The user can click on the image to open a pop-up panel that shows all images belonging to that scene at the top row, as shown in Figure 4.4a. The row below is for displaying all other scenes on the same day and in the temporal order from left to right. This pop-up window will remain until the user clicks on other parts of the screen. The UI also allows the user to perform location filtering by using the map area as depicted in Figure 4.4b. The user can draw a rectangle on the map to perform seeking within

the area of interest. The restricted area will be highlighted in red. Furthermore, this geographic map located below the screen also illustrates the location of where an image was captured whenever the user clicks on a specific image. This ability is useful for certain queries in which visualization of the locations plays a critical role, for example walking in a park near a river.



(a) Pop-up panel to explore all images in a scene

(b) Map filtering

Figure 4.4: Supplementary elements in MyScéal's interface

As discussed in Section 3.3.1.1, LSC has a penalty scoring scheme for an incorrect submission. Therefore, it is better to submit only when the user believes the currently selected image is correct although it might take time. Our UI introduces the "*Saved scenes*" panel located on the top-right of the screen which is shown in Figure 4.3. The user can temporarily save any images which are relevant to a given query. The candidate images remain in this panel while the user keeps searching for other relevant images. If the user cannot find any other appropriate images, the user can look back at the panel to decide whether to submit one of them or not. There are two small icons attached at the bottom-right of each image. The first icon is for saving it to the "Saved scenes" area, and the remaining icon is for submitting the image.

4.2.3 LSC'20 Configuration

4.2.3.1 Dataset

The lifelog data used in LSC'20 contained more than 191,000 images captured by one lifelogger over four months between the years 2015 to 2019. The dataset comprised parts of the three previous NTCIR-Lifelog challenges [32, 33, 34]. The images were recorded using the OMG Autographer and Narrative Clip wearable cameras with a resolution of 1024×768. All images were anonymized by removing potentially sensitive context or readable text and blurring all faces appearing in the images. In addition to the visual data, the dataset came with other information including textual metadata, biometrics, and GPS coordinates of where the images were captured. The textual metadata consisted of timestamps and physical activities of the lifelogger. Moreover, the organizer also provided a list of visual concepts such as objects, their bounding boxes, and scenes detected in the original images.

4.2.3.2 Query Description

LSC requires participating systems to find within the dataset one image that is relevant to a given query. A query may contain more than one correct image but teams only need to submit one of them. Because LSC evaluates interactive systems in a real-time manner, each query in this challenge is not entirely shown to participants at once but is gradually revealed. A query usually starts with a basic descriptive clue about a specific moment. The query is continuously extended with an additional hint containing more valuable contextual information after each iteration of 30 seconds. These hints make the query become more precise and easier to solve.

Figure 4.5 shows an example of an LSC query and its relevant images. The clues appearing in later iterations usually contain more explicit details, for instance, locations or times. This querying mechanism imitates the ordinary behavior of humans when they recall a memory in an incremental way. Each query has six



I was building a computer alone in the early morning on a Friday | at a desk | with a blue background. | Sometimes I needed to refer to the manual. | I remember some Chinese posters on the desk background. | I was in Dublin City University in 2015.

Figure 4.5: A semantic query and relevant lifelog images. The symbol " | " indicates hints in the query.

clues in total, including the first clue shown to teams at the beginning. Thus, it takes 150 seconds to fully reveal the complete query to participants. After that, expert users have an extra 30 seconds to find the answer (180 seconds in total), while novice users have another 150 seconds (300 seconds in total) due to the gap in knowledge between expert and novice users. Because of the COVID endemic, LSC'20 had to be online which was not suitable for organizing the session for novice users. Therefore, LSC'20 only had one session for expert users and the organizer decided that the maximum search time of expert users now is the same as novice users which is 300 seconds.

4.2.4 Performance

4.2.4.1 Other Systems

There was a total of 14 retrieval systems (including MyScéal) that participated in LSC'20. We will briefly introduce some noticeable systems. Most of them were the upgraded versions of the systems which participated in LSC'19. LifeXplore [157] and VIRET [158] are two systems that have attended all LSC iterations. In the LSC'20, VIRET employed the W2VV++ model [159] to encode and project a semantic query into a vector in the common space with visual vectors of images obtained from CNN-based models. This system also integrated the self-organizing map [102] that has been applied in several systems in previous LSC series. Meanwhile, LifeXplore in LSC'20 had minor improvements compared to its preceding version focusing on concepts enrichment. It applied the YOLO9000

object detection model [160] and Optical Character Recognition (OCR) detection model[3] to enlarge the number of concepts for each image. Similarly, the vitrivr system [161] used the Faster-RCNN model pre-trained on the OpenImages V4 dataset [4] to explore more concepts. Moreover, this system also included a temporal querying mechanism to address queries containing more than one scene. Although attended LSC'19 as a purely relevance feedback-based system, Exquisitor [162] added the free-text search engine in LSC'20 to support its main feature. VIRLE [163] was the updated version of its original system which won the first LSC [9]. VIRLE used the same search engine through the APIs provided by our MyScéal but was built in a virtual reality interface.

SOMHunter [103] was among four systems participating for the first time in LSC'20. This system obtained first place in VBS2020 and was modified to be appropriate for lifelog data. It supported the option for retrieving based on the user's relevance feedback like Exquisitor. Nevertheless, the system integrated the SOM to have a comprehensible display technique for many images. LifeGraph [143] and NTUnlp [146] were the other two systems that marked LSC'20 as their first appearance. As mentioned in Section 2.3, they both used graph-based structures in their retrieving process. Voxento [164] was the first lifelog retrieval system supporting searching through voice commands. The system utilized Google Web Speech API[5] to recognize the voice and convert it into the text to perform the retrieval which was based on a standard free-text search engine MongoDB. This system was the first proposal to build a truly hand-free interactive lifelog retrieval system.

4.2.4.2 Results

The detailed comparative performance of each system is illustrated in Figure 4.6 which depicts the precision and recall of some systems in the LSC'20 competition. It

[3] `https://github.com/tesseract-ocr`

[4] `https://storage.googleapis.com/openimages/web/index.html`

[5] `https://developer.chrome.com/blog/voice-driven-web-apps-introduction-to-the-web-speech-api/`

is noted that the order of the systems in the figure reflects their relative ranking to others in which MyScéal achieved first place in the challenge. As defined in Section 3.3.1.1, the precision is calculated by the number of correct submissions divided by the total submissions, whilst the recall is the portion of queries that the system answered correctly. Our system MyScéal obtained the highest score for both precision and recall metrics which helped us to achieve first place overall. The system managed to successfully solve 21/24 queries (recall was 87.5%) after 25 submissions of which four of them were incorrect (precision was 84%). The SOMHunter also solved the same number of queries as MyScéal, but it submitted more incorrect answers (9/30 submissions were wrong) leading to lower precision and getting second place. Also using the relevance feedback approach, Exquisitor had a quite similar precision score to SOMHunter but got fifth place in the leaderboard. This was because the latter system outperformed the former with a higher recall metric. Three systems that allowed users to search by drawing were vitrivr, lifeXplore, and VIRET. They all had comparable scores in terms of recall and precision metrics. Although the VR interface system used to claim first place in LSC'18, VIRLE finished LSC'20 in the middle of the leaderboard. Regarding graph-based retrieval systems, LifeGraph achieved the second-highest precision score with 80%, although its recall was relatively low. The same was true with NTUnlp when its recall was significantly lower than its precision. Meanwhile, Voxento found the correct answers for three queries after 10 incorrect submissions.

LSC series score a system based on not only the accuracy but also their retrieval time. Figure 4.7 illustrates the number of times that a system finds the correct answer quickly enough to be one of the top three speediest systems that successfully solve a query. This metric can reflect how efficient a system is when seeking the answer in terms of speed. As we can see in the figure, MyScéal surpassed all other systems in terms of appearances in the top three systems which submitted the correct answers faster than others. It is worth pointing out that our system found the relevant images faster than other teams in half of the competition with 13/21 queries. Both VIRET

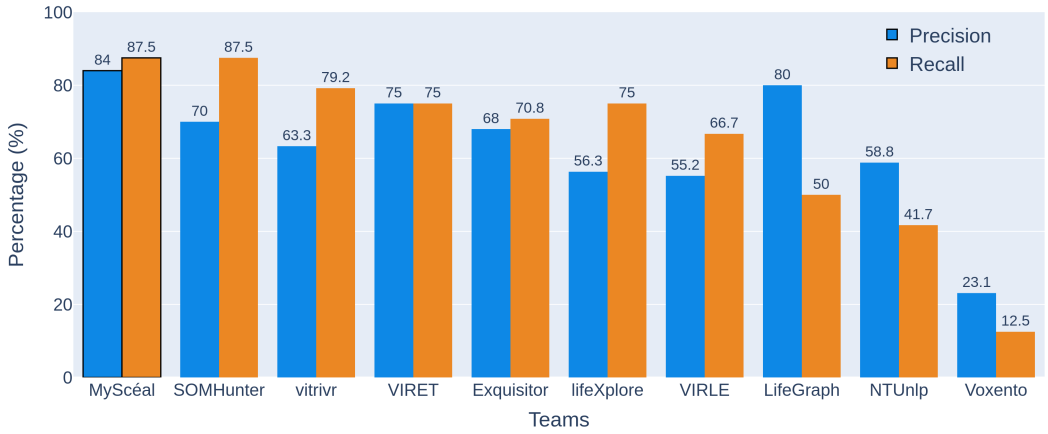


Figure 4.6: The precision and recall of some systems participated in LSC'20. The teams are ordered based on their official rank in the competition.

and SOMHunter used the hierarchical SOM visualization and acquired the second-highest metric of search time. While vitrivr, Exquisitor, and VIRLE had similar performance in terms of this metric with an average of five times, the remaining systems got half of that number.

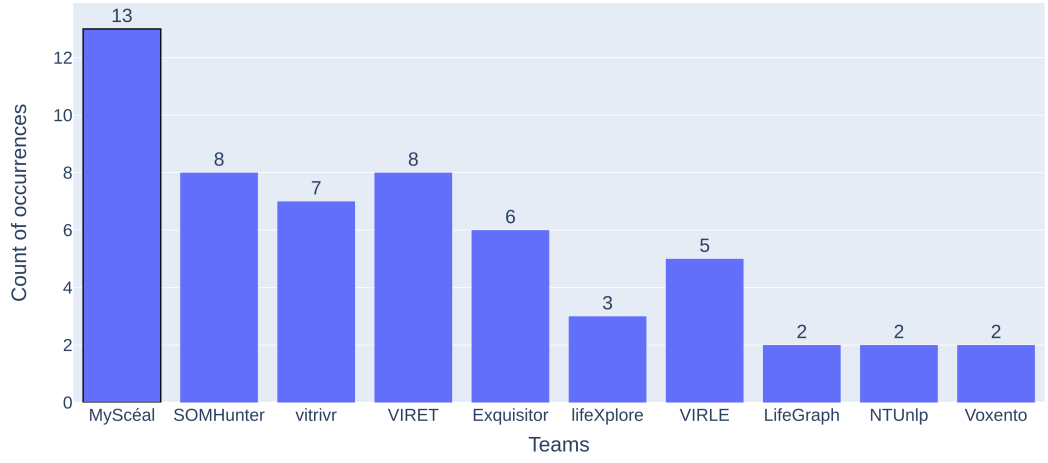


Figure 4.7: Number of times when a system was in the top three quickest systems submitting correct answers.

The result of LSC'20 proved that the first version of MyScéal can be considered the SOTA interactive lifelog retrieval system when the system not only obtained the highest accuracy (highest precision score) but also was the fastest system that found

the correct answer among other competitors.

4.2.5 Learning Improvement

Although MyScéal acquired the highest overall scores in LSC'20, we realized that it still had room for improvement. Because the nature of lifelogging is continuously capturing, it leads to a huge number of images in a day. Despite our introduction of scene grouping, we found it still took us a while to scroll through entire images taken in the day. Therefore, it would be more efficient to have a method that enables us to quickly scan through photos captured on the same day. Furthermore, the contribution of concepts in the retrieval process was hidden from users. Additionally, the retrieval process could benefit from displaying the score of each concept, allowing users to modify and adjust the results accordingly. These were the learning improvements for the modified version of MyScéal that participated in ImageCLEF2020-Lifelog.

4.3 MyScéal at ImageCLEF2020-Lifelog

While our initial aim in developing MyScéal was to find a single image relevant to a given query, we extended the system's capabilities to participate in the ImageCLEF2020-Lifelog challenge. This challenge required teams to submit all relevant images, presenting us with an opportunity to introduce new features to MyScéal and evaluate its performance in a different task, given that this challenge utilized the same lifelog dataset as LSC'20.

4.3.1 System Modification

We updated MyScéal in both the backend and UI aspects to address the requirement of ImageCLEF2020-Lifelog and to overcome the limitation mentioned in Section 4.2.5.

4.3.1.1 Visual Concept Enrichment

In this version of MyScéal, we still followed the pipeline of the original version which was to use the visual concepts in the metadata to annotate images. These concepts were extended with a list of detected objects obtained from the descriptor DeepLabv3+ [152]. Moreover, we tried applying the Microsoft Computer Vision API[6] to enrich the visual annotation further. This image analysis service from Microsoft can detect objects visible in an image as well as its contexts such as the scenery (*indoor*), or actions (*waiting*).

4.3.1.2 Visual Similarity Update

In contrast to LSC, where the goal was to find a single image relevant to a given query, ImageCLEF2020-Lifelog required teams to identify all relevant images. Therefore, we introduced a new visual similarity feature to address this requirement. This new functionality helps users to find all images that are visually similar to a selected image. We measured the similarity between images by using the cosine distance on their concatenated vector features which comprise SIFT and VGG features. We made this process offline to guarantee real-time retrieval. This meant we first extracted a list of similar images of each image, then we indexed the list into the ElasticSearch engine as the image's annotation. When the user requests to see similar images of an image, the engine will return the pre-computed results and show them to the user through the updated UI.

4.3.1.3 User Interface Update

Another thing that ImageCLEF2020-Lifelog was different from LSC'20 was the queries from the former challenge did not contain any *before* or *after* information but only one specific moment, for example "*Find all the moments the lifelogger is having a sushi in a restaurant*". Thus, we replaced three temporal search bars with only one conventional search bar as shown in Figure 4.8. After the user fills in the

[6] `https://azure.microsoft.com/en-us/products/cognitive-services/vision-services`

bar, the retrieved images are displayed in the central main panel of the UI in a ranked list. This result panel allows users to scroll in both vertical and horizontal ways. Each row is a group of scenes (known as clusters used by the organizer) within the same day that are sorted in descending order of similarity to the query. This change in the UI not only reduces the number of irrelevant images but also facilitates getting a higher cluster recall which is one of the evaluation metrics used in this challenge and is defined in Section 3.3.1.2. The reason is that the new UI will help the user to be easier to identify different scenes on different days, hence they can have more time to seek other relevant scenes and increase the recall metric.

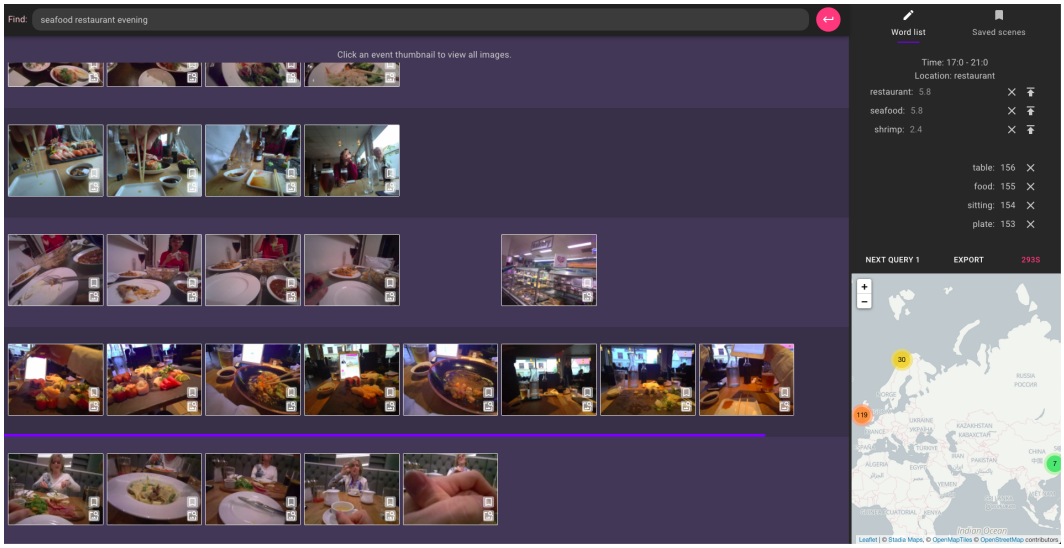


Figure 4.8: A modified UI of MyScéal for ImageCLEF2020-Lifelog

4.3.1.4 Word List Panel Update

The "*Word List*" panel is an additional feature that we created to make MyScéal easier for novice users to interact with. This feature shows all concepts, which are created after the query expansion stage, to the user. It can help the user to have more insight into how the internal search engine computes the similarity scores between the input query and images. Furthermore, this panel allows the user to adjust the score of each concept to modify or even remove their contribution to the retrieving process

and update the results. We decided to present only the 20 highest concepts to ensure the user will not be confused by a long list of words. Right below this scoring list is a brief summary of the retrieved results which shows the list of concepts appearing most frequently in the results along with their corresponding number of occurrences. The user can have an overview of the results by looking at this summary and then can decide to remove images containing irrelevant concepts. The "Word List" panel is located in the same area as the original "Saved scenes" panel in the top-right area of the UI as illustrated in Figure 4.8.

4.3.1.5 Event View Update

We introduced the ***event*** structure that supports the user to scroll images through a day at a faster pace. An event consists of one more multiple adjacent scenes whose boundaries are defined by a change in the context of scenes, such as a change in location or activities between two consecutive scenes, or by a significant time gap. For better clarification, a scene is a sequence of visually similar adjacent images while an event is a sequence of adjacent scenes which belong to the same context or the same location. Figure 4.9 depicts a revised pop-up window displaying scenes and events in a day. We kept the first and the second rows having the same functionality as Figure 4.4a, but the third row is for illustrating events in the day. Whenever a user clicks on an event in the third row, our UI will automatically navigate to the detailed images in the first row and the corresponding scenes in the second row. With this design, the user now can scroll the third row specialized for event views to quickly see all images in the day.

4.3.2 Performance

4.3.2.1 Other Systems

There were six teams that attended the ImageCLEF2020-Lifelog challenge, including the baseline system from the organizer called LIFER 2.0 [165]. This system was the improved version of LIFER [7] which supported faceted filtering

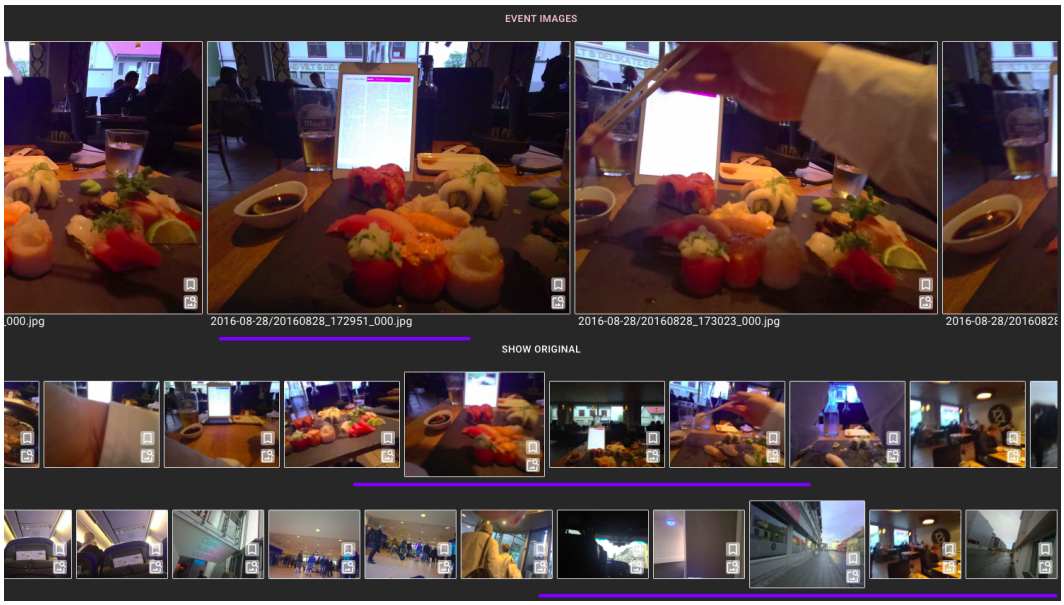


Figure 4.9: A modified pop-up window showing all scenes and events in a day

and full-text search retrieval. The similarity between images and a given query was basically measured based on the number of matching concepts. The REGIM-Lab system [166] enriched the visual concepts by employing YOLOv3 object detection model [167]. They trained an LSTM model to extract the concepts in the query that were already indexed in the Elasticsearch database. Then they used a similar similarity function from LIFER 2.0 to find the answer. Meanwhile the UAPT team [168] employed the SpaCy library '[169] to measure the cosine similarity between embedded concepts from the query and those from images' annotations. The system group retrieved images into distinct groups based on their timestamps so that users can easily submit different events to increase the evaluation metrics. On the other hand, BIDAL utilized the Fixmatch approach [170] to automatically split lifelog images into scenes, and used an attention-based model to find images relevant to the query. Also using the attention model, the HCMUS team encoded images into the same vector space with encoded queries obtained from the RoBERTa model [171] which was an evolution of the BERT model. The RoBERTa model introduced improvements such as adjusted training objectives, utilization of larger datasets, dynamic masking, and independence from sentence order

consideration. These enhancements in RoBERTa enhanced representation learning, making it a versatile and robust choice for various natural language processing tasks. The cosine similarity function then was used to compare them together to output the results. Furthermore, the system supported finding similar image functionality using the ResNet152 image encoder [120]. It is noted that there was only the REGIM system following the automatic retrieval configuration, whilst others addressed this challenge using their interactive retrieval systems.

4.3.2.2 Results

There were 10 queries in the ImageCLEF2020-Lifelog challenge [31]. The official evaluation metrics are Precision at 10 (P@10), Cluster Recall at 10 (CR@10), and the F1 Score at 10 (F1@10). These metrics are defined in Section 3.3.1.2 and are different from the precision and recall used in the LSC challenge. Because this task focuses on seeking images from all relevant clusters instead of only one cluster, it needs a post-processing step to refine the submissions to improve the scores. We rearranged the retrieved ranked list by choosing one image in each cluster to move to the top of the list to ensure a higher CR@10. The remaining items in the list were filled in with visually similar images of the aforementioned images in the cluster. This post-processing stage was conducted automatically offline after the user finished solving a query.

Figure 4.10 depicts the F1@10 of all participants. HCMUS got the highest score of 0.81, securing the first place, while the BIDAL team followed with a score of 0.69. MyScéal-Raw was the original submission of MyScéal without the post-processing stage, whereas MyScéal-Post was the result after we applied the modification steps to the results of MySceál-Raw. The initial submission of MyScéal received a score of 0.48. However, our post-processing step demonstrated its effectiveness by boosting the F1@10 score to 0.63, securing a third-place finish. Meanwhile, the only automatic system in the challenge REGIM had the lowest F1@10 score among all participants at 0.19. This result indicates that MyScéal

achieved a state-of-the-art performance in the interactive LSC challenge. Nevertheless, it demonstrated a mid-range performance in the new task, which demands the system to identify all potential answers.

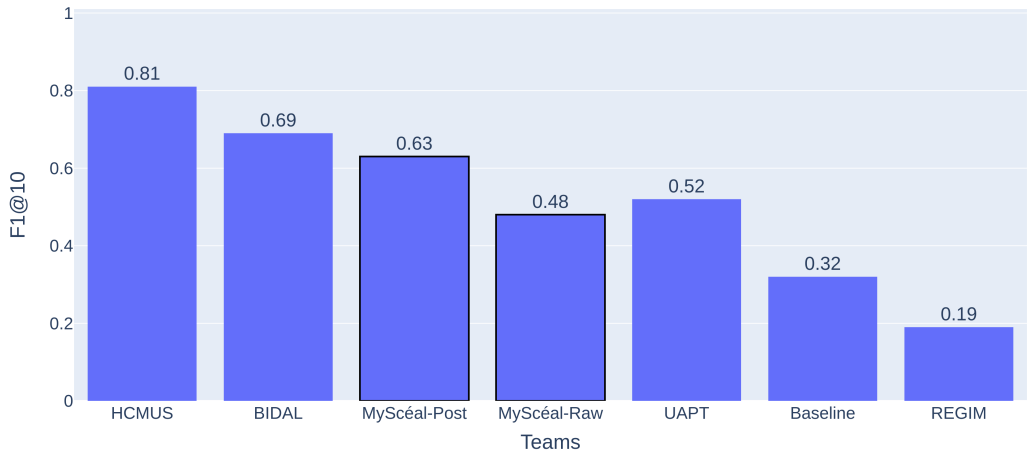


Figure 4.10: F1@10 scores of teams participating in ImageCLEF2020-Lifelog retrieval task

Because the ImageCLEF2020-Lifelog challenge was an offline benchmarking task, it allowed participants to submit multiple submissions and only recorded the best submission for the final ranking. We took this chance to conduct a user study in which three types of users address the ImageCLEF2020-Lifelog challenge using the modified version of MyScéal. The performance that was officially recorded by the ImageCLEF2020-Lifelog challenge and is discussed in the aforementioned paragraph was the result of our user study which is presented in the following section.

4.3.2.3 User Study

We attended this challenge since it was a chance not only to evaluate MyScéal on the new task but also to conduct a user study to gain insight into how other users operate the system. The user experiment consisted of three users solving 10 queries of the ImageCLEF2020-Lifelog challenge. Three users included us who were the system developer (expert user), the knowledgeable data owner (lifelogger), and a novice user. While the expert user knew in detail how to interact with MyScéal in

the best way, the lifelogger did not have any knowledge about the system but was expected to know the data including the correct answers to queries. Contrastingly, the novice user was not familiar with either the system or the data. We followed the official configuration of the challenge in which a user had a maximum of five minutes to solve each query. The time of reading the query was excluded from these five minutes. Lifelogger and the novice user were instructed by the expert user on how to use the system to solve some sample queries before starting the experiment. All users were encouraged to use the entire five minutes to find as many correct answers as possible to get a higher F1@10 score.

Table 4.1 offers insight into the performance of three users in our experiment. The *Raw* column in the table indicates the original result from users and the *Post* column shows the score after applying the post-processing step. In general, this step improved the scores of all users, especially because it could boost the F1@10 scores of several queries to the maximum value. The result shows that a user could obtain the performance of a lifelogger, who knew the data clearly, by using the MyScéal system in the most appropriate way such as the expert user. The expert user even got higher scores than the lifelogger in some queries. Although the novice user only found the answer for six out of 10 queries[7], they achieved the comparative scores to the other two users in these queries. In particular, the novice user got better scores than the lifelogger in query 1 and query 6. Compared to the expert user, the novice user performed better in query 3 and query 9. Meanwhile, they obtained the same score as the other two users in query 7 and query 10.

4.3.3 Learning Improvement

Following the user study, we discovered that the "Word List" panel did not prove as useful as anticipated, as users seldom interacted with this feature during searches. The panel was added based on the assumption that users would look at the concepts listed in the panel to modify their inputs. Nevertheless, the users (including expert

[7]The queries can be found at `https://www.imageclef.org/system/files/ImageCLEF2020-test-topics.pdf`

Table 4.1: $F1@10$ scores of three users. The *dot* indicates the user could not find the answer. Numbers in **bold** are the highest number in that query among all users.

Query	**Expert User**		**Lifelogger**		**Novice User**	
	Raw	Post	Raw	Post	Raw	Post
1	0.33	**1**	0.5	0.58	0.28	0.67
2	0.22	0.22	0.22	**0.72**	.	.
3	0.33	0.57	0.94	**1**	0.33	**1**
4	0.22	0.22	0.27	**0.31**	.	.
5	0.68	**0.68**	0.68	**0.68**	.	.
6	0.5	**0.5**	0.25	0.25	0.25	0.25
7	0.68	**0.89**	0.44	0.69	0.4	0.69
8	0.33	**1**	0.33	0.75	.	.
9	0.31	0.73	0.68	**0.8**	0.31	0.77
10	0.5	**0.5**	0.5	**0.5**	0.5	**0.5**
Overall	0.41	**0.63**	0.48	**0.63**	0.21	0.39

users) did not use this feature in most queries. One reason for this was that users mainly focused on looking through images to find the answers rather than thinking about the panel function. In contrast, we decided to keep other features, including event view and visual similarity, in the next version of MyScéal called MyScéal 2.0 participating in LSC'21. It was because we observed that the event view in the UI helped users to go through images in a day with a faster pace and the visual similarity functionality facilitated searching potential correct images sharing the same visual context with the selected image.

4.4 MyScéal 2.0 at LSC'21

We made some improvements in the original system at LSC'20 to create the next version of it, called MyScéal 2.0. MyScéal 2.0 is the latest version of our concept-based interactive lifelog retrieval system for this research. The updated system was developed to participate in LSC'21. The upgrades included those we introduced in ImageCLEF2020-Lifelog. Additionally, we also wanted to take into account feedback from novice users to improve the system. Nevertheless, LSC'20 did not organize the novice sessions due to the pandemic, whilst the user study in

the ImageCLEF2020-Lifelog challenge contained only one novice user, and this challenge evaluated the system on criteria that were different from LSC. Therefore, we conducted an experiment with more novice users to evaluate their performance in using the original version of MyScéal to solve several LSC'20 queries. The result and feedback from them revealed some drawbacks of the system which we based on to make some improvements.

4.4.1 Novice User Experiment for MyScéal at LSC'20

We designed the novice user experiment with the same configuration as the LSC series, in which we used the official scoring function of the LSC to evaluate the performance. There were eight novice volunteers in the study and each of them was required to solve five queries of LSC'20. The novice users were our friends who came from different research backgrounds and education levels ranging from undergraduate to master. All of them were not familiar with the lifelog retrieval task beforehand. These queries were selected based on their difficulties which were assessed by our experience in LSC'20. The queries were shown to users in a similar process to the LSC format, which is to be gradually revealed. The detail of queries is listed in Appendix A.1. Users were trained to use MyScéal before doing the experiment and needed to solve each query within five minutes.

Table 4.2 shows the scores of volunteers in our novice users experiment compared to the official scores of MyScéal in LSC'20 which was operated by the expert users. Five queries were ranked in ascending order of difficulty in which the first query was easy to solve and the fifth query was the hardest since we could not find the answer in LSC'20. While we who are the expert user successfully solved four queries, most of the novice users found the correct images for three queries. This led to the result that the average score of novice users was approximately half of the expert users. No user, including the expert user, got points in the last query.

The result showed that there remained a huge gap in the performance of novice and expert users. There were several reasons behind it. Firstly, novice users usually

Table 4.2: Experiment score of eight novice users compared to MyScéal team score in LSC'20

Query	**MyScéal**	**U1**	**U2**	**U3**	**U4**	**U5**	**U6**	**U7**	**U8**
1	94.86	92.5	94.86	85	86.81	98.19	87.92	90.69	82.36
2	78.06	54.86	50.69	68.61	47.08	59.03	57.92	43.33	59.58
3	87.5	53.47	0	0	53.33	0	0	0	0
4	78.61	0	77.5	51.11	0	52.5	0	64.44	0
5	0	0	0	0	0	0	0	0	0
Total	339.03	200.83	223.06	204.72	187.22	209.72	145.83	198.47	141.94

underestimated the usefulness of the map panel although using it could help searchers to find the answers easier in the first four queries. They commented that the position of the map panel in the interface (bottom right corner as shown in Figure 4.3) did not effectively grasp the attention of users. Secondly, although being guided on how to use the map, novice users still had some challenges in locating important locations mentioned in queries, such as *lifelogger's home*. Last, the major drawback of the first version of MyScéal was that the system did not employ OCR and color detection models. For instance, the second and the fifth queries would be solved quickly if MyScéal had these annotations in the database.

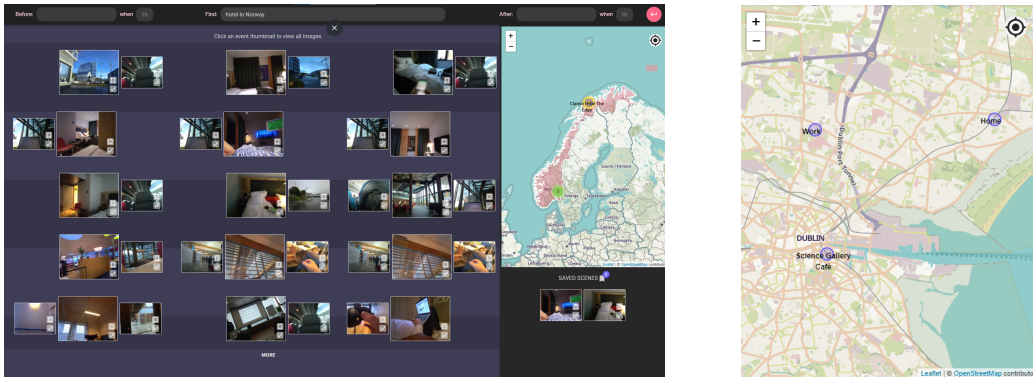
4.4.2 System Modification

We developed MyScéal 2.0 by extending the original MyScéal with features introduced in the ImageCLEF2020-Lifelog challenge. These updates are discussed in Section 4.3.1, excluding the UI and the "Word List" panel. Moreover, we made the following improvements based on the feedback of novice users after the experiment.

4.4.2.1 Map Panel Update

Because the old position of the map was not attractive enough for novice users, we decided to increase the size of the map and moved it to the top-right corner next to the search bars, as illustrated in Figure 4.11a, to encourage users to interact with the map feature more often. Moreover, in Figure 4.11b, we annotated certain locations

on the map, like "home" and "work", aiming to simplify the process of addressing queries related to location information and, consequently, minimizing retrieval time.



(a) The main UI of MyScéal 2.0 at LSC'21 (b) Annotation on the map

Figure 4.11: Updates on UI and the Map panel in MyScéal 2.0

4.4.2.2 Additional Annotation

Since OCR and color information played a critical role in solving LSC'20 queries, we extracted these concepts by employing the Microsoft Computer Vision API[8] service for the OCR, and the pre-trained bottom-up attention model [172] for recognizing the color attributes in the images. The updated pipelines of MyScéal 2.0 is depicted in Figure 4.12, which is similar to its first version but includes additional features illustrated in blue boxes.

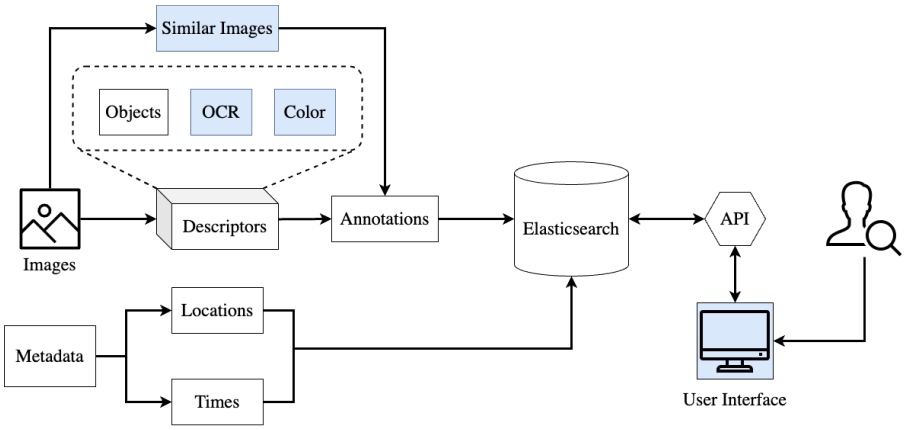


Figure 4.12: Pipeline of MyScéal 2.0

[8] https://portal.vision.cognitive.azure.com/gallery/ocr

4.4.3 LSC'21 Configuration

The configuration of LSC'21 remained the same as LSC'20. The same criteria were used to evaluate the performance of participating teams. In addition, the novice session still could not be organized due to the pandemic. Regarding the dataset, LSC'21 used the same dataset that had been used in LSC'20. Nevertheless, more than 8,000 images were removed due to data governance reasons. Thus, the LSC'21 dataset contained roughly 183,000 anonymized images with a resolution of 1024×768 captured by one lifelogger in four months spreading from 2015 to 2019. Similar to LSC'20, the dataset used in LSC'21 came with metadata including textual information and biometrics. A list of extracted concepts and bounding boxes of detected objects from each image was also given by the organizer.

4.4.4 Performance

There were 17 systems participating in LSC'21. We mention here several teams, including systems obtaining top-performance results, systems having unique features, and those using graph-based structures.

4.4.4.1 Other Systems

LifeSeeker 3.0 [173] was the third iteration of the LifeSeeker that has participated in the LSC since 2019. It used the same enriched concepts as MyScéal 2.0 but had the modification in the similarity function in the retrieving process. Instead of using the ordinary TF-IDF, LifeSeeker 3.0 employed the weighted term frequency along with the built-in function of Elasticsearch to find images relevant to a query. XQC system [174] was the first attempt to develop an interactive lifelog retrieval system on a mobile phone platform. This Android-based system followed the same approach as the Exquisitor system [162], which was to retrieve by using relevance feedback from searchers in addition to other filtering mechanisms. Regarding using graph-based structures, LifeConcept system [175] was the enhanced version of the

NTUnlp system [146] in LSC'20 which applied scene graph detection model to extract relations information between objects to enrich the concepts. In this LSC'21, this concept-based retrieval system added a new query expansion stage by using ConceptNet which was similar to MyScéal's approach. LifeGraph system [176] attended LSC'20 and made minor improvements to participate in LSC'21. The system used the knowledge graph from the external data[9] to obtain more detailed concepts that were not usually recognized by detection models. It then adopted an embedding technique for graph-based structures [177] to perform retrieval. LSC'21 witnessed an increase in the number of systems using embedding-based search engines. Most of them employed CLIP which was a recently released multimodal embedding model. SOMHunter+ [104] attended LSC'21 with the additional CLIP integrated besides the W2VV++ embedding model which was already used in LSC'20. Both Memento [178] and Voxento 2.0 [179] shared the same backend search engine. While the former system was first introduced in LSC'21, the latter system was the voice-controlled system updated from LSC'20.

4.4.4.2 Results

Table 4.3 illustrates the performance of some teams that participated in LSC'21. The *Overall Scores* column is the normalized final score scaled based on the highest score in the competition. There were 23 queries used in LSC'21 and LifeSeeker 3.0 was the system that solves the most queries among all participants with 20 queries. This made the team obtain the best recall scores at 86.95% and get third place in the final ranking table. On the other hand, Voxento 2.0 made a huge improvement in its performance compared to its previous version when this system achieved the highest precision at 85.71% with only three wrong submissions. Memento had the same backend search engine as Voxento 2.0, however, this system got more incorrect answers leading to a lower score. Although not achieving either the best precision

[9] `https://www.wikidata.org/`

and recall scores, MyScéal 2.0 and SOMHunter+ were the two best-performance systems in LSC'21 with competitive scores with LifeSeeker 3.0. The main reason was because of their fast retrieval speed when they both managed to achieve the most number of times of submitting the correct answers within the top-3 speediest systems. Regarding using graphs, while LifeConcept successfully found the correct answers for half of the queries, LifeGraph solved three of them. The Android-based XQC system solved a fifth of the queries and finished at the bottom half of the ranking.

The result showed the advantages of applying CLIP, which was the recently released SOTA image-text retrieval model at that time, in lifelog retrieval since most of the embedding-based systems obtained relatively high scores and competitive final rankings. This embedding model can not only capture the fine-grained and general information of images but also can recognize OCR within them. It is worth mentioning that OCR played a critical role in LSC'21 since 11/23 queries containing this type of information. Systems that supported the OCR feature could easily find the correct images at a fast pace and obtained high scores. Although MyScéal 2.0 and LifeSeeker 3.0 did not employ CLIP, both systems used the image analysis service from Microsoft to detect the OCR features and indexed them in the database for the retrieval stage. The detail of the comparison between systems as well as the queries in LSC'21 can be found in our research paper [180].

Table 4.3: Performance of several teams in LSC'21

	Solved Tasks	Wrong Submissions	Precisions (%)	Recalls (%)	Submitted in Top-3	Overall Scores
MyScéal 2.0	19	4	82.61	82.61	**12**	**100**
SOMHunter+	19	9	67.85	82.61	**12**	97.6
LifeSeeker 3.0	**20**	6	76.92	**86.95**	9	97
Voxento 2.0	18	**3**	**85.71**	78.26	11	91.4
Memento	16	11	59.25	69.56	9	77.2
LifeConcept	12	17	41.37	52.17	1	48.3
XQC	5	9	21.73	35.71	1	21.5
LifeGraph	3	5	37.5	13.04	0	11.5

4.5 Discussion and Conclusion

The performance of different versions of MyScéal in the LSC series and ImageCLEF2020-Lifelog challenges showed that our concept-based interactive lifelog retrieval system can be considered a SOTA in the lifelog retrieval task. Specifically, MyScéal and MyScéal 2.0 obtained first place in LSC'20 and LSC'21, accordingly. These concept-based systems achieved high accuracy and ensured a fast retrieval time compared to other participating systems in the LSC competition which is a popular international real-time benchmarking evaluation workshop. However, MyScéal had a mid-range performance in the ImageCLEF2020-Lifelog challenge. This was because this challenge required a system to find all relevant images rather than only one image of the LSC challenge. We developed MyScéal to match the requirement of LSC, thus our system did not perform well in the ImageCLEF2020-Lifelog task. Although some modifications in MyScéal were introduced to address the latter challenge, we gained third place in the final ranking. Nevertheless, ImageCLEF2020-Lifelog was a chance for us to evaluate our concept-based system on a new task. Additionally, we also conducted some user studies from which several learning improvements were discovered to ensure its SOTA performance.

4.6 Chapter Summary

This chapter addresses the first research question which is "*What is the state-of-the-art of the interactive lifelog retrieval systems following the conventional concept-based approach?*". The answer to this question is proposed by collaboratively developing the MyScéal system (and its updated version) and evaluating its performance through several well-known lifelog retrieval competitions with my colleague. The main compartment of MyScéal is to annotate images with their corresponding concepts which can be visual objects, scenery, OCR, and other

metadata such as time and locations. These concepts are then matched to those of a given semantic query by our aTF-IDF scoring function to retrieve relevant images. The system participates in three competitions, including LSC'20, ImageCLEF2020-Lifelog, and LSC'21, and obtains the results among top-3 in all of them. The performance of MyScéal iterations in these challenges proves that this system can be considered to be a SOTA concept-based lifelog retrieval system. Contrarily, the performances of graph-based lifelog retrieval systems indicated that their search engines are unsuitable as a baseline for this research. Recognizing that lifelog retrieval can be viewed as a sub-task of image-text retrieval, I chose to delve into the image-text retrieval task. This decision was driven by the understanding that exploring the intricacies of image-text retrieval provides a foundational framework for developing a novel graph-based retrieval model. The next chapter will discuss my solutions for the second research question which is about applying graph-based structures to improve the accuracy of current state-of-the-art retrieval models in the image-text retrieval task.

Chapter 5

Graph-based Models in Image-Text Retrieval

5.1 Introduction

After building MyScéal as the baseline for our hypothesis, the next step involves the development of a cutting-edge graph-based image-text retrieval model. This model is intended to be an internal search engine within the new lifelog retrieval system. Given the limited research on utilizing graph-based structures in the lifelog retrieval task, coupled with their performance being unsuitable for comparison, a more effective approach is to evaluate their effectiveness in the image-text retrieval task that contains numerous models leveraging graph-based structures for a more appropriate comparison. In this chapter, I introduce two innovative cross-modal retrieval models that outperform existing state-of-the-art (SOTA) models. The goal is to address the second research question (RQ2): "*Can using graph-based structures improve the retrieval accuracy of a current state-of-the-art cross-modal retrieval model in the image-text retrieval task?*" As I have discussed in Section 2.2, the proposed models should follow the independent embedding approach to be applicable to the real-time retrieval challenge. The first approach[1] is to use the scene graph data structure to capture the relations between objects in images and texts. Despite the application of this idea in a lifelog retrieval system (LifeConcept) as discussed in Chapter 4, it proved to be a preliminary approach

[1]Source code is available at `https://github.com/m2man/LGSGM`

and did not achieve satisfactory results. Unlike LifeConcept, I instead embed entire graphs in both image and text modalities into vectors in the common space to calculate the similarity and perform the retrieval. The CLIP model was introduced in 2021, which is after my first proposed model was released, and outperformed all models in vision-and-language tasks. The main reason for its significant results is that CLIP was pre-trained on a massive external dataset consisting of 400 million images. Many later SOTA models followed this idea. However, it raised the challenge of how to surpass these models without pre-training on a huge dataset that might take several days with multiple GPUs. My second proposed approach[2] aims to address this issue by leveraging graph-based structures to amalgamate various SOTA models and enhance their accuracy. This chapter describes the two models and the experiments to validate their performance.

5.2 A Deep Local and Global Scene-Graph Matching Model

5.2.1 Scene-graph Structure

Conventional approaches to image-text retrieval usually ignore the interactions between objects appearing in images. Such objects' occurrences and interactions are equivalently useful and important in this field as they are usually mentioned in the text. Scene-graph presentation [137] is a suitable method for the image-text matching challenge due to its ability to capture the inter-relationship information. A scene-graph structure is a graph including nodes and edges connecting nodes together. In the image-text retrieval field, a node and an edge represent objects and the associations among objects detected in images or captions, as depicted in Figure 5.1 where the green rectangles are nodes in the graph indicating detected objects and the orange rectangles are edges illustrating the relations between them. This design has been applied and achieved excellent results in various fields

[2]Source code is available at `https://github.com/m2man/HADA-LAVIS`

[181, 182, 183, 184]. Recent research [53, 54] could obtain better results in this retrieval area by utilizing these relations information. Hence there is a promising result when applying this useful data structure in the retrieval field. Among the retrieval models utilizing a graph-based structure, SGM [53] incorporates the scene graph structure to capture object relations and achieves state-of-the-art performance. However, a significant drawback of SGM is its method of measuring graph similarity, where it divides them into sub-graphs, consequently reducing the overall graph detail. Furthermore, using ResNet as the backbone restricts the model's potential performance, especially considering the existence of more accurate CNN-based networks developed in recent years [134]. Additionally, SGM lacks the application of any normalization technique, making it prone to overfitting. I adopt the scene-graph structure from the SGM model as a foundation for my approach. To enhance this baseline model, I introduce an additional graph convolution network that captures both the global structure of the graph and its local representation, which is generated by the baseline. The selection of the SGM model as the baseline is based on its state-of-the-art performance in this domain, particularly among models that utilize the scene-graph structure.

5.2.2 Methodology

I tried to improve the performance of the SGM method by employing a graph convolution model to capture the overall information of a graph which can be considered as one of its weaknesses mentioned above. This was because it could encode the entire graph into a vector containing all summarized information of the graph, thus addressing the limitation of the SGM model which only focused on separated parts of the graph but underlooked the overall detail. My model utilized scene graph structures for images and sentences and encoded them into vector forms for storing overall essential information to calculate the similarity between graphs instead of operating at the graph level exclusively. I named this model "*Local and Global Scene Graph Matching*" (LGSGM). Compared to SGM, the

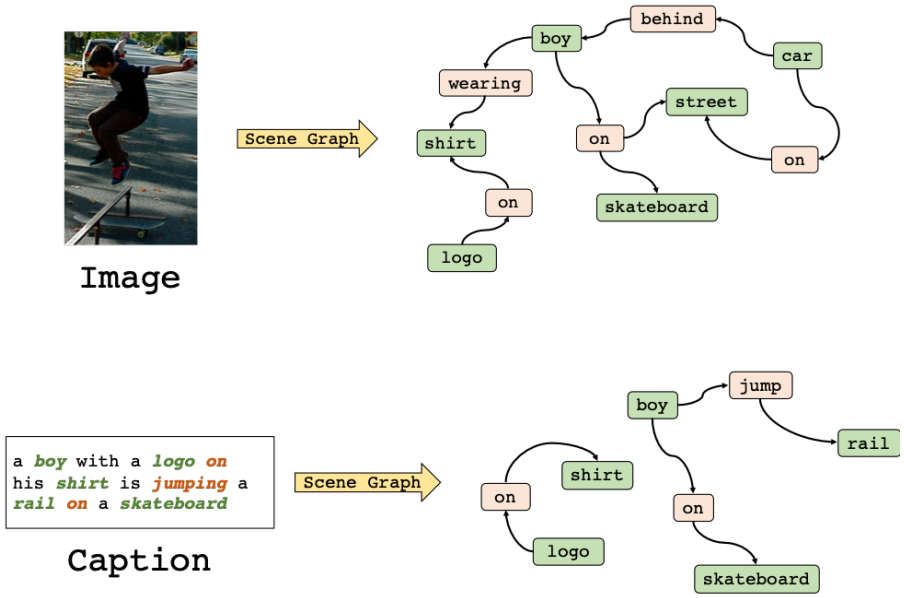


Figure 5.1: An example of scene graphs generated from an image and its caption. The green and the red color indicate objects and their interactions, respectively.

proposed LGSGM had two improvements. Firstly, the SGM measured the similarity between graphs by dividing them into sub-graphs, hence diminishing the overall detail of a graph. I overcame this problem by building a graph embedding module to summarise all information of a graph into a vector form and combining it with the graphical form to calculate the similarity of a pair of graphs. Secondly, the EfficinetNet model [134] was applied to extract the visual features of detected objects in images instead of using conventional Residual structure [120] in the SGM model as the former network was shown to be more accurate although having a simpler design. Moreover, I also integrated normalization techniques including Dropout and Batch Normalization to mitigate the potential of overfitting.

Figure 5.2 depicts the workflow of LGSGM that integrates an extra graph embedding stage to the SGM baseline. Initially, the preprocessing data stage started with the scene graph construction. The graphs of all images and texts were extracted beforehand and stored in the database. A generated scene graph of a query, which could be an image or a sentence, then went through a graph encoder module to get a graph feature. Meanwhile, each sample in the remaining modality

database also followed the same scheme but separately. All feature graphs were passed through a Siamese-structured [185] graph embedding layer to learn the summarised attributes of them, then the vector-level features of those graphs were obtained. The Siamese model can learn the similarity between two inputs which are visual features and textual features in this case. The similarity of a pair of graphs, which was a query graph and a sample graph in the database, could be calculated based on the feature graphs and their embedded vectors representing local and global information, accordingly. Finally, these scores were ranked descending to find the most relevant answers from the database.

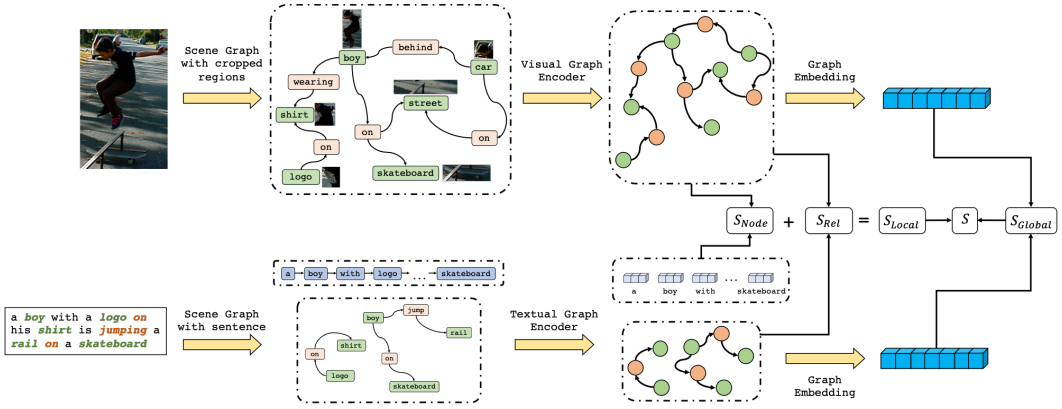


Figure 5.2: Our proposed LGSGM pipeline

5.2.2.1 Visual Graph Encoder

A scene graph of an image was established beforehand by using any available scene graph generation methods [186, 187, 188, 189]. The graph $\mathcal{G}$ after extracted could be seen as a tuple $\mathcal{G} = (O, B, R)$ where:

- $O = \{o_1, ..., o_{N_o}\}$ was the set of N_o semantic labels of detected objects appearing the image.
- $B = \{b_{o_1}, ..., b_{o_{N_o}} | b_{o_i} \in \mathbb{R}^4\}$ was the set of bounding boxes where b_{o_i} was the coordinate of the box of the corresponding object o_i in the set O.
- $R = \{r_1, ..., r_{N_r}\}$ was the set of N_r predicted relations between objects in

the image. Each r_m was a tuple of (o_i, p_{ij}, o_j) where p_{ij} was the label of the association between two objects o_i and o_j. It was noted that R could be seen as the set of edges connecting nodes, which were objects in O, of the scene graph $\mathcal{G}$.

As shown in Figure 5.3, after the graph was constructed, I embedded the label of the nodes and edges into vectors with the help of trainable word embedding layers to gain their semantic information. On the other hand, there was also rich information from the images themselves. I extracted an image feature for each object in O based on its associated bounding box in B. Regarding an edge p_{ij}, its image feature was computed through the union regions of two boxes b_{o_i} and b_{o_j}. These regions were fed into a CNN-based model, which was EfficientNet in my setting due to its high accuracy in many vision tasks with fewer parameters [134], to get the image features. The feature of one node or an edge was the fusion of its image feature and semantic features. I updated these features to learn the connection between nodes and edges features by using the Graph Convolution Network [140] (GCN) to get the final visual feature graph. The graph was finally fed to the Siamese graph embedding model to get its global representation.

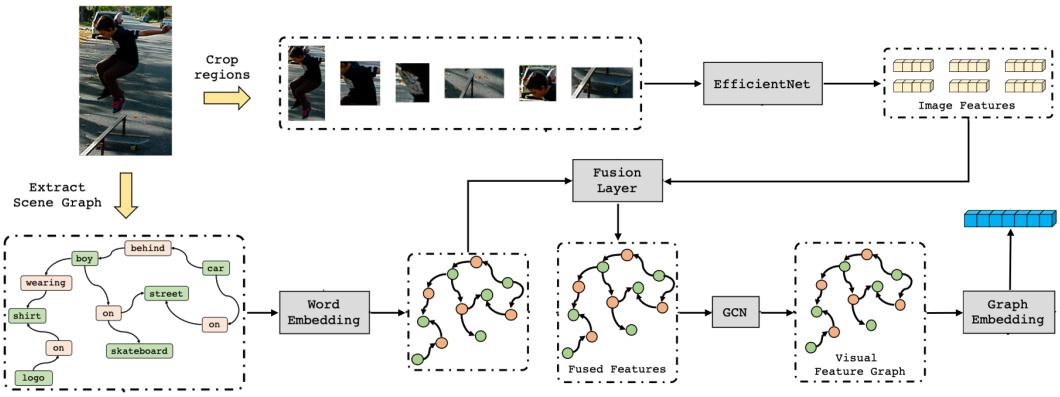


Figure 5.3: The workflow of an image in our LGSGM model.

Word Embedding. I exploited the information of semantic labels by embedding names of objects and relations in O and R into vectors with two distinct embedding layers for each. Each object label o_i and predicate label p_{ij} was transformed into

one-hot vector I_{o_i} and $I_{p_{ij}}$. Their embedding vectors (e_{o_i} and $e_{p_{ij}}$) were calculated as follows:

$$e_{o_i} = W_o I_{o_i} \ , \ W_o \in \mathbb{R}^{d_W \times C_o} \tag{5.1}$$

$$e_{p_{ij}} = W_p I_{p_{ij}} \ , \ W_p \in \mathbb{R}^{d_W \times C_p} \tag{5.2}$$

where W_o and W_p were the trainable parameters in the layer, C_o and C_p were the number of categories of objects and relations supported in the scene graph generation method. I set $d_W = 300$ and utilized the pre-trained embedding Glove model [190] to initialize W_o and W_p.

Image Features. An image feature v_{o_i} of an object o_i could be obtained by applying a pre-trained CNN-based network to the cropped region b_{o_i} around the object. Similarly, the cropped region for an edge p_{ij}, which was the area covering both b_{o_i} and b_{o_j} regions, was also used to extract its d_I-dimension image feature v_{ij}.

Fused Features. A simple trainable fully connected neural network (linear layer) was applied to the semantic feature's concatenated vector and the image feature to get the fused feature that could combine both modality detail. The fused representation of a node and an edge was achieved as follows:

$$u_{o_i} = f_{act}(W_u[v_{o_i} \ , \ e_{o_i}]) \tag{5.3}$$

$$u_{ij} = f_{act}(W_u[v_{ij} \ , \ e_{ij}]) \tag{5.4}$$

where [.] was the concatenating operation, f_{act} was an activation function, and $W_u \in \mathbb{R}^{d_F \times (d_I + d_W)}$ was the trainable parameters.

Graph Network. I used the GCN to learn the connection and update the fused features of nodes (u_{o_i}) and edges (u_{ij}) of the graph. The GCN works similarly to a normal CNN operation but is more flexible to many types of graph-structured data than only the grid data of a traditional CNN. In this GCN model, the features of a node were only updated based on itself solely to mitigate noise effects from surrounding nodes [53]. In contrast, because an edge bridges two nodes together, its

features should be related to those nodes' features to learn the association between them. Regarding an n-layer GCN model, updated features of a node and an edge at the l^{th} layer were formulated as follows:

$$h_{o_i}^{l} = \text{MLP}_o(h_{o_i}^{l-1}) \tag{5.5}$$

$$h_{p_{ij}}^{l} = \text{MLP}_p([h_{o_i}^{l-1}\ ,\ h_{p_{ij}}^{l-1}\ ,\ h_{o_j}^{l-1}]) \tag{5.6}$$

where MLP_o and MLP_p were two separate neural network models, and $h_{o_i}^{0} = u_{o_i}$ and $h_{p_{ij}}^{0} = u_{ij}$. The final output of the visual feature graph was the updated features of nodes and edges which were denoted as h_{o_i} and $h_{p_{ij}}$.

5.2.2.2 Textual Graph Encoder

Similarly, a sentence also followed the same process of images. Figure 5.4 illustrates the process of getting local and global features of the sentence. Firstly, the caption was converted into a scene graph to describe the relationships in detail. This triplet information, such as *"man-wears-hat"*, could be obtained by using the SPICE[3] technique [183]. This technique can extract a scene graph from a sentence by establishing syntactic dependencies among words in the sentence, resulting in a dependency syntax tree. This tree then is utilized to map from dependency structures to scene graphs representing the objects, attributes, and relationships through a rule-based system. However, relying on the graph only would ignore other important clues in the sentence. Therefore, two different modules were used to extract the data from both pathways. Initially, every word in the sentence was encoded by a word embedding module similar to that in the visual feature graph. Two distinct LSTM models [191] were then employed to learn the features of each word in the sentence as well as the features of each extracted relation triplet. I expanded the SGM model by introducing an additional GCN to the graph created from the triplets. This graph was used to summarise information of all relations in

[3]SPICE: Semantic Propositional Image Caption Evaluation

the sentence and compared to that of visual modality afterward.

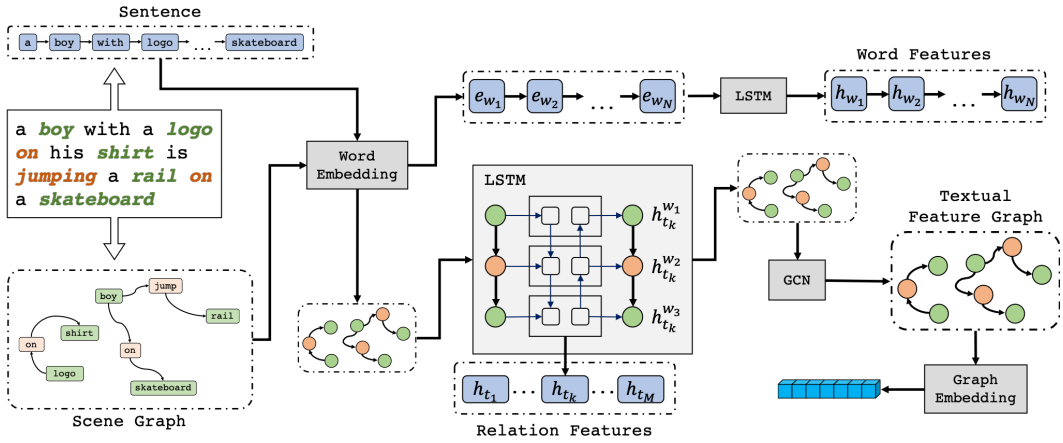


Figure 5.4: The workflow of a sentence in our LGSGM model. The N and M indicate the number of words and the number of relations in the sentence.

Word Embedding. It is worth noting that there are a massive number of vocabularies in a real-world deployment. Hence, it is not suitable for the word embedding part to learn all words in the entire semantic modality database as an overfitting issue will occur in the test condition. I constructed the dictionary from the textual database, excluding the test set. It mainly comprised frequently occurring common words in the database. I replaced the least common ones whose appearance frequency was less than four with the *"unknown"* word, which was considered a popular vocabulary. During the test phase, words not included in the dictionary were converted to *"unknown"*.

Word Features. After the embedding stage, the entire sentence went to the bidirectional LSTM hierarchy to get the features from both forwards and backward ways. Then the representation of a word w_i in the sentence, denoted as h_{w_i}, was the average of two paths and computed as follows:

$$h_{w_i} = \frac{\overrightarrow{\text{LSTM}_w}(e_{w_i}, \overrightarrow{h_{w_{i-1}}}) + \overleftarrow{\text{LSTM}_w}(e_{w_i}, \overleftarrow{h_{w_{i+1}}})}{2} \tag{5.7}$$

where LSTM_w was the bi-LSTM model, e_{w_i} was the word embedding vector whilst $\overrightarrow{h_{w_{i-1}}}$ and $\overleftarrow{h_{w_{i+1}}}$ were the hidden states of the word w_i from forward and backward

directions.

Graph Network. I organized the set of relations of a caption as $T = \{t_1, ..., t_{N_t}\}$ with N_t was the total number of relations in the sentence parsed by SPICE. Each triplet t_m was a sequence of words (w_i, w_{ij}, w_j) where the word w_{ij} was the association between two objects w_i and w_j such as ("man", "wears", "hat"). I used another bidirectional LSTM structure, named LSTM_t, and shared it among all triplet t_i for $i \in [1, N_t]$. The feature of a word in a triplet was the hidden state itself while the feature of the entire triplet was the last state of the sequence model. Specifically, they were calculated as:

$$h_{t_k}^{w_n} = \frac{\overrightarrow{\text{LSTM}_t}(e_{t_k}^{w_n}, \overrightarrow{h_{t_k}^{w_{n-1}}}) + \overleftarrow{\text{LSTM}_t}(e_{t_k}^{w_n}, \overleftarrow{h_{t_k}^{w_{n+1}}})}{2} \tag{5.8}$$

$$h_{t_k} = \frac{\overrightarrow{h_{t_k}^{w_i}} + \overleftarrow{h_{t_k}^{w_j}}}{2} \tag{5.9}$$

where triplet t_k was a sequence of words starting with w_i and ending with w_j, $e_{t_k}^{w_n}$ was the word embedding vector of w_n in t_k. An extra GCN model having a similar structure to that in the visual modality (Eq. 5.5 and Eq. 5.6) was then applied to the graph formed by the set T. Particularly, the graph would have $h_{t_k}^{w_n}$ as the feature of nodes and $h_{t_k}^{w_{ij}}$ for edges which all went through two neural networks to get the graph-level features of nodes and edges which were denoted as $hg_{t_k}^{w_n}$ and $hg_{t_k}^{w_{ij}}$.

5.2.2.3 Graph Embedding

After feature graphs of both modalities were extracted, I embedded them into vector forms using an attention mechanism called multi-scale node attention [192]. Given a graph $\mathcal{G}$ having N nodes $\{h_1, ..., h_N\}$ and M edges $\{r_1, ..., r_M\}$ where h_i and $r_j \in \mathbb{R}^d$ was a feature of a node and an edge accordingly, the embedded vector of $\mathcal{G}$, noted

as $a_{\mathcal{G}} \in \mathbb{R}^{2d}$, was obtained by the following formula:

$$a_{\mathcal{G}}^{h} = \sum_{n=1}^{N} \sigma(h_n^{\mathrm{T}} \mathrm{ReLU}(W_h(\frac{1}{N}\sum_{m=1}^{N} h_m)))h_n \tag{5.10}$$

$$a_{\mathcal{G}}^{r} = \sum_{n=1}^{M} \sigma(r_n^{\mathrm{T}} \mathrm{ReLU}(W_r(\frac{1}{M}\sum_{m=1}^{M} r_m)))r_n \tag{5.11}$$

$$a_{\mathcal{G}} = [a_{\mathcal{G}}^{h}, a_{\mathcal{G}}^{r}] \tag{5.12}$$

where σ and ReLU indicated the sigmoid and rectified linear unit activate function, W_h and $W_r \in \mathbb{R}^{d \times d}$ was the training parameter in the model, subscript T denoted the transpose operation, and [.] was the concatenation. I shared this attention structure with the visual feature graph and textual feature graph as a Siamese model to get their vector-level representation a_V and a_T respectively. It is noted that the node features and edge features of a visual graph were h_{o_i} and $h_{p_{ij}}$ while those of the textual graph were $hg_{t_k}^{w_n}$ and $hg_{t_k}^{w_{ij}}$.

5.2.2.4 Similarity and Loss Functions

The similarity between the two scene graphs was measured using both local and global approaches. In the local form, each part of a visual (textual) graph was compared to every part of a textual (visual) graph; for instance, matching on each node-level feature and on each edge-level feature. Regarding the global approach, the embedding vectors of the entire two graphs were used. This similarity score could be used to rank the retrieved result as higher scores mean more relevant answers.

Local similarity. The local score was the sum of a node score and an edge score. The former score was the average of the matching scores of all words in a sentence with their most relevant object detected in an image. The relevant score of a word w_i in the caption with an object in the image o_k was their dot product $h_{w_i}^{\mathrm{T}} h_{o_k}$. Assuming there were N_w words in the caption and N_o objects in the image,

the node score was calculated as:

$$S_{Node} = \frac{1}{N_w}\left(\sum_{i=1}^{N_w} \max_{k\in[1,N_o]} h_{w_i}^{\mathrm{T}} h_{o_k}\right) \tag{5.13}$$

Similarly, the edge score was the mean score between relations in the textual data with their most matching edge in the visual graph. This was formulated as:

$$S_{Rel} = \frac{1}{N_t}\left(\sum_{i=1}^{N_t} \max_{p_{ij}\in R} h_{t_i}^{\mathrm{T}} h_{p_{ij}}\right) \tag{5.14}$$

The local similarity then could be obtained as follows:

$$S_{Local} = S_{Node} + S_{Rel} \tag{5.15}$$

Global similarity. The global vectors provided the overall information of graphs. I used the conventional cosine distance to measure the degree of matching between two entire graphs. Specifically, the similarity score, S_{Global}, of two embedding vectors of visual graph a_V and textual graph a_T was:

$$S_{Global} = \frac{a_V^{\mathrm{T}} a_T}{\|a_V\|\,\|a_T\|} \tag{5.16}$$

Finally, the similarity between the two graphs took both local and global details into account. Hence it was calculated by the sum of them as follows:

$$S = S_{Local} + S_{Global} \tag{5.17}$$

Loss Function. My loss function used the bi-level similarity score S during the training phase. I applied the hardest negative triplet loss which was adopted in other SOTA models [53, 119]. Our loss function was:

$$L(k,l) = \max(0, m - S_{kl} + S_{k\hat{l}}) + \max(0, m - S_{kl} + S_{\hat{k}l}), \tag{5.18}$$

where m was a margin hyperparameter, S_{kl} was the similarity score of a true pair of image k and its caption l, and $\hat{k}$, $\hat{l}$ was the least matching image and sentence with l and k in the mini-batch, respectively.

5.2.3 Experiment Configuration

5.2.3.1 Dataset and Evaluation Metrics

The proposed model was evaluated on the Flickr30K dataset which is one of the most popular datasets in this image-text matching field due to its high quality of textual annotation compared to others [193]. The dataset consists of $31,783$ images, and each of them has five corresponding captions. An example of image-sentence pairs and their generated scene graph is illustrated in Figure 5.1. The Flickr30K data is split into training, validating, and testing sets. The number of the training images is $29,783$, whilst the two latter subsets both have $1,000$ images each. These splitting sets [116] have been used in all other SOTA retrieval models in the same research field.

As discussed in Section 3.3.2, the evaluation metric was a common Recall at K (R@K) value that has been used in many SOTA research. The R@K is the proportion of queries that the retrieval model finds their correct matching answers in the top K of the ranking result. In the experiments, I used three values of K, which are 1, 5, and 10, and their sum for the overall evaluation.

5.2.3.2 Model Setting

I used Neural Motifs [186] to generate a scene graph from an image which was similar to the SGM approach. I only kept the top $N_o = 36$ detected objects and $N_r = 25$ predicates in the result sorted by their confidence scores for each scene graph. Regarding visual features of objects, I employed the EfficientNet [134], which is considered to be more accurate in the image classification challenge, although having fewer parameters than ResNet as used in the SGM baseline. The region surrounding the object was firstly cropped based on its bounding box and resized to

the desired format, then went through the EfficientNet, which was excluded from the classification layer, to get the visual feature of the detected object. I set $d_I = d_F = 2048$ and chose the pre-trained EfficientNet-b5 model to get the 2048-dim feature vectors as comparable with the baseline. The number of layers in GCN models for both modalities and the LSTM models was 1. The dimension of the output vector from GCN was configured to be $d = 1024$, which was also the size of the hidden state of LSTM models. I also applied Dropout and Batch Normalisation to avoid the potential overfitting in both hierarchies. I used Swish [194] as an activate function when creating fused features (Eq. 5.3 and Eq. 5.4) and in GCN models[4]. The margin m in the loss function (Eq. 5.18) was selected as 0.35. The model was trained with the batch size of 128 at the learning rate of 0.0003 with Adam optimizer.

5.2.3.3 Baseline and Other Models

My primary focus was to compare the proposed LGSGM with the SGM model. Nevertheless, I also compared the proposed LGSGM with other SOTA models to provide a comprehensive perspective on the performance of my approach. Besides the SGM, other competing models were:

- PFAN [122] which used the position-focused attention network to extract the features of the location of objects in images.
- IMRAM [195] with the attention mechanism to learn the matching fragments between images and sentences.
- MMCA [196] applying attention and transformer components to exploit the relationship between objects in images and words in texts within themselves.

5.2.4 Results and Discussion

The LGSGM and others' results on Flickr30K are depicted in Table 5.1. The value in bold is the highest number in that metric which is higher meaning better

[4]I also tried other activation functions (such as ReLU, Leaky ReLU, or Tanh), but Swish had the best performance among these functions.

performance. *Image-to-Text* (I2T) indicates that a model needs to find texts that are relevant to a query image. On the contrary, *Text-to-Image* (T2I) is used when a query is a sentence. It is important to note that other models' metrics were taken from their original report since I used the same subset for training, validating, and testing.

Table 5.1: Performance of LGSGM and others on Flickr30K Dataset. R-Sum is the sum of all R@K metrics. The column **Total** is the sum of R-Sum of Image-to-Text and Text-to-Image.

Methods	**Image-to-Text**				**Text-to-Image**				**Total**
	R@1	R@5	R@10	R-Sum	R@1	R@5	R@10	R-Sum	
PFAN	70.0	91.8	95.0	256.8	50.4	78.7	86.1	215.2	472.0
IMRAM	74.1	**93.0**	**96.6**	**263.7**	53.9	79.4	87.2	220.5	484.2
MMCA	**74.2**	92.8	96.4	263.4	54.8	81.4	87.8	224.0	487.4
SGM	71.8	91.7	95.5	259.0	53.5	79.6	86.5	219.6	478.6
LGSGM	71.0	91.9	96.1	259.0	**57.4**	**84.1**	**90.2**	**231.7**	**490.7**

It was easier for all models to find captions when an image was given as a query than the reversed retrieving because the I2T retrieval section's scores were higher than those in the T2I retrieval section. Regarding I2T retrieval, both attention-based techniques, IMRAM and MMCA, shared similar results as there was no significant gap in all recall metrics. They performed best among all models where the differences in the R-Sum could be up to roughly 7%. These models were more accurate than the proposed LGSGM with 3.7% on the R-Sum. Compared with the SGM baseline, although I obtained higher scores at R@5 and R@10 with a margin less than 0.5%, the SGM model had a slightly better R@1 metric than LGSGM, which were 71.8% and 71%, respectively. The PFAN model performed worst with the lowest R-Sum. However, my LGSGM achieved the highest scores on all three recall metrics in the T2I field, which were 57.4%, 84.1%, and 90.2% accordingly. It made the proposed method have the best R-Sum of 231.7%, which was 7.7% higher than the MMCA in the second place. Compared with the SGM, the suggested improvement created a huge increase of 12.1% in R-Sum. The PFAN was still the model with the lowest recall.

In general, the proposed LGSGM outperformed all models with the total R-Sum at 490.7%. Especially, my model surpassed SGM by a large margin with a difference of 12% in the total R-Sum. This showed the benefits of the improvements in introducing global information in addition to local information as used in the original SGM. Although the attention approaches (IMRAM and MMCA) were better than my scene graph model in the I2T retrieval, LGSGM still managed to get the highest score on the remaining experiment. Moreover, LGSGM scores better than those models concerning the sum of recall of both subtasks. This result proved that LGSGM using the graph-based structure to capture the interaction information obtained the new SOTA score as mentioned in RQ2, especially our introduction of graph encoding component helped to increase the score up to 12% compared to the baseline SGM.

5.3 HADA: A Graph-based Amalgamation Framework

Although I developed LSSGM obtaining competitive performance compared to other SOTA retrieval models at that time, the CLIP model was introduced in 2021 and became the game-changer in not only this image-text retrieval field but also other vision-and-language tasks. This model can embed images and texts into vectors in a unified space where they can be compared to calculate similarity scores. Being pre-trained on a huge external dataset consisting of 400 million images is the main benefit that helps CLIP achieved the SOTA results in many tasks. CLIP also has the capability to recognize other information rather than only objects in an image, such as the context of the image or OCR or activities happening in the image. Many other retrieval models developed later are all pre-trained on massive datasets like CLIP and obtain new SOTA performance [133, 135, 136].

Hence, it was necessary to devise a new graph-based retrieval model capable of outperforming existing ones to address RQ2. However, proposing an entirely new model with a unique architecture and conducting extensive training on a massive

dataset using multiple GPUs, as commonly done with other state-of-the-art (SOTA) models, poses significant challenges. To tackle this, I introduced a graph-based framework called HADA[5] that can leverage the strength of combining pre-trained models, offering improved results compared to building a model from scratch. Firstly, I created a graph structure in which the nodes were the features extracted from the pre-trained models and the edges connecting them. The graph structure was employed to capture and fuse the information from every pre-trained model. Then a graph neural network was applied to update the connection between the nodes to get the representative embedding vector for an image and text. Finally, I employed cosine similarity to match images with their relevant texts and vice versa to ensure a low inference time. The main contribution was that HADA did not need to train on any external dataset and only required a single GPU to train due to the small number of trainable parameters but my proposed model still obtained new SOTA results.

5.3.1 Methodology

This section describes how my HADA addresses the retrieval challenge by combining any available pre-trained models. Figure 5.5 depicts the workflow of HADA. I started with two models ($N_{models} = 2$) as illustrated in Figure 5.5 for simplicity. However, HADA could be extended with a larger $N_{models} > 2$. HADA began using some pre-trained models to extract the features from each modality. I then built a graph structure to connect the extracted features together, which were fed into a graph neural network (GNN) later to update them. The outputs of the GNN were concatenated with the original global features produced by the pre-trained models. Finally, simple linear layers were employed to get the final representation embedding features for images and texts, which could be used to measure similarity to perform the retrieval. For evaluation, I extracted the representation features offline to guarantee high-speed inference time.

[5]HADA: Graph-based Amalgamation

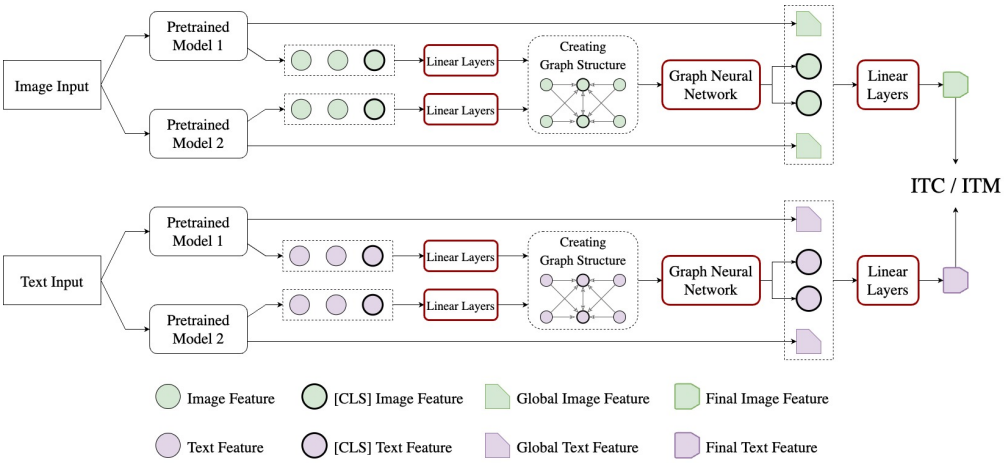


Figure 5.5: The pipeline of HADA. The red borders indicate trainable components. The ITM and ITC infer the training tasks which are Image-Text Matching and Image-Text Contrastive Learning accordingly. They will be discussed in Section 5.3.1.4.

5.3.1.1 Revisit State-of-the-art Models

I only used the pre-trained models without using the cross-modality transformer structure to extract features as depicted in Figure 2.13 in order to reduce the number of computations and ensure the high-speed inference time. They usually apply the architecture as illustrated in Figure 5.6. Basically, they used a unimodal encoder to get the features of an image or a text followed by a transformer network to embed them and obtain the *[CLS]* embedding. This *[CLS]* token was updated by one or many fully connected layers to become a representative global feature that could be compared with that of the remaining modality to get the similarity score.

HADA started with the output of the transformer layer from the pre-trained models. In detail, for an input image $\boldsymbol{I}$, I obtained the sequence of patch tokens from each model i denoted as $\mathbf{v^{(i)}} = \{v_{cls}^{(i)}, v_1^{(i)}, v_2^{(i)}, ..., v_{N_i}^{(i)}\}$, where $v_j^{(i)} \in \mathbb{R}^{d_v^{(i)}}$ and N_i was the length of the sequence. This length depends on the architecture of the image encoder network employed in the pre-trained model. For example, it can be the number of patches if the image encoder is a Vision Transformer (ViT) network [52], or the number of detected objects or regions of interest if the encoder is a Faster-RCNN model [123]. Additionally, I also extracted the global visual representation feature

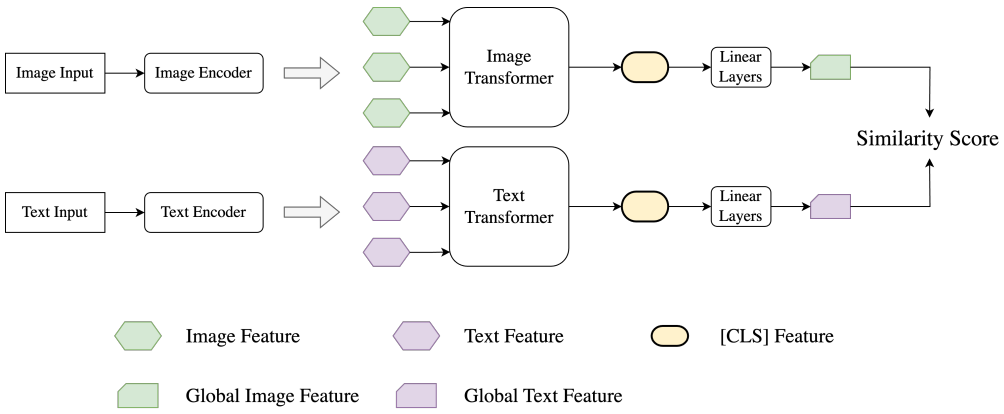


Figure 5.6: The pipeline of many SOTA models following the independent embedding approach.

$h_v^{(i)} \in \mathbb{R}^{d_h^{(i)}}$ from $v_{cls}^{(i)}$ as illustrated in Figure 5.6. Regarding the semantic modality, I used the same process as that of the visual modality. Specifically, I extracted the sequence of patch tokens $\mathbf{w^{(i)}} = \{w_{cls}^{(i)}, w_1^{(i)}, w_2^{(i)}, ..., w_L^{(i)}\}$ where $w_j^{(i)} \in \mathbb{R}^{d_w^{(i)}}$ and L was the length of the text, and the global textual representation embedding $h_w^{(i)} \in \mathbb{R}^{d_h^{(i)}}$ for an input text $\boldsymbol{T}$ using the pre-trained model i. The input model i matched a pair of an image $\boldsymbol{I}$ and a text $\boldsymbol{T}$ by calculating the dot product $\langle h_v^{(i)}, h_w^{(i)} \rangle$ of their global features. However, HADA used not only the global embedding but also the intermediate transformer tokens to make the prediction. As illustrated in Figure 5.5, I leveraged both [*CLS*] and the patch tokens to obtain the global embedding. The learned *[CLS]* tokens helped to improve the global features. Notably, the prediction was derived not only from the global embedding but also from the embedded output of the input models, as depicted in Figure 5.5. This approach allowed my prediction to capitalize on the benefits of pretraining on a massive dataset using the input models.

5.3.1.2 Create Graph Structure

Each pretrained model i produced different *[CLS]* features $v_{cls}^{(i)}$ and $w_{cls}^{(i)}$ for an image and text, respectively. Since my purpose was to combine the models, I needed to fuse these *[CLS]* tokens to obtain the unified ones for each modality separately. In

each modality, for example, the visual modality, HADA not only updated $v_{cls}^{(i)}$ based on $\mathbf{v^{(i)}}$ solely but also on those of the remaining pre-trained models $\{\mathbf{v^{(j)}} \mid j \neq i\}$. Because these $\mathbf{v}$ came from different models, their dimensions would not be similar to each other. Therefore, I applied a list of linear layers $f_v^{(i)} : \mathbb{R}^{d_v^{(i)}} \rightarrow \mathbb{R}^{d_p}$ to map them in the same dimensional space:

$$\mathbf{p^{(i)}} = \{f_v^{(i)}(x) | x \in \mathbf{v^{(i)}}\} = \{p_{cls}^{(i)}, p_1^{(i)}, p_2^{(i)}, ..., p_{N_i}^{(i)}\} \tag{5.19}$$

I performed a similar process for the textual modality to obtain:

$$\mathbf{s^{(i)}} = \{f_w^{(i)}(x) | x \in \mathbf{w^{(i)}}\} = \{s_{cls}^{(i)}, s_1^{(i)}, s_2^{(i)}, ..., s_L^{(i)}\}, \text{where } f_w^{(i)} : \mathbb{R}^{d_w^{(i)}} \rightarrow \mathbb{R}^{d_s} \tag{5.20}$$

I then used graph structures $\mathcal{G}_p = \{\mathcal{V}_p, \mathcal{E}_p\}$ and $\mathcal{G}_s = \{\mathcal{V}_s, \mathcal{E}_s\}$ to connect these mapped features together, where $\mathcal{V}$ and $\mathcal{E}$ denoted the list of nodes and edges in the graph $\mathcal{G}$ accordingly. In HADA, nodes indicated the mapped features. Specifically, $\mathcal{V}_p = \{\mathbf{p^{(i)}}\}$ and $\mathcal{V}_s = \{\mathbf{s^{(i)}}\}$ for all $i \in [1, N_{models}]$. Regarding edges, I symbolized $e_{a \rightarrow b}$ as a directed edge from node a to node b in the graph, thus the set of edges of the visual graph $\mathcal{E}_p$ and the textual graph $\mathcal{E}_s$ were:

$$\mathcal{E}_p = \{e_{x \rightarrow p_{cls}^{(j)}} \mid x \in \mathbf{p^{(i)}} \text{ and } i, j \in [1, N_{models}]\} \tag{5.21}$$

$$\mathcal{E}_s = \{e_{x \rightarrow s_{cls}^{(j)}} \mid x \in \mathbf{s^{(i)}} \text{ and } i, j \in [1, N_{models}]\} \tag{5.22}$$

To be more detailed, I created directed edges that go from every patch feature to the *[CLS]* feature, including from the *[CLS]* itself, for all pre-trained models but not in the reverse direction, as shown in Figure 5.5. The reason is that *[CLS]* was originally introduced as a representation of all input data, so it would summarize all patch tokens [50, 52, 197]. Therefore, it would be the node that receives information from other nodes in the graph. This connection structure ensured that HADA can update the *[CLS]* tokens based on the patch tokens from all pre-trained models in a fine-grained manner.

5.3.1.3 Graph Neural Network

There are many GNN structures have been introduced recently [198, 199, 200, 201, 202, 203]. HADA applied the modified Graph Attention Network (GATv2), which was recommended to be used as a baseline whenever employing a GNN [203], to fuse the patch features from different pre-trained models together to get the unified *[CLS]* features. Let $\mathcal{N}_k = \{x \in \mathcal{V} \mid e_{x \to k} \in \mathcal{E}\}$ be the set of neighbor nodes from which there is an edge connecting to node k in the graph $\mathcal{G}$. GATv2 uses a scoring function se to weight every edge indicating the importance of the neighbor nodes x in $\mathcal{N}_k$ before updating the node $k \in \mathbb{R}^d$:

$$se(e_{x \to k}) = \mathbf{A}^\top \text{LeakyRELU}(\mathbf{W}_1 x + \mathbf{W}_2 k]) \tag{5.23}$$

where $\mathbf{A} \in \mathbb{R}^{d'}$, $\mathbf{W}_1 \in \mathbb{R}^{d' \times d}$, and $\mathbf{W}_2 \in \mathbb{R}^{d' \times d}$ are learnable parameters. These weights are then normalized across all neighbor nodes in $\mathcal{N}_k$ by using a softmax function to get the attention scores:

$$\alpha_{e_{x \to k}} = \frac{\exp(se(e_{x \to k}))}{\sum_{y \in \mathcal{N}_k} \exp(se(e_{y \to k}))} \tag{5.24}$$

The updated node $k' \in \mathbb{R}^{d'}$ is then calculated based on its neighbors in $\mathcal{N}_k$, including k if we add an edge connect it to itself:

$$k' = \sigma(\sum_{x \in \mathcal{N}_k} \alpha_{e_{x \to k}} \cdot \mathbf{W}_1 x) \tag{5.25}$$

where σ is a nonlinear activate function. Furthermore, this GATv2 network can be enlarged by applying a multi-head attention structure, and improve performance [200]. The output now is a concatenation of each head output, which is similar to Transformer architecture. An extra linear layer is used at the end to convert these concatenated nodes to the desired dimensions.

I used different GATv2 structures with H attention heads for each modality in

this stage, as illustrated in Figure 5.5. HADA took the input graphs $\mathcal{G}_p$ and $\mathcal{G}_s$ with nodes $\mathcal{V}_p$ and $\mathcal{V}_s$ in the vector space of d_p and d_s dimensions and updated them to $\mathcal{V}'_p = \{\mathbf{p}'^{(\mathbf{i})}\}$ and $\mathcal{V}'_s = \{\mathbf{s}'^{(\mathbf{i})}\}$ with dimensions of d'_p and d'_s. I then concatenated the updated *[CLS]* nodes p'_{cls} and s'_{cls} from all pre-trained models with their corresponding original global embedding h_v and h_w. Finally, I fed them into a list of linear layers to get the normalized global representation $h_p \in \mathbb{R}^{d_h}$ and $h_s \in \mathbb{R}^{d_h}$.

5.3.1.4 Training Tasks and Loss Function

Image-Text Contrastive Learning. HADA encoded the input image $\boldsymbol{I}$ and text $\boldsymbol{T}$ to h_p and h_s, accordingly. I used a similarity function that was a dot product $S(\boldsymbol{I}, \boldsymbol{T}) = \langle h_p, h_s \rangle = h_p^\top h_s$ to ensure that a pair of relevant image-text (positive pair) would have a higher similar representation compared to irrelevant pairs (negative pairs). The contrastive loss for Image-to-Text (I2T) retrieval and Text-to-Image (T2I) retrieval for the mini-batch of M relevant pairs $(\boldsymbol{I}_m, \boldsymbol{T}_m)$ were:

$$\mathcal{L}_{I2T}(\boldsymbol{I}_m) = -\log \frac{\exp(S(\boldsymbol{I}_m, \boldsymbol{T}_m)/\tau)}{\sum_{i=1}^{M} \exp(S(\boldsymbol{I}_m, \boldsymbol{T}_i)/\tau)} \tag{5.26}$$

$$\mathcal{L}_{T2I}(\boldsymbol{T}_m) = -\log \frac{\exp(S(\boldsymbol{T}_m, \boldsymbol{I}_m)/\tau)}{\sum_{i=1}^{M} \exp(S(\boldsymbol{T}_m, \boldsymbol{I}_i)/\tau)} \tag{5.27}$$

where τ was a temperature parameter that could be learned during training. Such contrastive learning has been used in many vision-and-language models and has been proven to be effective [51, 130, 135, 136]. In my experiment, I trained HADA with the loss that optimizes both subtasks:

$$\mathcal{L}_{ITC} = \frac{1}{M} \sum_{m=1}^{M} (\mathcal{L}_{I2T}(\boldsymbol{I}_m) + \mathcal{L}_{T2I}(\boldsymbol{T}_m)) \tag{5.28}$$

Inspired by ALBEF [135], I also applied momentum contrast (MoCo) [204] and their momentum distillation strategy for this unsupervised representation learning to cope with the problem of noisy information in the dataset and improve accuracy. I trained

the task by using the same configuration as their work.

Image-Text Matching. This objective is a binary classification task to distinguish irrelevant image-text pairs that have similar representations. This task ensures that they are different in fine-grained details. I implemented an additional disciminator layer $dc : \mathbb{R}^{4d_h} \rightarrow \mathbb{R}$ on top of the final embedding features h_p and h_s to classify whether the image $\boldsymbol{I}$ and the text $\boldsymbol{T}$ is a positive pair or not:

$$dc(h_p, h_s) = \text{sigmoid}(\mathbf{W}^\top [h_p \| h_s \| \text{abs}(h_p - h_s) \| h_p \odot h_s]) \tag{5.29}$$

where $\mathbf{W} \in \mathbb{R}^{4d_h}$ was trainable parameters, $\|$ indicated the concatenation, abs(.) was the absolute value, and $\odot$ denoted elementwise multiplication. I used the binary cross-entropy loss for this ordinary classification task:

$$\mathcal{L}_{itm}(\boldsymbol{I}, \boldsymbol{T}) = y\log(dc(h_p, d_s)) + (1 - y)\log(1 - dc(h_p, d_s)) \tag{5.30}$$

where y was the one-hot vector representing the ground truth label of the pair.

For each positive pair in the minibatch of M positive pairs, I sampled one hard negative text for the image and one hard negative image for the text. These negative samples were chosen from the current mini-batch in which they were not relevant based on the ground-truth labels but had the highest similarity dot product score. Therefore, the objective for this task was:

$$\mathcal{L}_{ITM} = \frac{1}{3M} \sum_{m=1}^{M} (\mathcal{L}_{itm}(\boldsymbol{I}_m, \boldsymbol{T}_m) + \mathcal{L}_{itm}(\boldsymbol{I}_m, \boldsymbol{T}'_m) + \mathcal{L}_{itm}(\boldsymbol{I}'_m, \boldsymbol{T}_m)) \tag{5.31}$$

where $\boldsymbol{T}'_m$ and $\boldsymbol{I}'_m$ were the hard negative text and image samples in the mini-batch that were corresponding with the $\boldsymbol{I}_m$ and $\boldsymbol{T}_m$, respectively. The final loss function in HADA was:

$$\mathcal{L} = \mathcal{L}_{ITC} + \mathcal{L}_{ITM} \tag{5.32}$$

5.3.2 Experiment Configuration

5.3.2.1 Dataset and Evaluation Metrics

Similar to the experiment of LGSGM, I evaluated HADA on the well-known Flickr30K dataset. Furthermore, I also used the MSCOCO dataset [43] which consists of 123,000 images and each image contains five relevant texts or captions that describe the image (same format as Flickr30K). I split the MSCOCO dataset into train/evaluate/test on 113,000/5,000/5,000 images which are the standard split [116] that other works have been using. These two datasets are the most commonly used datasets in evaluating image-text retrieval models. Regarding the evaluation metric, I still used R@K as I did in LGSGM's experiment and this metric has been employed in other models. The definition of this metric is described in Section 3.3.2.

5.3.2.2 Model Setting

In this experiment, I combined two SOTA models that had available pre-trained weights fine-tuned on the Flickr30K dataset: ALBEF[6] and LightningDOT[7]. I chose these two models because I wanted to combine different models with distinct embedding backbones to utilize the advantages of each of them. Although ALBEF and LightningDOT used the same BERT architecture to encode a text, the former model employed the ViT network to encode an image, while the latter model applied the Faster-RCNN model. Both of them followed the fusion approach as we discuss in Section 2.2.3. Therefore, none of them used the cross-modality transformer structure when retrieved to ensure the fast inference speed[8]. While LightningDOT was considered a pioneer attempt at the fusion method, ALBEF was one of the SOTA models obtaining higher accuracy than CLIP in most vision-and-language tasks including the image-text retrieval task.

[6] `https://github.com/salesforce/ALBEF`

[7] `https://github.com/intersun/LightningDOT`

[8] Indeed, these two models applied the cross-modality transformer network to rerank the initial result in the subsequent step. However, I did not focus on this stage.

Regarding ALBEF, their ViT network encoded an image to 577 patch tokens including the *[CLS]* one ($N_{ALB} = 576$ and $d_v^{(ALB)} = 768$). This *[CLS]* was projected to the lower dimension to obtain the global feature ($d_h^{(ALB)} = 256$). Because LightningDOT encoded an image based on the detected objects produced by the Faster-RCNN model, its N_{DOT} varied depending on the number of objects in the image. The graph neural network, unlike other conventional CNNs, could address this inconsistent number of inputs due to the flexible graph structure with nodes and edges. Unlike ALBEF, the dimensions of image features and global features from LightningDOT were the same with $d_v^{(DOT)} = d_h^{(DOT)} = 768$. In terms of text encoder, the outputs of both models were similar since they used the same BERT network: $d_w^{(ALB)} = d_w^{(DOT)} = 768$. I projected these features to a latent space where $d_p = d_s = 512$, which was the average of their original dimensions. I used a 1-layer GATv2 network with $H = 4$ multi-head attention to update the graph features while still keeping the input dimensions of $d'_p = d'_s = 512$. I also applied Dropout with $p = 0.7$ in linear layers and graph neural networks. In total, the proposed HADA contained less than 10 million trainable parameters.

The input models were already pre-trained on several large external datasets. For example, ALBEF was pre-trained on 14 million images compared to only 29,000 images on Flickr30K that I used to train HADA. Therefore, I utilized this advantage in the final prediction instead of training HADA in millions of samples. I modified the similarity score to a weighted sum of the initial HADA predictions and the original prediction of the input models. Thus, the weighted similarity score that I used was:

$$S(\boldsymbol{I}, \boldsymbol{T}) = (1 - \alpha)\langle h_p, h_s \rangle + \alpha \langle h_v^{(ALB)}, h_w^{(ALB)} \rangle \tag{5.33}$$

where α was a trainable parameter. I did not include the original result of the LightningDOT model since its result was lower than ALBEF by a large margin and, hence, could have a negative impact on overall performance[9].

[9] I tried including the LightningDOT in the weighted similarity score, but the result was lower than using only ALBEF.

I trained HADA for 50 epochs (early stopping[10] was implemented) using the batch size of 20 on one NVIDIA RTX3080Ti GPU. The AdamW [205] optimizer was used with a weight decay of 0.02. The learning rate was set at $1e^{-4}$ and decays to $5e^{-6}$ following cosine annealing [206]. Similarly to ALBEF, I also applied RandAugment [207] for data augmentation. The initial temperature parameter was 0.07 [208], and I kept it in the range of $[0.001, 0.5]$ during training. To mitigate the dominant effect of ALBEF global features on the weighted similarity score, I first trained HADA with $\alpha = 0$. After the model had converged, I continued to train but initially set $\alpha = 0.5$ and kept it in the range of $[0.1, 0.9]$.

5.3.2.3 Baseline and Other Models

I added UNITER, and VILLA [209] to the comparison. These approaches both applied cross-modality transformer architecture and gained the SOTA results in the image-text retrieval task. While the former model was one of the pioneers in the cross-embedding approaches, the latter model was the first attempt at applying adversarial training on vision-and-language tasks. Additionally, I built two baselines that also integrated ALBEF and LightningDOT as input to show the advantages of using graph structures to fuse these input models.

Baseline B1. This baseline did not require any additional training. The baseline calculated the average of the original ranking results obtained from ALBEF and LightningDOT, interpreting this average as the distance between images and text. This meant that the relevant pairs should be ranked at the top, whilst irrelevant pairs would rank lower.

Baseline B2. Instead of using a graph structure to fuse the features extracted from the pre-trained models, I only concatenated their global embedding and fed them into the last linear layers to obtain the unified features. I trained this baseline B2 following the same strategy as described in Section 5.3.2.2 using the weighted similarity score.

[10] In my experiment, it converged after roughly 20 epochs.

5.3.3 Results and Discussion

5.3.3.1 Fusing ALBEF and LightningDOT

Table 5.2 illustrates the evaluation metrics of the different models in the Flickr30K dataset. Many image-text retrieval models contain the cross-modality transformer rerank component to enhance their accuracy, as discussed in Section 2.2.3. However, this time-consuming step can introduce a slight delay in calculations. In many practical scenarios, only the preliminary embedded features are utilized, and the rerank module is omitted. Moreover, my primary aim was to avoid using the cross-modality transformer for the image-text retrieval task. Therefore, I only presented results from LightningDOT and ALBEF, excluding the time-consuming reranking step. If the model has a better initial result, it can have a better-reranked result by using the cross-modality transformer later.

Table 5.2: Performance of models on Flickr30K Dataset. The symbol † indicates the results are originally reported in their research, while others are from my re-implementation using their public pre-trained checkpoints. The column △R shows the gap compared to ALBEF.

Methods	**Image-to-Text**				**Text-to-Image**				**Total**	**△R**
	R@1	R@5	R@10	R-Sum	R@1	R@5	R@10	R-Sum		
UNITER†	87.3	98	99.2	284.5	75.56	94.08	96.76	266.4	550.9	↓13.68
VILLA†	87.9	97.2	98.8	283.9	76.26	94.24	96.84	267.34	551.24	↓13.34
LightningDOT	83.6	96	98.2	277.8	69.2	90.72	94.54	254.46	532.26	↓32.32
LightningDOT†	83.9	97.2	98.6	279.7	69.9	91.1	95.2	256.2	535.9	↓28.68
ALBEF	92.6	99.3	99.9	291.8	79.76	95.3	97.72	272.78	564.58	0
B1	90.7	99	99.6	289.3	79.08	94.5	96.94	270.52	559.82	↓4.76
B2	91.4	99.5	99.7	290.6	79.64	95.34	97.46	272.44	563.04	↓1.54
HADA	**93.3**	**99.6**	**100**	**292.9**	**81.36**	**95.94**	**98.02**	**275.32**	**568.22**	**↑3.64**

It is clear that the proposed HADA obtained the highest metrics at all recall values compared to others. HADA achieved slightly better R@5 and R@10 in Image-to-Text (I2T) and Text-to-Image (T2I) subtasks than ALBEF. However, the gap became more significant at R@1. I improved the R@1 of I2T by 0.7% (92.96 → 93.3) and the R@1 of T2I by 1.6% (79.76 → 81.36). In total, my R-Sum was 3.64% higher than that of ALBEF (564.58 → 568.22).

The experiment also showed that LightningDOT, which encoded images using

Faster-RCNN, performed worse than ALBEF when its total R-Sum was lower than that of ALBEF by approximately 30%. One possible explanation for this is that LightningDOT was pre-trained on 4 million images, which is significantly fewer than the 14 million images used to train ALBEF. Furthermore, the former model used RCNN to detect objects in images while the latter model utilized the ViT network. Although also using object detectors as the backbone but applying a cross-modality network, UNITER and VILLA surpassed LightningDOT by a large margin at 15%. It proved that this intensive architecture can make a large impact on multimodal retrieval, although it is time-consuming.

Regarding the two baselines, B1 and B2, both of them failed to get better results than the input model ALBEF. Model B1, using the simple strategy of taking the average ranking results and having no learnable parameters, performed worse than model B2, which used a trainable linear layer to fuse the pre-trained features. Nevertheless, the R-Sum of B2 was lower than HADA by 5.18%. It showed the advantages of using a graph structure to fuse the information between models to obtain a better result.

5.3.3.2 Fusing CLIP and BLIP

To show the stable performance of HADA, I used it to combine two other different pre-trained models, including BLIP and CLIP. While CLIP is well-known for its application in many retrieval challenges, BLIP is the enhanced version of ALBEF with the bootstrapping technique in the training process and is the model obtaining the highest accuracy at that time. I used the same configuration as described in 5.3.2.2 to train and evaluate HADA in Flickr30K and MSCOCO datasets. The pre-trained BLIP and CLIP were collected from the LAVIS library [210]. It is noted that the CLIP I used in this experiment was the zero-shot model since the fine-tuned CLIP for these datasets has not been available yet.

Table 5.3 shows the comparison between HADA and the input models. CLIP performed worst on both Flickr30K and MSCOCO with huge differences compared

Table 5.3: Performance of models on the test set in Flickr30K and MSCOCO datasets. The column △R shows the difference compared to BLIP in that dataset.

Dataset	**Methods**	**Image-to-Text**				**Text-to-Image**				**Total**	**△R**
		R@1	R@5	R@10	R-Sum	R@1	R@5	R@10	R-Sum		
Flickr30K	BLIP	94.3	99.5	99.9	293.7	83.54	96.66	98.32	278.52	572.22	0
	CLIP	88	98.7	99.4	286.1	68.7	90.6	95.2	254.5	540.6	↓31.62
	HADA	**95.2**	**99.7**	**100**	**294.9**	**85.3**	**97.24**	**98.72**	**281.26**	**576.16**	**↑3.94**
MSCOCO	BLIP	**75.76**	**93.8**	**96.62**	**266.18**	57.32	81.84	88.92	228.08	494.26	0
	CLIP	57.84	81.22	87.78	226.84	37.02	61.66	71.5	170.18	397.02	↓97.24
	HADA	75.36	92.98	96.44	264.78	**58.46**	**82.85**	**89.66**	**230.97**	**495.75**	**↑1.49**

to BLIP and HADA because CLIP was not fine-tuned for these datasets. Regarding the Flickr30K dataset, HADA managed to improve the R-Sum by more than 3.9% compared to that of BLIP. Additionally, HADA obtained the highest scores in all metrics for both subtasks. The proposed framework also increased the R-Sum of BLIP by 1.49% in the MSCOCO dataset. However, BLIP performed slightly better than HADA in the I2T subtask, while HADA achieved higher performance in the T2I subtask.

5.4 Discussion and Conclusion

At first, I proposed LGSGM which utilized the scene-graph structure to capture interaction information in images and texts. The experiment on the Flickr30K dataset showed that LGSGM surpassed other models, especially in the T2I subtask which is similar to the lifelog retrieval task. Despite achieving a slightly lower score in the I2T subtask, the introduction of the graph encoding stage to extract global information from the graph significantly boosted the total recall score by over 10%, surpassing the baseline SGM model. Nevertheless, the release of CLIP has shown the benefits of pre-training on hundreds of millions of images in the image-text retrieval task. I proposed HADA, a graph-based fusion framework that combines pre-trained image-text retrieval models to enhance overall performance by leveraging the graph-based structure to fuse fine-grained features from input models without extensive pre-training again from scratch. The results proved that HADA increased the recall scores of the two input models on both popular datasets

including Flickr30K and MSCOCO. The main contribution of HADA is that my model with a few trainable parameters did not need to pre-train on huge external datasets as other models have been doing but still got the SOTA score. The performance of LGSGM and HADA gives the answer to RQ2 which is "*Can we improve the retrieval performance of a current state-of-the-art cross-modal retrieval model in the image-text retrieval task by using graph-based structures?*".

5.5 Chapter Summary

In this chapter, I manage to seek the answer to my second research question about finding the SOTA cross-modal retrieval model using graph-based structures in the general image-text retrieval task. I address this question by developing two novel models in which LGSGM is created before CLIP is released and HADA is after that. My models all surpass the current SOTA models at the time they are introduced. Both models follow the independent embedding pipeline to be able to apply in lifelog retrieval later which needs a search engine to run in a real-time manner. While LGSGM employs scene-graph structures to directly capture relations between objects in images, HADA is a fusing framework that uses graph-based structures to combine any pre-trained SOTA models to increase their overall accuracy. The experiments on Flickr30K and MSCOCO datasets show that both LGSGM and HADA achieve higher recall metrics than baselines and marked new SOTA results. Unlike CLIP and BLIP, the proposed HADA, with only approximately 10 million trainable parameters, achieves superior performance without the need for a pre-training stage. The next chapter is about applying graph-based retrieval models to lifelog retrieval field which includes a pilot study and a comprehensive user study experiment of using HADA to solve LSC queries.

Chapter 6

Graph-based Lifelog Retrieval System

6.1 Introduction

After building the graph-based model for the image-text retrieval task, I move to the next step to apply it to the lifelog retrieval task. I integrate the model proposed in Chapter 5 as a backend search engine of a lifelog retrieval system MyScéal introduced in Chapter 4. Subsequently, I conducted a user study to assess how novice users interacted with both the concept-based and graph-based approaches, aiming to compare their effectiveness. The result of the user experiment will shed light on finding the answer to the last research question which is "*To what extent can a graph-based lifelog interactive retrieval system improve retrieval performance compared to the conventional concept-based system?*". My user study follows the official format of the LSC competition in which a query will be gradually revealed to novice volunteers and I evaluate their performance based on their submissions and retrieval time as discussed in Section 3.3.1.1. I eliminate potential bias factors that could influence the outcome of the experiment by ensuring that both systems utilize the same user interface, thereby the comparison remains unaffected by UI factor. Furthermore, all volunteers in this user study do not have any prior knowledge about either lifelog dataset or lifelog retrieval task. This will help me to have a fair comparison between the two approaches. However, before conducting this comprehensive experiment, I carry out a simple pilot study

to have a preliminary insight into the effectiveness of employing graph-based structures to capture relations between concepts in images and texts. Both experiments are designed to mitigate the issue of learning bias when a user might gain knowledge during their first attempt leading to an impact on their performance in the second run on the remaining approach.

6.2 Pilot Study: Scene Graph Matching in Lifelog Retrieval

Although the LifeConcept system applied the scene-graph structure to the lifelog retrieval task in LSC'20 and LSC'21 [146, 175], their implementation was limited and not clearly described. Hence, it inspired me to have further investigation into how to apply this graph-based structure to lifelog retrieval to capture the interactions between concepts in images and semantic queries as well as to evaluate its performance compared to the conventional concept-based method which does not focus on the interaction between objects in the lifelog data or the query[1]. I first describe how I used the scene-graph structure to address the lifelog retrieval task including how I created a graph from an image or a query and how to compare them to perform the search. Then, I outline the experimental design used to compare this graph-based approach with the MyScéal system, as detailed in Section 4.2.

Figure 6.1 depicts my pipeline of using the scene graph in the lifelog retrieval task. Each lifelog image was converted to a graph, followed by an embedding stage to be stored as an object and relation matrices. A query was then parsed to a graph and encoded to matrices, which would be compared to those of each image by the scoring function. Given that I was working with multimodal lifelogs, the graph-matching process's outcome could be filtered by other information, such as geolocation or time which are automatically extracted from the query, and was then

[1]Source code is available at `https://github.com/m2man/MMM2021-LifelogRetrieval`

returned as a ranked list. This approach facilitated the capture of the interactions between objects in images and the comparison of them with those described in the textual input which could not be fully captured using the concept-based approach.

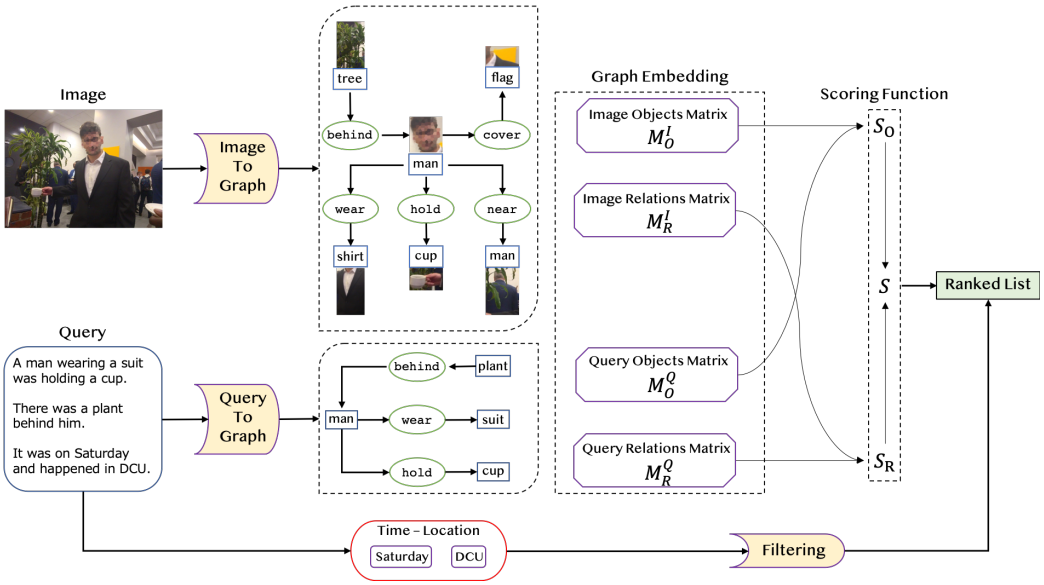


Figure 6.1: The overview of our scene-graph indexing approach for lifelog data and lifelog queries for retrieval purposes.

6.2.1 Data Indexing

As my system aimed to solve the interaction between objects within an image and a semantic query by using a scene graph, it raised the challenge of how to represent these two distinct types of data in a standard graph structure.

6.2.1.1 Image to Graph

Although there were many methods for generating a scene graph for a given image, Neural Motifs [139] was chosen due to its accurate performance [211]. A generated scene graph, noted as $\mathcal{G}$, contained a set O indicating detected objects in the image, with its corresponding set of bounding box B, and the set of visual relations R where:

- $O = \{o_1, ..., o_m\}$: m labels of recognised objects in the image. Each object o_i was a single node in the graph $\mathcal{G}$.

- $B = \{b_1, ..., b_m\}$: m bounding boxes of objects in O respectively in which $b_i = \{x_i, y_i, width, height\} \in \mathbb{R}^4$ where (x_i, y_i) was the top-left coordinates of the object $o_i \in O$.

- $R = \{r1, ..., r_n\}$: n detected relationships between objects. Each r_k was a triplet association of a starting object $o_i \in O$, an ending object $o_j \in O$, and a predicate $p_{i \to j} \in \mathcal{P}$ where $\mathcal{P}$ was a set including all pre-defined labels of predicates in the Visual Genome [212]. These relations could be considered as edges in $\mathcal{G}$.

All elements in each set were assigned their confidence score after running the Neural Motif model. The Neural Motif is a prediction model that can recognize objects in a given image and generate a scene graph describing interactions between the detected objects. I first removed inaccurate prediction by setting a threshold for object and relation. To expand the graph to obtain more interaction information in the image that could be not entirely captured by the model, I then created a fully connected graph of $\mathcal{G}$, called $\mathcal{G}_{fc}$, in which there was an edge connecting any two nodes. $\mathcal{G}_{fc}$ was obtained by building missing edges in $\mathcal{G}$ with the procedure of visual dependency representations (VDR) [213] based on the bounding box set B. The starting node and the ending node of the constructed edge could be decided based on their predicted scores, whose higher score would be the starting subject and lower is the ending object. One drawback of this $\mathcal{G}_{fc}$ was that there were many noisy and unimportant relations since not all objects correlated to others. I applied Maximum Spanning Tree on the graph $\mathcal{G}_{fc}$ to remove the least meaningful edges with low weights to get $\mathcal{G}_{mst}$. The Maximum Spanning Tree is an algorithm to create a graph having the minimum number of edges where each node has at least one edge connecting to another node. It achieves this by iteratively selecting edges with the highest weights, thus prioritizing connections that contribute the most to the overall weight of the tree. Starting from an arbitrary node, the algorithm incrementally adds edges until all nodes are connected or until further additions violate the tree's integrity. In my

scene graph, the weight of an edge was the sum of both nodes' scores and that of the predicate connecting them. The score of a predicate will be the score of r_j if this edge is in $\mathcal{G}$ or 0 if it is created by the VDR. It is worth noting that I only filtered out edges from VDR and retained the original relations in $\mathcal{G}$. In general, the expansion stages of getting the $\mathcal{G}_{mst}$ was to enlarge the set R and left two sets O and B intact. The entire process is illustrated in Figure 6.2 in which a blue node and an arrow represent an object and a relation respectively. The new relations generated by the VDR are illustrated by dotted arrows. A green node depicts a detected object in the image but was not included in the generated scene graph. Red arrows show relations discarded by the Maximum Spanning Tree but still kept as they were originally in the initially predicted scene graph.

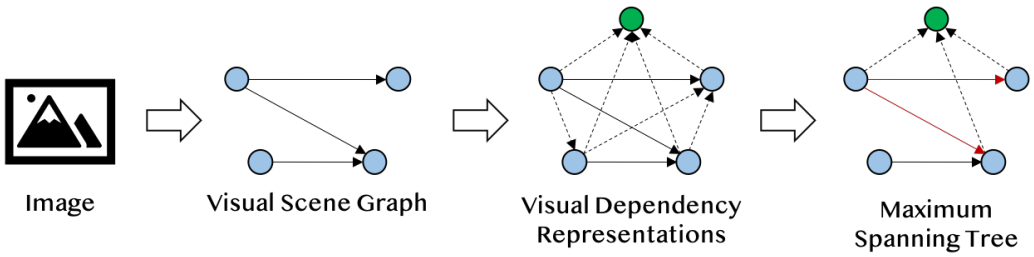


Figure 6.2: Image to Graph Pipeline.

6.2.1.2 Query to Graph

Besides images, a textual query may also contain several objects and the relations between them. For my proposed process, it was vital to have a graph describing the context of the topic. I applied the rule-based approach[2] [214] inspired by the Stanford Parser technique [215] to generate a semantic graph representing query items and their interactions, as described in the query text. This technique parsed a sentence into a graph based on the part-of-speech tagging of words in the sentence. Nouns in the sentence are nodes in the graph and edges in the graph are relations between nouns described in the sentence. Before applying this parser, any location and time query information was extracted and archived for the later filtering mechanism

[2] `https://github.com/vacancy/SceneGraphParser`

by analyzing part-of-speech tagging from the topic, as utilized in MyScéal [39]. All words from the query were pre-processed to exclude stopwords and then lemmatized.

6.2.2 Lifelog Image Retrieval

After building the graph structure for both images and a query, the retrieval problem became how to calculate the similarity between the semantic graph of the topic with the set of scene graphs of lifelog images. This section describes how I embedded the graphs and then calculated their similarity score.

6.2.2.1 Graph Embedding

Recall that a scene graph of an image $\mathcal{G}_{mst}$ had a set of detected objects $O = \{o_1, ..., o_m\}$ and the expanded relation set $R = \{r1, ..., r_k\}$ gained after the spanning process. I represented the graph in two embedded matrices: M_O^I and M_R^I describing the objects and relations information accordingly. $M_O^I \in \mathbb{R}^{m \times d}$ created with each row was the d-dimension feature vector of a label name of the corresponding object encoded by the Word2Vec [73] model ($d = 300$). Similarly, $M_R^I \in \mathbb{R}^{k \times 3d}$ could be obtained with each row, which was a concatenated embedded vector of a subject, predicate, and an object in the relation. With the same method, a description query could also be encoded into M_O^Q and M_R^Q.

6.2.2.2 Similarity Score

My similarity function was adopted from SGM work [53] in which I matched the object and relation matrices of a text to those of an image, respectively. Regarding the object matrices, I took the score of the most relevant object in the M_O^I with each of the objects in the M_O^Q. After that, I got the average for the object's similarity score S_O. Assuming there were n_O^I and n_O^Q objects in an image and a query, the S_O could be calculated as:

$$S_O = \frac{1}{n_O^Q} \sum_{t=1}^{n_O^Q} MaxRow[M_O^Q * Transpose(M_O^I)] \tag{6.1}$$

where $*$ was the normal matrix multiplication, $MaxRow(X)$ was the function to calculate the highest value of each row of a matrix X, and $Transpose(X)$ was the matrix operation to find X^T. Likewise, suppose that there were n_R^I and n_R^Q relations detected in an image and a query, the relation similarity scores S_R could be measured as follows:

$$S_R = \frac{1}{n_R^Q} \sum_{t=1}^{n_R^Q} MaxRow[M_R^Q * Transpose(M_R^I)] \tag{6.2}$$

Finally, the similarity score between two graphs, S, was defined as $S = \alpha * S_O + \beta * S_R$ where α, β were obtained from empirical experimentation.

6.2.3 Experiment

6.2.3.1 Dataset

The lifelog data I used is the official data provided by the LSC'19 comparative benchmarking challenges [94]. This dataset consists of multimodal lifelog data captured by a single lifelogger wearing a small camera. The camera passively captured images at a resolution of 1024 x 768 pixels every 30 seconds over a span of 27 days, resulting in a collection of over 40,000 images. All identifiable information in the dataset was removed by blurring faces and readable textual content. The data also came with biometric data (heart rate, galvanic skin response, etc.), physical activities (standing, walking, etc.), and GPS location along with its timestamps. Nevertheless, I only used the visual data with its location and date for this work, the remaining factors were not taken into account because this pilot study focused on applying the scene-graph structure to capture relations in the lifelog images and queries.

Regarding the queries in the experiment, there were 20 queries, of which ten of them were randomly chosen in the official queries used in the LSC'19 and denoted as the LSC type. The remaining ten queries were manually created that focused more on the interactions between objects and were denoted as the **Descriptive Interaction Topics** (DIT) type. I used the DIT queries to assess how different

the proposed graph-based retrieval system can improve compared to the concept-based conventional system when the relations between objects come into account. An example of each query type is:

- (LSC) Eating fishcakes, bread, and salad after preparing my presentation in PowerPoint. It must have been lunchtime. There was a guy in a blue sweater. I think there were phones on the table. After lunch, I made a coffee.
- (DIT) I was eating a pizza. My **hand** was **holding** a **pizza**. There was a **guy wearing** a pink **shirt** talking to me. There was a black **box on** a **table**. It was on Friday morning. It happened at my workplace

Both query types contained information about when and where a moment happened, but the DIT had a more direct association between objects (as in bold in the above example). The answers to those queries can be illustrated in Figure 6.3. The detail of these queries is discussed in Appendix A.2. In my experimental analysis, I reported separately on the results using both types of queries.



(a) Sample result for query LSC31



(b) Sample result for query DIT02

Figure 6.3: Sample results for the example queries of the two types of queries

6.2.3.2 Evaluation Metric

In this experiment, I wanted to verify whether capturing interactions between objects could improve the accuracy of the retrieval. As discussed in Section 3.3.3, I used the Mean Reciprocal Rank (MRR) on the top 100 images found by the systems' users

within the experimental timeframes. I chose this metric because it was sensitive to the ranking position, which was also the main criterion in my assessment in this experiment to evaluate the retrieval system's accuracy. I did not consider the retrieval time in the experiment.

6.2.3.3 Experiment Design

Regarding baseline, I used the version of MyScéal which participated in the ImageCLEF2020-Lifelog challenge. The reason why I chose this version instead of the original version in LSC'20 is that I did not take the temporal retrieval issue into account. This was because I only focused on the accuracy of the retrieval models solely and did not take the user experience into account in this pilot study. The baseline utilized a standard design of a typical lifelog retrieval system that facilitated textual queries and generated a ranked list by utilizing an aTF-IDF scoring function. Both baseline and proposed methods were configured to return the top 100 images matching a given query and show them to searchers. The users could revise their inquiries until they thought the answers appeared at the top of the retrieved list. This was because the evaluation metric was MRR which would be higher if the answer was ranked at the top place. The users were guided on each system using several sample queries before the official experiment. There was a time limit of five minutes for a volunteer to solve a single query in each system. They were also told that the evaluation in the experiment was MRR so they needed to utilize these five minutes to modify their input in order to make the relevant images appear in the highest position. After five minutes or when the users were happy with their result, the current retrieved ranked list was recorded for evaluation.

The experiment involved 20 queries divided into four smaller runs (A, B, C, and D), with each run containing five topics from both the LSC and DIT types. Eight volunteers, all with backgrounds in computer science ranging from BSc to Ph.D., participated in the study. Each volunteer performed two runs, one for each lifelog

retrieval system. Notably, some participants already had prior knowledge of the lifelog retrieval task. A single volunteer would use one system to do five queries and then use the other system to find the answers to another five queries. To avoid any potential learning bias between the first and second runs, I designed the experiment according to Table 6.1. With this configuration, it was ensured that each setting's pair would be performed twice with different orders of the systems used to do the retrieval. For example, a couple of A-B experiment was done two times with User 1 and User 7 in a distinct context. While User 1 did run A with the baseline first followed by run B with the proposed system, User 7 used the proposed system for run A before doing run B with the baseline. This configuration allowed the entire query set to be executed twice on each system.

Table 6.1: The assignment of query subsets and systems for each user in our experiment in which A, B, C, and D are our four runs (groups of five topics).

	Baseline	**Proposed**		**Proposed**	**Baseline**
User 1	A	B	**User 5**	C	D
User 2	B	C	**User 6**	D	A
User 3	C	D	**User 7**	A	B
User 4	D	A	**User 8**	B	C

6.2.4 Results and Discussion

Table 6.2 shows the MRR scores of two approaches. The graph-based method achieved a higher result (MRR of 0.28) compared to 0.15 for the concept-based system by considering all queries. By examining specific query types, the scene-graph technique also obtained a better score for each type. Regarding the DIT type, the proposed method surpassed the baseline when the MRR score of each approach was 0.41 and 0.2, respectively. The proposed system also got a higher MRR score of 0.15 compared to that of the baseline with 0.1. It can be seen that both methods performed better on DIT queries than on LSC queries. This may be because the DIT queries described the lifelog events in more detail than those in LSC as they had more objects and interaction information in the queries.

However, there was only a minuscule change in the MRR scores of the baseline when the MRR for DIT queries was 0.1 higher than that of LSC queries. In contrast, the graph-based retrieval engine witnessed an increase in the scores between two types of topics since this technique could capture the relationships between objects in the query and images, which was the critical point in the DIT set. The MRR of this graph-based system on DIT was nearly three times higher than LSC, which were 0.41 and 0.15 accordingly. The result of this pilot experiment showed the effectiveness of applying graphs to the lifelog retrieval task in capturing relations information. Nevertheless, this experiment was only a preliminary study and did not focus on other factors such as the retrieval time and the interactive mechanism used in the LSC series. This was because the goal of this pilot study was to evaluate the accuracy of the graph-based retrieval model in the lifelog retrieval task. Hence, my final experiment should follow the format of the LSC challenge where I will use the LSC scoring function to evaluate systems and apply the revealing query mechanism of the challenge.

Table 6.2: Mean Reciprocal Rank scores of two systems on each type of query and entire dataset.

	LSC	**DIT**	**Entire**
Baseline	0.1087	0.2027	0.1557
Proposed	**0.1548**	**0.4166**	**0.2857**

6.3 An Empirical Comparison of Performance of HADA and Other Approaches in Lifelog Retrieval

The first experiment I described above shows the benefit of using interaction information with a graph-based structure in the lifelog retrieval task. The next experiment is about applying HADA to this task[3]. Although HADA obtains a SOTA result in the image-text retrieval task, its performance in lifelog retrieval

[3]Source code is available at `https://github.com/m2man/Comparing-models-in-Lifelog-Retrieval-Task.git`

remains unknown. In this section, I assess it by conducting experiments in two ways including the automatic (non-interactive) manner and the interactive manner. Regarding automatic manner, I solely utilize many lifelog search engines which consist of the concept-based method and other SOTA embedding-based models including HADA to find answers for LSC'21 queries without using their interactive systems, for example, there will be no users or no interactive operations. In contrast, I integrate HADA and the concept-based search engine used in MyScéal 2.0, which is described in Section 4.4, in two interactive lifelog retrieval systems having the same user interfaces. This setting guarantees that the UI will not make an impact on the comparison.

6.3.1 HADA Modification

As I have discussed in Section 5.3, HADA is the framework designed for combining any embedding models using a graph neural network. In this experiment, I made a slight change in HADA instead of using its original configuration. The original HADA measured the similarity of an image I and a text T using the output of a single input model. Specifically, the similarity function used in HADA to combine CLIP and BLIP was as follows:

$$S(I,T) = (1-\alpha)\langle h_I, h_T\rangle + \alpha\langle h_I^{(B)}, h_T^{(B)}\rangle \tag{6.3}$$

where α was a trainable weight, $\langle\,,\rangle$ was the dot product, h_I, h_T were the encoded vector of the image I and the text T from HADA while $h_I^{(B)}$, $h_T^{(B)}$ were the original encoded features from BLIP. The reason for employing only BLIP instead of both input models in the similarity function was that BLIP obtained a much better result than CLIP on the Flickr30K and MSCOCO datasets used in their experiment. Because BLIP was fine-tuned on these image-text retrieval datasets while CLIP was the zero-shot model, using CLIP when measuring the similarity score might lower the performance. However, BLIP and CLIP are both zero-shot

models in lifelog data. Thus, I modified the function to:

$$S(I,T) = \frac{\alpha\langle h_I, h_T\rangle + \beta\langle h_I^{(B)}, h_T^{(B)}\rangle + \gamma\langle h_I^{(C)}, h_T^{(C)}\rangle}{\alpha+\beta+\gamma} \tag{6.4}$$

where α, β, γ were trainable weights, $h_I^{(C)}$ and $h_T^{(C)}$ were the original encoded features from CLIP. With this update, I expected that using original features from all input models would help HADA to perform better than using only one in the zero-shot retrieval setting.

6.3.2 Datasets

6.3.2.1 Pre-training Dataset

Because I modified HADA and introduce new trainable weights, I needed to pre-train this graph-based model. Following the pre-trained BLIP on the Flickr30K dataset, I also pre-trained HADA on this dataset. Flickr30K is the popular dataset for training and evaluating image-text retrieval models and comprises 31,000 images collected from the Flickr website leading to more than 150,000 pairs of image-text. I used the zero-shot CLIP model to be an input model for HADA in this pre-training stage. It is because training CLIP on the Flickr30K dataset was a challenge for us due to resource constraints. However, I could train HADA due to this model's compactness with a few training parameters.

6.3.2.2 Fine-tuning Dataset

Following LifelogCLIP's pipeline [216], which fine-tuned CLIP on a lifelog dataset, I tried fine-tuning HADA on the lifelog LLQA dataset. LLQA was a manually annotated lifelog dataset collected from LSC'20 in which lifelog images were grouped into events based on the locations and content they captured. Each event was manually annotated with a description that could be longer than one sentence, as depicted in Figure 6.4. I adopted a similar data split from LifelogCLIP to train and validate HADA using the LLQA dataset. There were 11,982 pairs of image-text for

training and 1,234 pairs for validating. I used this split to fine-tune the HADA that is already pre-trained on Flickr30K beforehand.



The lifelogger is working at the cafeteria. He is walking back to the office while holding a cup of coffee by hands.

Figure 6.4: A sample in the LLQA dataset with the description of an event

6.3.2.3 Evaluation Dataset

My experiments were inspired by the LSC format, which meant that a retrieval model only needed to find one correct lifelog image relevant to a semantic query. I employed the official dataset of the LSC'21 challenge as my main evaluation dataset. There were 23 official queries and five testing queries in this query bank[4]. Each query contained six hints that are gradually revealed to the searchers, one hint every 30 seconds, each of which made the query easier to solve. A hint usually provided an additional visual description of the image or information such as the time or location of the required content. The list of these lifelog queries is described in Appendix A.2.

The visual dataset of LSC'21 was a four-month multimodal dataset that consisted of 183,299 lifelog images. The images were captured by a wearable camera and had a resolution of 1024×768. All LSC'21 images were anonymized and redacted, by blurring faces and removing user-identifiable data from within the images. Along with visual images, the dataset also came with metadata including

[4] `http://lifelogsearch.org/lsc/resources/lsc21-topics-qrels-shared.txt`

location information (GPS), physical activity annotation, and detailed timestamps for all content.

Regarding the experiment in an automatic manner, I used 23 official queries for the comparison and discarded the testing queries. I followed the pipeline of other works about comparing lifelog retrieval systems [178, 216]. Therefore, I only used visual-related information in the queries and removed non-visual clues about time and location. Instead of using all six hints as the original LSC'21 queries, I filtered and use the first three hints of a query since most of the later hints are about time and location.

Unlike the automatic manner, the experiment in an interactive manner used 24 queries for the evaluation, including 23 official queries and one testing query. The reason is that there were eight users in our user study; hence 24 queries would be easier for the configuration. Additionally, the four remaining testing queries were used to train users to know how to operate a lifelog retrieval system.

6.3.3 Non-Interactive Experiment

6.3.3.1 Exeperiment Design

The non-interactive experiment aimed to compare the accuracy of many lifelog retrieval models without using any interfaces. There were eight methods in this comparison which included the concept-based backend search engine of MyScéal 2.0 and seven embedding-based models containing CLIP, BLIP, HADA, and their variant versions. This experiment consisted of different versions of CLIP that had been applied in several lifelog retrieval systems. One version was the search engine of E-MyScéal [99], which won the LSC'22 challenge. It implemented the basic version of CLIP when it used the ViT-L/14 structure. LifelogCLIP [216], a ViT-B/16-based CLIP fine-tuned specially for a lifelog retrieval task, was also added to the comparison. Additionally, the experiment included the best option of CLIP with ViT-L/14-336 configuration[5]. Although BLIP obtained better results

[5]CLIP had not released better options at the time I was experimenting

than CLIP in image-text retrieval tasks on Flickr30K and MSCOCO, this new model had yet to be applied to lifelog retrieval. Therefore, I included BLIP in my experiment to validate whether BLIP outperforms CLIP in the lifelog search task. Last but not least, I evaluated HADA which fused CLIP and BLIP Additionally, HADA was our main focus in this research and I used HADA to fuse CLIP and BLIP with the assumption of improving their accuracy. It is worth noting that BLIP and HADA used in our experiments were priorly pre-trained on the Flickr30K dataset. Each model in the comparison received the same pre-processed LSC'21 queries and returned the ranked list of relevant images which was then used to compute the evaluation metrics.

6.3.3.2 Evaluation Metrics

I followed previous research [178, 216, 217] and used the same evaluation metric called R@K, which has been defined in Section 3.3.2. For a given query, R@K is defined as whether a lifelog retrieval model can find at least one correct lifelog image in the top-K retrieved images of the ranked list. I selected $K \in [1, 3, 5, 10, 20, 50, 100]$ for consistency with the LifelogCLIP setting. Smaller values of $K \in [1, 3, 5, 10]$ are popular in an ordinary image-text retrieval task to evaluate the performance of retrieval models [51, 53, 119, 135, 136, 184, 218]. However, bigger values of $K \in [20, 50, 100]$ are also crucial in lifelog retrieval tasks because most lifelog retrieval systems operate in an interactive manner. Although the search engine may find the answer at a lower rank, searchers who use the system can easily see that image by glancing through the list of pictures shown on the system's user interface.

I also used some average metrics to facilitate an easier overview. Table 6.3 showed six additional metrics, including AR, ARS, ARB, MAR, MARS, and MARB. Particularly, **AR** is an average R@K at each number of hints. **ARS** and **ARB** are the average R@K for small values ($K \leq 10$) and big values ($K > 10$) of K, accordingly. Similarly, the three remaining metrics **MAR**, **MARS**, and **MARB** are the mean of AR, ARS, and ARB, respectively.

6.3.3.3 Results and Discussion

Table 6.3 illustrates the R@K of several search engines solving 23 queries of LSC'21 with three hints. Only MyScéal 2.0 followed the concept-based method, while others used embedding models as their search engines. It is important to note that the statistics of E-MyScéal were obtained from its research paper [99]. HADA-1 was the HADA model that was pre-trained on Flickr30K using the similarity function in Equation (6.3), while HADA-2 utilized Equation (6.4) and HADA-2-FT was fine-tuned on LLQA dataset.

Table 6.3 shows that the concept-based retrieval model used in MyScéal 2.0 obtained the lowest score in the experiment. This basic search engine could not find the correct lifelog images in the top-10 results with the first hint. As more hints were given, MyScéal 2.0 improved the accuracy. Nevertheless, there was no big improvement between using two and three hints in the concept-based model since the difference in AR was only 1.2%. It might be because the third hint did not give more information about the visual concepts but more about the context of the images. The same was not true in other embedding-based approaches because they did make a considerable increase in the AR when they used more hints. Significantly, HADA-1, HADA-2, and HADA-2-FT improved at least 10% in AR for each additional hint released. E-MyScéal obtained the highest average R@K at the first hint with AR = 0.4471, while the zero-shot HADA-2 got the best scores at the two remaining hints with AR = 0.5342 and AR = 0.677, respectively.

In general, CLIP-L14-336, the best structure among all CLIP versions, had a similar performance to E-MyScéal using the ordinary CLIP-L14 structure since their MAR scores were 0.47 and 0.4752, accordingly. Although BLIP outperformed CLIP on Flickr30K and MSCOCO datasets by a large margin [136], BLIP obtained slightly better results than CLIP-L14-336 on LSC'21 queries with 2.48% higher in the MAR metric. Combining these two SOTA models helped to boost the average R@K up to 5% when HADA-2 achieved the highest MAR at 0.5445. This meant that HADA-

2, which was pre-trained with function (6.4) performed better than HADA-1 using the function (6.3). Interestingly, fine-tuning embedding models on LLQA did not significantly increase the H@K compared to the zero-shot models. In specific, the MAR of the HADA-2-FT model was 0.5279, which was slightly lower than that of the HADA-2 model at 0.5445. The same was true with LifelogCLIP. Although fine-tuned on LLQA, this search engine obtained a lower MAR than other CLIP-based models, including E-MyScéal and CLIP-L14-336.

Regarding small values of $K \leq 10$, E-MyScéal got the highest ARS metric when using one hint. Meanwhile, HADA-2 obtained the best ARS metrics for using two and three hints leading to the highest MARS score at 0.471. It meant that HADA-2 could find the correct images at the top of the ranked list. However, HADA-2-FT performed better than others when the bigger values of $K > 10$ were taken into account and managed to get the highest MARB score at 0.6666. Overall, embedding-based search engines surpassed the traditional concept-based models in the automatic experiment. Additionally, the HADA-2 model performed best among all embedding models with the highest MAR. Since the embedding models outperformed the concept-based model by a large margin, it showed the opportunity to apply this new generation of search engines to interactive lifelog retrieval systems. Among embedding models, HADA-2 solved more queries on average. Thus, I will use it to compare with the traditional concept-based MyScéal 2.0 in my interactive experiment.

6.3.4 Interactive Experiment

I carried out a user study experiment to have a detailed comparison in an interactive manner, which is considered the most appropriate mode in the lifelog retrieval task [1]. I used MyScéal 2.0 as the concept-based approach and HADA-2 as the graph-based approach since they are the two main targets in my hypothesis. It is important to remark that although the search engine of MyScéal 2.0 performed worst in a non-interactive manner, MyScéal 2.0 was a winner of the

Table 6.3: Performance of embedding-based and concept-based search engines on 23 queries of LSC'21. The symbol † indicates that the results are originally reported in their research. The symbol ★ depicts that the model is fine-tuned on the LLQA dataset. The superscripts ^{1,2,3} depict the best values in each evaluation metric for hints 1, 2, and 3, accordingly.

Model	Hint	R@1	R@3	R@5	R@10	R@20	R@50	R@100	AR	ARS	ARB	MAR	MARS	MARB
MyScéal 2.0	1	0	0	0	0	0.0435	0.0435	0.0435	0.0186	0	0.0435	0.0663	0.0363	0.1063
	2	0.0435	0.0435	0.0435	0.087	0.087	0.087	0.1304	0.0746	0.0544	0.1015			
	3	0.0435	0.0435	0.0435	0.087	0.087	0.1304	0.1739	0.087	0.0544	0.1304			
E-MyScéal†	1	0.26	0.35	0.39^{1}	0.52^{1}	0.52^{1}	0.52	0.57	0.4471^{1}	0.38^{1}	0.5367	0.4752	0.4125	0.5589
	2	0.30	0.35	0.43	0.48	0.57	0.57	0.57	0.4671	0.39	0.57			
	3	0.35	0.43	0.52	0.57	0.57	0.57	0.57	0.5114	0.4675	0.57			
LifelogCLIP★	1	0.2608	0.3043	0.3043	0.3478	0.3478	0.5217	0.6522^{1}	0.3913	0.3043	0.5072	0.4472	0.3659	0.5555
	2	0.2173	0.3478	0.4348	0.4783	0.4783	0.5652	0.6522	0.4534	0.3696	0.5652			
	3	0.3043	0.4348	0.4783	0.4783	0.5217	0.6087	0.6522	0.4969	0.4239	0.5942			
CLIP-L14-336	1	0.3043^{1}	0.3478	0.3913^{1}	0.3913	0.5217^{1}	0.5652^{1}	0.5652	0.441	0.3587	0.5507	0.47	0.3913	0.5749
	2	0.2609	0.4348^{2}	0.4348	0.4348	0.4348	0.6087	0.6087	0.4596	0.3913	0.5507			
	3	0.2609	0.4783	0.4783	0.4783	0.5652	0.6522	0.6522	0.5093	0.424	0.6232			
BLIP	1	0.2609	0.3913^{1}	0.3913^{1}	0.3913	0.3913	0.4783	0.4783	0.3975	0.3587	0.4493	0.4948	0.4058	0.6135
	2	0.2609	0.3478	0.3913	0.4783	0.5217	0.6522	0.6957	0.4783	0.3696	0.6232			
	3	0.2609	0.4348	0.6087	0.6522	0.7391^{3}	0.7826^{3}	0.7826	0.6087	0.4892	0.7681			
HADA-1	1	0.2174	0.3478	0.3478	0.3478	0.3913	0.5217	0.6087	0.3975	0.3152	0.5072	0.5135	0.4275	0.628
	2	0.3043	0.3913	0.4348	0.5217^{2}	0.5217	0.6087	0.6522	0.4907	0.413	0.5942			
	3	0.4348	0.5217	0.6087	0.6522	0.7391^{3}	0.7826^{3}	0.8261	0.6522	0.5544	0.7826			
HADA-2	1	0.2609	0.3913^{1}	0.3913^{1}	0.3913	0.4783	0.5217	0.5217	0.4224	0.3587	0.5072	**0.5445**	**0.471**	0.6425
	2	0.3913^{2}	0.4348^{2}	0.5217^{2}	0.5217^{2}	0.6087	0.6087	0.6522	0.5342^{2}	0.4674^{2}	0.6232			
	3	0.4783^{3}	0.6087	0.6087	0.6522	0.7391^{3}	0.7826^{3}	0.8696^{3}	0.677^{3}	0.587^{3}	0.7971^{3}			
HADA-2-FT★	1	0.2174	0.3043	0.3478	0.3478	0.4348	0.5652^{1}	0.6522^{1}	0.4099	0.3043	0.5507^{1}	0.5279	0.4239	**0.6666**
	2	0.1739	0.3913	0.4348	0.5217^{2}	0.6522^{2}	0.7391^{2}	0.7391^{2}	0.5217	0.3804	0.7101^{2}			
	3	0.3478	0.6522^{3}	0.6522^{3}	0.6957^{3}	0.7391^{3}	0.7391	0.7391	0.6522	0.587^{3}	0.7391			

LSC'21 challenge for the interactive lifelog retrieval systems [180]. However, LSC'21 evaluated the participating systems' performance with their developers as users. Therefore, the performance was based not only on an internal search engine but also on the user who knew how to utilize the system appropriately and their knowledge about the lifelog data. My user study aimed to have an unbiased comparison between the concept-based approach and the graph-based HADA-2 model. Hence the user's background knowledge of the system and the lifelog data should not have an impact. I integrated these search engines into two interactive lifelog retrieval systems in this interactive study. However, they had the same interfaces but different backend search engines, meaning that UI would not affect the result.

6.3.4.1 Interactive Lifelog Retrieval Systems

As I have mentioned about removing any bias from the UI in the interactive manner comparison, I used two lifelog retrieval systems for MyScéal 2.0 and HADA-2. They both had similar UI and the same filtering process for time and location information in lifelog data. The systems I used in the user study had the UI adopted from the E-MyScéal that won LSC'22. E-MyScéal was also collaborative research of my colleague and me. The UI was developed to create a simple and efficient interface for novice users unfamiliar with the lifelog retrieval task [99]. Figure 6.5 shows the UI of the system I used in this interactive experiment. Similar to the UI of MyScéal 2.0, the UI of E-MyScéal consists of search bars on the top and the main panel showing the retrieved images in the middle. The relevant images are sorted from left to right and from the top to the bottom. There is a map at the right of the UI where users can perform location filtering based on hints from a query. Users can type directly into the search bar triggering filtering mechanisms, such as time of day or weekday.

Nevertheless, there were some updates in the new UI compared to the previous version in MyScéal 2.0. The first update involved modifying the triplet layout, as depicted in Figure 4.11a, so that it was not always visible to users conducting

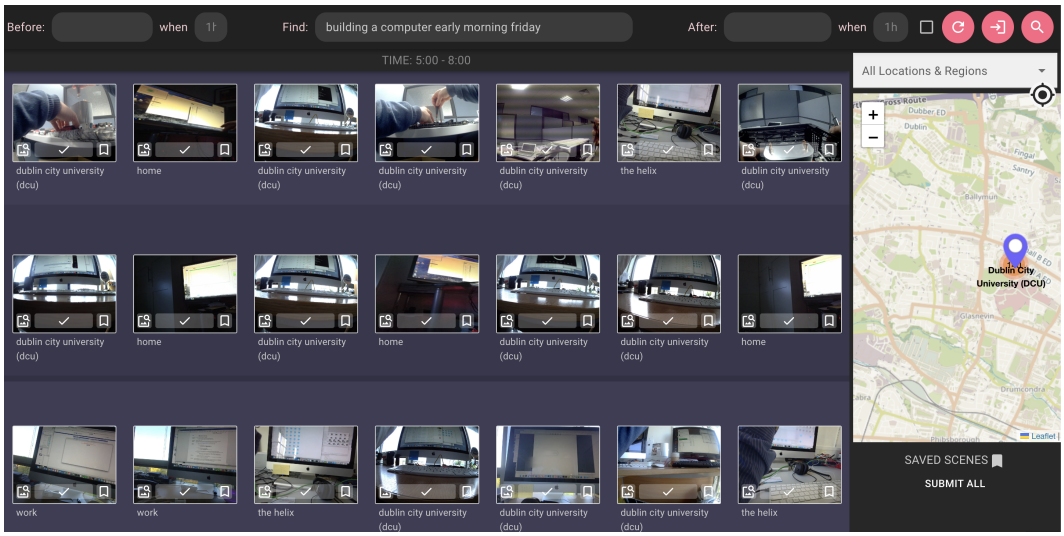


Figure 6.5: User interface of the lifelog retrieval system E-MyScéal used in the user study experiment

searches. This triplet layout illustrated the retrieved result in the temporal context describing a sequence of activities (scenes) of the lifelogger. However, it will not be meaningful if a given query does not contain temporal information. Furthermore, it can be harmful because it will reduce the number of relevant scenes shown in the main panel by displaying temporal scenes. Thus, there were several minor updates in the UI in which the main panel can be flexibly changed depending on the content of the query. The triplet layout only works if a user fill in the "*Before*" or "*After*" bars. Otherwise, only the main relevant scenes are visualized to users as shown in Figure 6.5. The second update was that the event view which is the third row illustrated in Figure 4.9 was removed and a new scene view was introduced as depicted in Figure 6.6. This new scene view will be popped-up whenever the user clicks on a retrieved scene. This pop-up window displays all scenes on the same day with the selected scene and they are sorted in temporal order. It also shows similar scenes and all images in the selected scene and can be navigated through scrolling. It is important to remind that two lifelog retrieval systems in the interactive experiment used this UI but employed different internal search engines which were the concept-based MyScéal 2.0 and the graph-based HADA-2.

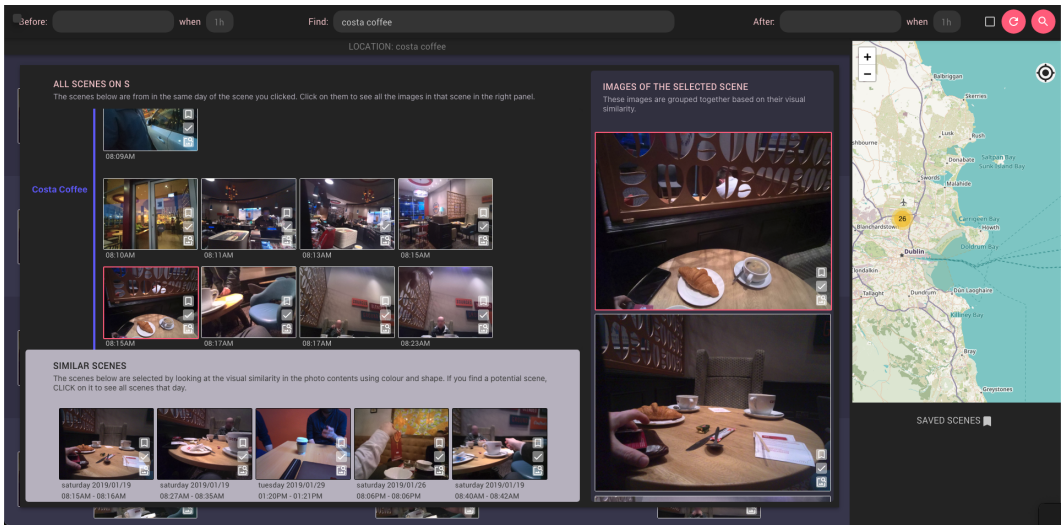


Figure 6.6: The scene view pop-up window of E-MyScéal

6.3.4.2 Evaluation Metrics

I followed LSC's format for the interactive benchmarking and employed its scoring function that was explained in Section 3.3.1.1. LSC evaluates systems based on two criteria: correctness and elapsed time to retrieval. In this user study, I set $T = 300$ seconds, meaning a user had five minutes to solve a query. Because each query contained six hints that were gradually revealed after every 30 seconds, a user would have 150 seconds to find the correct lifelog images after all hints appeared. This is also an official configuration of LSC for novice users [11]. Additionally, I also reported other metrics such as Precision, Recall, and F1 scores to have a deeper insight into the performance of both systems. The *Precision* is defined as the percentage of correct attempts among all their submissions, whilst the *Recall* is the percentage of queries that the user successfully found the correct answers.

6.3.4.3 Exeperiment Design

This experiment was designed to be similar to the pilot work which compared the performance of two interactive lifelog retrieval systems. There were 23 official queries and five testing queries in LSC'21 which were used in the interactive experiment. I randomly divided 24 queries (23 official and one testing) into four smaller batches,

namely A, B, C, and D. Each batch contained six queries. Eight volunteers were involved in the experiment, and each had to use the lifelog retrieval system to solve two batches (one for each search engine). A participant used the concept-based system to solve one batch, then used the graph-based system to solve another batch. All volunteers were at the bachelor's level, and none of them had prior knowledge of lifelog retrieval before participating in the experiment. This ensured that all volunteers were novice users, providing a fair basis for comparison.

Table 6.4 illustrates the user study design. As can be seen from the table, each pair of query batches was done twice but with different orders of the lifelog retrieval approaches used to do the retrieval. For example, User 1 and User 7 did the exact batches with corresponding models but in different orders. User 1 used MyScéal 2.0 first to solve batch A and then used HADA-2 to solve batch B. Meanwhile, User 7 also did the same pairs but used the graph-based system first for batch B and then the concept-based system for batch A. This design ensured no learning bias between the first and the second performance of the volunteers. The volunteers had a short time to learn how to use the systems and to try on the four remaining testing queries. All volunteers were novice users who had yet to learn about lifelog retrieval before experimenting.

Table 6.4: The assignment of query batches and search engines for each user in the interactive experiment in which A, B, C, and D are 6-query batches.

	MyScéal 2.0	**HADA-2**		**HADA-2**	**MyScéal 2.0**
User 1	A	B	**User 5**	D	C
User 2	B	C	**User 6**	A	D
User 3	C	D	**User 7**	B	A
User 4	D	A	**User 8**	C	B

6.3.4.4 Results and Discussion

Table 6.5 shows the performance of each user in our experiment. Column *Solved* indicates the number of queries for which a user successfully found the correct lifelog images. Column *WS* is the number of wrong submissions of the user during their

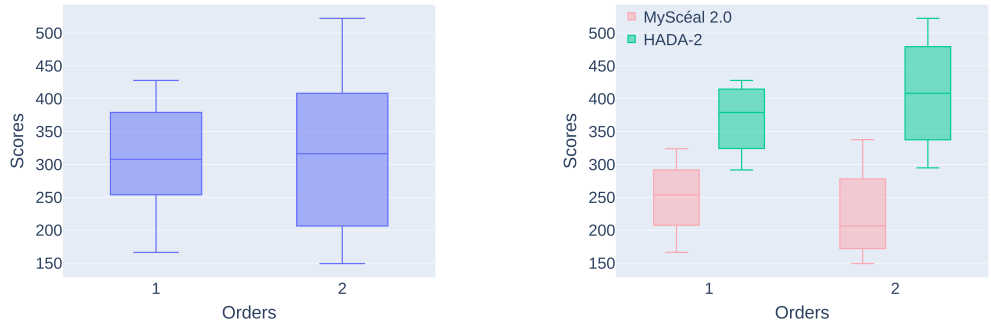
solving process. Column *Times* is the average time (in seconds) it took a volunteer to solve a query, and *Scores* is their total scores calculated by function described in Section 3.3.1.1. Regarding a query that the user could not find the answer to, we calculated their solve time for that query as 300 seconds which was the maximum allowed search time. Because a successfully solved query would award a user up to 100 points, the maximum point a volunteer could obtain for each query batch in the experiment was 600. In contrast, the volunteer got 0 points when they could not solve the query. Table 6.5 shows that all users solved more queries and obtained better scores with lower average searching time when they used HADA-2 to retrieve relevant lifelog images. Interestingly, User 1 using MyScéal 2.0 had a lower average retrieval time for batch A than User 6 using HADA-2. Nevertheless, User 6 obtained higher final scores than User 1 due to the higher number of queries that were solved. Regarding other statistics, participants using both systems had the same number of wrong submissions on average. However, users solved more queries with HADA-2 which made the graph-based system get higher Recall and F1 scores. On average, The F1 score of the graph-based system was 0.78 while that of the conventional system was 0.57.

Figure 6.7 depicts the distribution of points obtained by volunteers concerning performance orders. It shows no difference in the average score regarding the order of the system used in the experiment as shown in Figure 6.7a. The average score of the system used in the first run was 309.48, while that of the second run was slightly better at 316.68. Nevertheless, regarding the search engines, Figure 6.7b illustrates that volunteers using HADA-2 achieved higher points than those using the concept-based method regardless of the order of the system used.

Figure 6.8 illustrates the score gained using each system based on query batches. On average, volunteers who used HADA-2 had 388.94 points, surpassing MyScéal 2.0 at 237.23 points. Regarding query batches, the lifelog retrieval system with HADA-2 as a search engine performed better than MyScéal 2.0 on all batches. In summary, the graph-based HADA-2 outperformed the concept-based MyScéal 2.0 where the

Table 6.5: Performance of each retrieval model on different query batches. WS is short of Wrong Submission. The bold values indicate the best results for that evaluation metric in each row.

Query Batch	Order	MyScéal 2.0								HADA-2							
		User	Solved	WS	Precision	Recall	F1↑	Times↓	Scores↑	User	Solved	WS	Precision	Recall	F1↑	Times↓	Scores↑
A	1	User 1	3	2	0.6	0.5	0.5454	**201.16**	248.83	User 6	5	4	0.5556	0.8333	**0.6667**	229.33	**291.66**
	2	User 7	2	8	0.2	0.3333	0.25	250.83	149.16	User 4	4	2	0.6667	0.6667	**0.6667**	**195.16**	**294.83**
B	1	User 2	2	4	0.3333	0.3333	0.3333	223.83	166.16	User 7	6	1	0.8571	1	**0.9231**	**162.16**	**427.83**
	2	User 8	5	1	0.8333	0.8333	0.8333	202.16	337.83	User 1	6	1	0.8571	1	**0.9231**	**153.83**	**436.16**
C	1	User 3	4	1	0.8	0.6667	**0.7272**	166	324	User 8	5	4	0.5556	0.8333	0.6667	**138.66**	**401.33**
	2	User 5	3	2	0.6	0.5	0.5454	232	218	User 2	5	5	0.5	0.8333	**0.625**	**140.66**	**380.33**
D	1	User 4	4	1	0.8	0.6667	0.7272	231	259	User 5	5	2	0.7143	0.8333	**0.7692**	**183**	**357**
	2	User 6	3	0	1	0.5	0.6667	255.16	194.83	User 3	6	0	1	1	**1**	**77.66**	**522.33**
Average	1	-	3.25	2	0.6333	0.5417	0.5833	205.5	249.5	-	5.25	2.75	0.6706	0.875	**0.7564**	**178.29**	**369.46**
	2	-	3.25	2.75	0.6583	0.5417	0.5739	235.04	224.96	-	5.25	2	0.7559	0.875	**0.8037**	**141.83**	**406.42**
	All	-	3.25	2.375	0.6458	0.5417	0.5786	220.27	237.23	-	5.25	2.375	0.7133	0.8749	**0.78**	**160.06**	**388.94**



(a) (b)

Figure 6.7: The distribution of scores based on the order of performance solely (a) and with the systems (b)

former model not only obtained faster searching time but also achieved a higher number of successfully solved queries than the latter model in the user study.

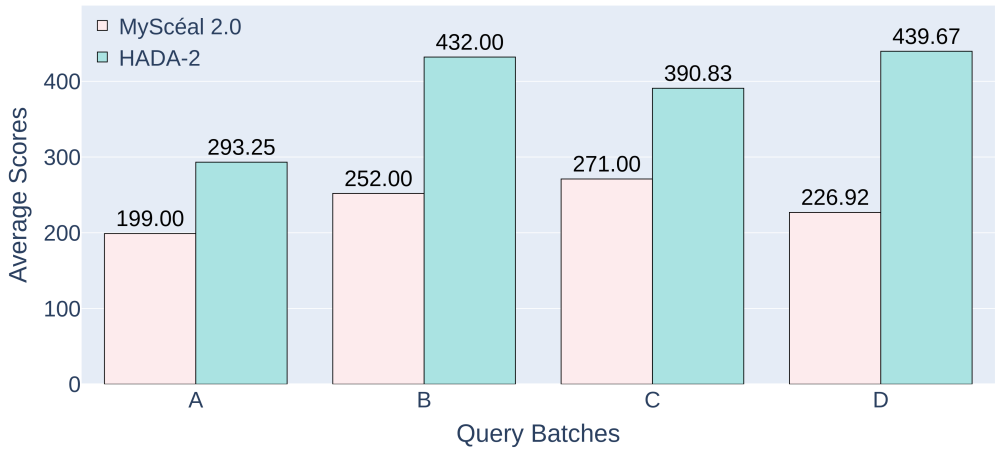


Figure 6.8: The average score of users gained on each batch based on the system that they used

6.4 Conclusion

The pilot study in which I applied scene-graph structure to a lifelog retrieval system and compared it with the concept-based MyScéal showed that capturing the interaction information could boost the accuracy of the system with higher MRR scores. The graph-based system obtained only better performance on

ordinary LSC queries but also on DIT queries focusing on the interaction between objects. However, this preliminary experiment evaluated systems solely based on the MRR metric, which was used to assess the accuracy, and underlooked the retrieval time criteria as applied in the LSC challenge. Therefore, it was necessary to have a comprehensive experiment in which the evaluation metric should take time and accuracy into account. I adopted LSC's benchmarking format and its scoring function to conduct the last user experiment where eight novice users used both the graph-based HADA system and the concept-based MyScéal 2.0 system to solve several LSC'21 queries. The result showed that the graph-based approach outperformed the concept-based method in both accuracy and retrieval time, making HADA obtain a higher overall score than MyScéal 2.0. Additionally, the automatic experiment also proved that HADA surpassed MyScéal 2.0 by a large margin in the Recall metrics. These results of the experiments gave the answer for my last RQ3 which is "*To what extent can a graph-based interactive lifelog retrieval system improve retrieval performance compared to the conventional concept-based system?*".

6.5 Chapter Summary

In this chapter, I investigate how to apply a graph-based search engine to a lifelog retrieval system and conduct experiments to compare the performance of the graph-based approach and that of the conventional concept-based method in lifelog retrieval. I carry out two distinct experiments in which the first study is about using scene-graph structure to capture interactions between objects and the second study is about employing the HADA framework to address the lifelog retrieval task. I carefully design our experiments to ensure that participants can use both systems using these search engines without having the learning bias factor. In the scene-graph study, I convert each image and query into matrices representing their extracted scene graphs. These matrices can be compared to each other to calculate

the similarity score between the query and the image. The result of this pilot experiment shows that the lifelog retrieval system using the scene-graph structure can improve the accuracy of retrieved results by obtaining better MRR metrics in not only the DIT queries but also in ordinary LSC queries. Regarding the experiment of applying HADA to lifelog retrieval, I use HADA to combine CLIP and BLIP. I then evaluate HADA performance in this retrieval task in two different manners: automatic (non-interactive) and interactive experiments. The results prove that the graph-based approach outperforms other SOTA methods including the traditional concept-based model in both evaluation criteria which are accuracy and search time. In conclusion, the finding of this chapter helps me to answer RQ3 and to prove the main hypothesis of the dissertation which is *"Applying graph-based structure into an interactive lifelog retrieval system can help to increase the performance in terms of speed and accuracy compared to conventional concept-based interactive retrieval systems"*.

Chapter 7

Conclusion

In this thesis, my hypothesis is applying a graph-based search engine can enhance the performance of an interactive lifelog retrieval system compared to the traditional concept-based search engine. I proposed three research questions to validate the hypothesis. My comprehensive experiments and evaluations help us to find the answers to these primary questions. Through that, I can prove the correctness of my hypothesis. This final chapter will briefly summarize how I addressed each research question to get their answers and conclude the hypothesis. Along with it, the research contributions are highlighted and my publications related to each research question are also listed. Furthermore, some limitations of this research and future works will be discussed in this chapter.

7.1 Proving the Hypothesis

My thesis focuses on proving the hypothesis which is *"Applying a graph-based structure to an interactive lifelog retrieval system can help to increase performance in terms of speed and accuracy compared to conventional concept-based interactive retrieval systems"*. Three research questions have been proposed to concrete this hypothesis.

7.1.1 Research Question 1

The first research question is "*What is the state-of-the-art of the interactive lifelog retrieval systems following the conventional concept-based approach?*". This

question aims to find the appropriate concept-based lifelog retrieval system to be the baseline in the comparison of the hypothesis. The baseline needs to have not only a friendly user interface but also an accurate retrieval model as an internal search engine. Therefore, I and my colleague developed MyScéal and its upgraded version MyScéal 2.0 to address the first research question, which was described in Chapter 4. MyScéal is the interactive lifelog retrieval system that uses the search engine following the concept-based approach. The system employed the unique aTF-IDF similarity function to match concepts in a semantic query with annotated concepts of lifelog images, which were indexed by using the ElasticSearch database, to perform the retrieval and return the ranked list of relevant images. Furthermore, the system has an interface with a straightforward design that is expected to facilitate retrieval for novice users. Instead of using faceted filtering components on the UI, we integrate them into search bars allowing the free-text search where a user can fill in any concepts or date and time information. Additionally, this concept-based system contains a geolocation map panel which is distinct from other lifelog retrieval systems. The MyScéal system participates in several well-known lifelog retrieval competitions including ImageCLEF-Lifelog, NTCIR-Lifelog, and LSC series. The top performance of MyScéal in these challenges has proved that this system can be considered a SOTA interactive lifelog retrieval system and answers the first research question positively.

7.1.2 Research Question 2

While doing a literature review during solving the first question, I realize that there has been just a few research that applied graph-based structures to the lifelog retrieval task. Although LifeConcept and LifeGraph were the first two systems integrating this type of structure, their performance in the LSC series was still limited. Moreover, their implementations were not described in detail. Therefore, I decided to start developing a graph-based retrieval model in the field of image-text retrieval in which there have been many SOTA models using graph-based

structures. This leads to the second research question which is "*Can using graph-based structures improve the retrieval accuracy of current state-of-the-art cross-modal retrieval models in the image-text retrieval task??*". Chapter 5 described how I address this question. The SOTA cross-modal retrieval model at that time was the SGM model which employed scene graphs to capture relations between concepts in images. Each image and query were represented in graph structures and the SGM model used the local information, such as nodes or edges, of these graphs to calculate their similarity score. This method underlooked the global information of the entire graph. I extend the SGM model and address this limitation to introduce my LGSGM model by using additional graph neural networks to embed the whole graphs into vectors for comparison. Additionally, I replace the GRU architecture in the original SGM with the LSTM network in LGSGM. My updates help LGSGM surpass the SGM by 10% in the experiment on the popular Flickr30K dataset which makes LGSGM to be a new graph-based SOTA in this field. Nevertheless, the CLIP model which was introduced in late 2021 has marked a milestone in many vision-and-language tasks including the image-text retrieval task. By being pre-trained on 400 million images with multiple GPUs, it outperformed all SOTA models including the LGSGM. CLIP has opened a new trend in pre-training on massive external datasets. This tendency has been adopted by many later SOTA retrieval models such as ALIGN, ALBEF, or BLIP. It raised the challenge of how to develop an image-text retrieval model better than theirs without extensive pre-training using many GPUs.

Therefore, I proposed HADA, a graph-based amalgamation framework designed to integrate multiple state-of-the-art (SOTA) models for enhanced performance. HADA uses a graph structure to connect output features derived from input SOTA models, such as CLIP and BLIP. This graph is then updated through a graph attention network to combine these features together and output a unified feature. This feature is concatenated with the original features from input models to obtain the final representative feature vector of an image or a sentence. Although HADA

consists of roughly 10 million trainable parameters hence it is easy to train with only one GPU, my experiments show that this off-the-shelf add-on can increase the performance up to 4% in terms of recall evaluation metrics on Flick30K and MSCOCO datasets. These SOTA results answer my second research question.

7.1.3 Research Question 3

After developing SOTA graph-based image-text retrieval models in Chapter 5, I apply them to the lifelog retrieval task and proposed the final research question which is "*To what extent can a graph-based interactive lifelog retrieval system improve retrieval performance compared to the conventional concept-based system?*". As discussed in Chapter 6, I integrate the graph-based models as backend search engines of an interactive lifelog retrieval system and compared it to the baseline MyScéal system introduced in Chapter 4. The first comparison is the pilot study in which I focus on the accuracy of the search engine using scene-graph structures to capture relations between concepts compared to the concept-based method used in MyScéal. The pilot study with eight volunteers is designed carefully so that there will be no learning bias affecting the experiment. The configuration is described in Section 6.2.3.3. This arrangement ensured that each batch of questions was solved by at least two participants, albeit in different sequences, thereby mitigating any learning bias. I create Descriptive Interaction Topics (DIT) lifelog queries, incorporating additional details regarding interactions between objects, to evaluate whether the graph-based model outperforms traditional models when considering object interactions. The result shows that the search engine using scene-graph structures performs slightly better than the concept-based system on ordinary LSC queries. Nevertheless, there is a big gap in their MRR metrics on the DIT queries when the graph-based model outperforms the concept-based model with scores at 0.4166 and 0.2027, accordingly. Although the pilot experiment partially solves the third research question, it does not consider other factors that might impact the results such as search time or the user

interfaces. My final experiment in Section 6.3 considers these aspects and is conducted in two manners: non-interactive (automatic) and interactive settings. The experiment aims to compare HADA as the graph-based search engine with other retrieval models including the concept-based approach MyScéal, BLIP, and CLIP in solving official LSC'21 queries. It is noted that the interactive manner experiment asks eight novice participants to solve queries using the two interactive lifelog retrieval systems having similar interfaces: one system utilizes MyScéal 2.0 as the backend search engine, and the remaining system uses HADA. While the automatic manner experiment employs the R@K as the evaluation metric, the interactive manner experiment follows LSC's setting and uses the LSC's scoring function. The result from the non-interactive manner shows that HADA surpasses not only MyScéal but also BLIP and CLIP in R@K. The result from the remaining manner experiment reveals that the graph-based search engine can reduce the search time by 27% and thus improve the score by 63%. The performance of HADA in the comprehensive experiment as well as of the scene-graph structures in the pilot study answers the third research question.

With all research questions addressed, I can validate my hypothesis and conclude that employing a graph-based search engine yields superior performance compared to conventional concept-based search engines, particularly in terms of retrieval time and accuracy.

7.2 Research Contributions

This dissertation introduces **five** main contributions which can be summarized as follows.

1. I and my colleague develope the SOTA interactive lifelog retrieval system MyScéal and its subsequent versions using a concept-based search engine. Chapter 4 describes how MyScéal operates and its top performance at the LSC series and the ImageCLEF-Lifelog challenge.

2. I apply graph neural networks to the SGM image-text retrieval model to achieve a new SOTA performance at that time. This research, which is discussed in Section 5.2, extends the SGM work by introducing the global information of a representative graph of an image or a text that helps to increase the performance up to 10% on the Flickr30K dataset.

3. I introduce HADA which was a graph-based amalgamation framework that can be trained with a single GPU due to its tiny number of trainable parameters. The experiments on Flickr30K and MSCOCO dataset, as discussed in Section 5.3, show that HADA can increase the evaluation metrics by roughly 4% compared to BLIP.

4. I conduct the pilot study in Section 6.2 which compares the performance of lifelog retrieval systems using the scene-graph structure to focus on relations between concepts and using the traditional concept-based search engine in MyScéal. The results reveal that the graph-based system achieves higher MRR scores compared to MyScéal in solving both LSC queries and DIT queries which contain more information about interaction between objects.

5. I conduct a comprehensive comparison between the concept-based search engine from MyScéal 2.0 with HADA, CLIP, and BLIP for the lifelog retrieval task. Section 6.3 explains the final experiment in detail. The experiment is designed in both automatic and interactive configurations in which user interface and prior knowledge of lifelog data are ensured not to affect the comparison. The result proves that HADA surpasses MyScéal 2.0 by a large margin in both settings.

7.3 Limitations

There are several limitations that remain in this Ph.D. research. Although MyScéal achieves the SOTA performance in many lifelog retrieval challenges, the system has

not utilized other lifelog data rather than visual images, time, and location information. The biometric data of a lifelogger may contain important information that can be analyzed for deeper insights into personal daily activities. Another limitation is that MyScéal is built with the target of finding a single relevant lifelog image as fast as possible, hence the system might not perform well in other tasks such as the ad-hoc task and the question-answering (QA) task. While the ad-hoc task is similar to the ImageCLEF-Lifelog task that requires a system to submit all images, the question-answering is the new task introduced since LSC'22 which needs a system to find the answer for the lifelog query, for example "*What was the number of the lifelogger's office door?*". The third limitation is the HADA structure. Currently, HADA employs the graph attention network to update a graph's features. This architecture is a baseline graph model and there are other networks that can be applied such as graph transformer [202]. Furthermore, the edge feature in the graph needs to be considered. HADA does not implement the edge feature in the experiment, but they can be learnable parameters in graph neural networks. Regarding user studies mentioned in this dissertation, eight participants are considered an appropriate setting. However, it can be extended with more users and the number of queries if I have more time to conduct the experiments. It is a challenge to get a volunteer because it takes roughly 2.5 hours for each user to finish their turn.

7.4 Future Works

My dissertation proposes that applying a graph-based search engine to a lifelog retrieval system can help the system improve its performance compared to using the traditional concept-based search engine. The experiments in three research questions have validated the hypothesis. There is still room for improvements in future works to address several limitations mentioned in Section 7.3. Specifically, I suggest the following research for further investigation:

7.4.1 Lifelog Retrieval System

Since the current MyScéal 2.0 system does not take biometric data and other personal information such as a music playlist of a lifelogger into account, using them may discover many analyses of the lifelogger's daily activity. For example, recent research can detect whether a person is stressed or not based on their collected biometric data [219]. It can be beneficial if there is a lifelog retrieval system integrating the stress moment filtering feature. The system can support a user to access any specific moment from their lifelog data that they find stressful or anxiety. Similarly, biometric data can also offer insight into the emotion of the lifelogger, for example when they feel happy or sad [220, 221]. It will open the possibility to apply the lifelog retrieval system to medical fields for health monitoring and treatments.

Another work that can be conducted in the future is to use personal information of the lifelog data such as names of songs or locations to develop a recommendation model which can be a component in the system. One potential feature is that the model will give a suggestion to the lifelogger about the list of recommended restaurants after searching through lifelog images about their dinner. The same is true for music playlists. The lifelogger can use the system to ask for the appropriate songs for a particular purpose like working or relaxing.

Regarding the question-answering issue, a solution can be modifying and applying HADA to a QA task since BLIP has been adopted in this domain and gained noticeable results. Additionally, it is necessary to refine the user interface of the lifelog retrieval system to be suitable for this task as well as the ad-hoc task.

7.4.2 Graph-based Image-Text Retrieval Model

Because the current version of HADA does not focus on the edges information of a graph, one promising direction for future investigation is to include the edges attribute in the HADA. The edge features may contain detail of how nodes transfer information between each other and they can be learnable along with node features

during the training phase. Moreover, other graph neural networks such as graph transformer [202] should be considered rather than using the baseline graph attention network.

In Section 5.3, I employ HADA to combine two models (CLIP - BLIP, and ALBEF - LightningDOT). Nevertheless, HADA can be used to fuse a larger number of models. Instead of two, future work might utilize HADA to fuse three or four models for further analysis although it might take a decent amount of time to run it in real-time. It is noted that the input models of HADA should come from different approaches in order to make HADA can learn the benefits of distinct models.

Most of the current image-text retrieval models were trained using the contrastive learning method in which the relevant pair of image-text gets a higher similarity score than the irrelevant pair. A recent research suggested that calculating the similarity score with a hyperbolic distance can improve the accuracy of a model [222]. The author of the research projected embedded features obtained from the Vision Transformer into the hyperbolic space instead of ordinary Euclidian space. Therefore, I think that this hyperbolic-based technique can enhance HADA performance without introducing a huge number of extra parameters.

7.5 Publication List

This section presents the list of my research papers published in several peer-reviewed conferences and journals while finding answers to three research questions. Additionally, other publications which are not directly related to this dissertation are also reported. The publications are sorted in chronological order.

7.5.1 Chapter 4 - RQ1

1. Ly-Duyen Tran, **Manh-Duy Nguyen**, Binh T Nguyen, and Liting Zhou. Myscéal: a deeper analysis of an interactive lifelog search engine. Multimedia

Tools and Applications, pages 1–18, 2023.

2. Ly-Duyen Tran, **Manh-Duy Nguyen**, Duc-Tien Dang-Nguyen, Silvan Heller, Florian Spiess, Jakub Lokoč, Ladislav Peška, Thao-Nhu Nguyen, Omar Shahbaz Khan, Aaron Duane, et al. Comparing interactive retrieval approaches at the lifelog search challenge 2021. IEEE Access, 11:30982–30995, 2023.

3. Ly-Duyen Tran, **Manh-Duy Nguyen**, Nguyen Thanh Binh, Hyowon Lee, and Cathal Gurrin. Myscéal 2.0: A revised experimental interactive lifelog retrieval system for lsc'21. In Proceedings of the 4th Annual on Lifelog Search Challenge, pages 11–16. 2021.

4. Ly-Duyen Tran, **Manh-Duy Nguyen**, Binh Thanh Nguyen, and Cathal Gurrin. An experiment in interactive retrieval for the lifelog moment retrieval task at imagecleflifelog2020. In CLEF2020 Working Notes, CEUR Workshop Proceedings, Thessaloniki, Greece, 2020.

5. Ly-Duyen Tran, **Manh-Duy Nguyen**, Nguyen Thanh Binh, Hyowon Lee, and Cathal Gurrin. Myscéal: An experimental interactive lifelog retrieval system for lsc'20. In Proceedings of the Third Annual Workshop on Lifelog Search Challenge, pages 23–28, 2020.

7.5.2 Chapter 5 - RQ2

7. **Manh-Duy Nguyen**, Binh T Nguyen, and Cathal Gurrin. HADA: A graph-based amalgamation framework in image-text retrieval. In Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part I, pages 717–731, 2023.

8. **Manh-Duy Nguyen**, Binh T Nguyen, and Cathal Gurrin. A deep local and global scene-graph matching for image-text retrieval. In New Trends in

Intelligent Software Methodologies, Tools and Techniques, SoMeT 2021, Cancun, Mexico, September 21–23, 2021, Proceedings, pages 510–523, 2021.

7.5.3 Chapter 6 - RQ3

8. **Manh-Duy Nguyen**, Binh T Nguyen, and Cathal Gurrin. Graph-based indexing and retrieval of lifelog data. In MultiMedia Modeling: 27th International Conference, MMM 2021, Prague, Czech Republic, June 22–24, 2021, Proceedings, Part II 27, pages 256–267. Springer, 2021.

7.5.4 Others

9. Chenyang Lyu, **Manh-Duy Nguyen**, Van-Tu Ninh, Liting Zhou, Cathal Gurrin, and Jennifer Foster. Dialogue-to-video retrieval. In Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part II, pages 493–501, 2023.

10. Van-Tu Ninh, **Manh-Duy Nguyen**, Sinéad Smyth, Minh-Triet Tran, Graham Healy, Binh T. Nguyen, and Cathal Gurrin. An Improved Subject-Independent Stress Detection Model Applied to Consumer-grade Wearable Devices. In: Fujita, H., Fournier-Viger, P., Ali, M., Wang, Y. (eds) Advances and Trends in Artificial Intelligence. Theory and Practices in Artificial Intelligence. IEA/AIE 2022. Lecture Notes in Computer Science(), vol 13343. Springer, Cham.

11. Ly-Duyen Tran, **Manh-Duy Nguyen**, Binh Nguyen, Hyowon Lee, Liting Zhou, and Cathal Gurrin. E-myscéal: Embedding-based interactive lifelog retrieval system for lsc'22. In Proceedings of the 5th Annual on Lifelog Search Challenge, pages 32–37. 2022.

12. Thao-Nhu Nguyen, Tu-Khiem Le, Van-Tu Ninh, Ly-Duyen Tran, **Manh-Duy Nguyen**, Minh-Triet Tran, Binh T Nguyen, Annalina Caputo, Cathal Gurrin, Graham Healy, et al. Dcu and hcmus at ntcir-16 lifelog-4. In Proceedings of

the 16th NTCIR Conference on Evaluation of Information Access Technologies. NTCIR, 2022.

13. **Manh-Duy Nguyen**, Thao-Nhu Nguyen, Thanh-Binh Nguyen, Caputo Annalina, and Cathal Gurrin. 2022. DCU Team at the NTCIR-16 RCIR Task. In Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies (NTCIR-16). Tokyo, Japan.

14. Van-Tu Ninh, Tu-Khiem Le, **Manh-Duy Nguyen**, Sinéad Smyth, Graham Healy, and Cathal Gurrin. A Preliminary Assessment of Game Event Detection in Emotional Mario Task at MediaEval 2021.

15. Ly-Duyen Tran, **Manh-Duy Nguyen**, Thao-Nhu Nguyen, Graham Healy, Caputo Annalina, Binh T. Nguyen, and Cathal Gurrin (2021) A VR interface for browsing visual spaces at VBS2021. In International conference on multimedia modeling. Springer, pp 490–495, 2021.

16. Tu-Khiem Le, **Manh-Duy Nguyen**, Ly-Duyen Tran, Van-Tu Ninh, Cathal Gurrin, and Graham Healy. Dcu team at the ntcir-15 micro-activity retrieval task. In Proceedings of the NTCIR-15 Conference, 2020.

Appendix A

List of Lifelog Queries Used in Lifelog Experiments

A.1 Novice User Experiment in Chapter 4

Queries used in the LSC'19[1], LSC'20[2], LSC'21[3], and ImageCLEF2020-Lifelog[4] can be accessed on the official websites of the challenges. Table A.1 shows the list of queries I used in the novice user experiment described in Section 4.4.1. The queries in this experiment were chosen from the query bank of LSC'20.

A.2 Experiments in Chapter 6

A.2.1 Pilot Study

Regarding the pilot study of assessing the performance of a lifelog retrieval system using scene-graph structures described in Section 6.2, I use 10 queries from LSC'19 (denoted as LSC queries) and create 10 queries that focus on interaction information (denoted as DIT queries). The queries are shown in Table A.2 along with the query batch that they belong to. The query batches are discussed in Section 6.2.3.3 and each batch contains five queries. It is noted that a query is shown at once, instead

[1]LSC'19 queries: `http://lifelogsearch.org/lsc/resources/lsc2019-topics.xml`

[2]LSC'20 queries: `http://lifelogsearch.org/lsc/2022/resources/lsc20-topics-qrels.txt`

[3]LSC'21 queries: `http://lifelogsearch.org/lsc/2022/resources/lsc21-topics-qrels-shared.txt`

[4]ImageCLEF-2020 Lifelog queries: `https://www.imageclef.org/system/files/ImageCLEF2020-test-topics.pdf`

Table A.1: List of queries used in the experiment of evaluating the performance of MyScéal operated by novice users. Symbol " | " indicates the hints in a query.

ID	Description
1	Taking a photograph of an A380 airplane | in Germany | before boarding a flight | in the late afternoon | in 2015 | on the 19th March.
2	It was the best cake I had in years | in an antique store. | I was alone drinking tea and eating cake. | I think I finished all the cake in 3 minutes. | It was in the UK | on a Saturday morning.
3	I was having beer after a long day of meetings. | It was a "corona extra" beer in a bottle. | I remember the room was dark. | I was relaxing in a hotel lobby bar. | I don't remember there was anyone else there. | It was in May 2018, in Wuhan.
4	Passing by a clocktower while running | in a park near my home. | It was in the early morning, around dawn. | I drove to the park | and I drove home again afterward. | It was a Saturday morning in February.
5	Four red figures, maybe they are aliens. | It looked like a painting of aliens. | There were walking on the desert. | There was a big red wall behind the painting. | And a TV, I think there was also a TV there. | I was having tea and sandwiches in March 2015.

of gradually revealed, in this pilot experiment.

Table A.2: Query batches and their queries used in the pilot study

Batch	ID	Description
A	LSC31	Eating fishcakes, bread, and salad after preparing my presentation in powerpoint. It must have been lunchtime. There was a guy in a blue sweater. I think there were phones on the table. After lunch, I made a coffee.
	LSC28	I remember a doll's house, a white doll's house. There were other people there, and candles too, I remember candles. There was a nice village scene in front of a lake in the picture. It was a Saturday.

Continued on next page

Table A.2 – continued from previous page

Batch	ID	Description
	LSC38	I was in my office taking a Skype call on a Friday morning. I clearly remember a bright green background on my laptop and a large image of a man's face on the screen. I had previously been having coffee in the Helix. I recall that it was a Friday.
	MMM07	I was using my laptop in someone's office. There were many cards hanging on the wall. There was a red jacket hanging on a door. There were many papers/badges stuck on the wall. It was noon. It was on Tuesday.
	MMM05	There was a mix of fruits in a large bowl. There was a white square pillow behind it. A vase was in the scene, near the pillow. There was a ladder at the back. It was at night time. It was in my house.
B	LSC26	A red car beside a white house on a cloudy day. I had driven for over an hour to get here. It was a Saturday in August and it was in the early afternoon.
	LSC35	I was shopping for clothes in the early morning. I think I was in Ireland and I was looking for a shirt and a belt. I don't think I bought anything. I went to a salon instead and waited for a haircut. Yeah, it was a Saturday.
	MMM04	There was a person wearing a yellow jacket. He was carrying a bag. He was riding a bicycle in front of the big glass building. There was a man at the back. It was in the early morning. It was on Monday.

Continued on next page

Table A.2 – continued from previous page

Batch	ID	Description
	MMM10	I was using my laptop in the Helix. My hand was typing on the keyboard. There was a headphone on the table beside the phone. There was a cup on the table also. I could see a plant. It was Friday morning.
	MMM06	There was a big plant. There was a man near the plant. The man was wearing a suit (with no tie) and holding a cup. Another man carrying a bag was at the back. It was on Tuesday. Prior to this, I was attending a presentation.
C	MMM03	There was a man wearing a blue shirt. He was carrying a big red suitcase. There was another man behind him. They were walking out of an elevator. It was on Thursday. I was at an airport.
	MMM02	I was eating a pizza. My hand was holding a pizza. There was a guy wearing a pink shirt talking to me. There was a black box on a table. It was on Friday morning. It happened at my workplace.
	MMM08	I was using my laptop with my hand clearly in the picture. There was a television on the wall. It was showing a football match. I could see the clock on the wall. I was lying on my couch at home. It was on Wednesday night.
	LSC48	Drinking tea from a large black cup when I was visiting a family member. There were other people there also. I think there was a hen or a cock on the table. There was some kind of pie also. I don't think I ate any pie though. It was a Saturday.

Continued on next page

Table A.2 – continued from previous page

Batch	ID	Description
D	LSC40	There was car racing on TV. It was F1 motor racing. I was at home with two other people on a Saturday. This was after going shopping.
	MMM09	I was having a meal with a woman in a restaurant. There was a vase on a table. There was also a candle on the table. A lot of food was on the table. It was in September. It was Thursday night.
	MMM01	I was talking to a woman. She is holding a bottle. There was a yellow bus and tables behind her. It was before noon. After that, I had a coffee with a guy in the Helix. It was on Tuesday.
	LSC36	I was making coffee in the morning using a professional-looking coffee machine. After making coffee, I was talking to someone. No, it was a few people. I remember the yellow doors behind them. After a few minutes, I went back to work. It was a Monday in Norway.
	LSC29	Catching my reflection in the bathroom mirror early on a Saturday morning when I was preparing to brush my teeth. There were pink wall tiles and I was wearing a white shirt. I had just finished watching some TV and then I drove somewhere.
	LSC25	Find the time when I was looking at an old clock, with flowers visible. There was a lamp also, and a small blue monster (perhaps a long rabbit) watching me. Maybe there were two monsters. It was a Monday or a Thursday. I was at home and in a bedroom.

A.2.2 Automatic and Interactive Manner Experiments

The list of lifelog queries used in the final experiments in Section 6.3 is shown in Table A.3. The queries are chosen from the query bank of LSC'21, including 23 official queries and five testing queries. The official queries and one testing query (LSC76) are randomly divided into four batches, namely A, B, C, and D, for the interactive manner experiment described in Section 6.3.4. Batch T contains four remaining testing queries which are used to instruct novice users on how to use the systems. Queries in this experiment follow the format of the LSC format in which queries are revealed gradually to users after every 30 seconds. Regarding the automatic manner (non-interactive) experiment described in Section 6.3.3, only use 23 official queries in four batches (except LSC76) are used.

Table A.3: Query batches and their queries used in the experiments in Section 6.3. Symbol " | " indicates the hints in a query. Batch T contains testing queries that are used to guide searchers on how to use systems. LSC76 is the testing query in LSC'21 but is randomly chosen to be a query in batch B for the interactive manner experiment.

Batch	**ID**	**Description**
A	LSC85	I was getting too much junk mail | so I put a sign on my door | asking for no more junk mail. | I remember I was wearing a blue shirt with cufflinks. | It was in 2015 | on the same day that I took a flight somewhere.
	LSC86	I was shopping for blue cups on a Wednesday evening. | I remember I was there with someone else who was wearing a blue jacket. | I recall buying two bags of stuff, | but I can't remember if it was actually cups or not. | Afterwards, I went grocery shopping | and then went home.
Continued on next page		

Table A.3 – continued from previous page

Batch	ID	Description
	LSC82	There was a white T-shirt for sale. | I remember it said I love bicycle. | It was in a bicycle and parts store. | A big sale, bicycles were half price. | It was in the afternoon before I drove to another store and then drove to my home. | It was on the 15th May 2015.
	LSC92	I remember the TagHeuer advertisement for a watch. | It was a footballer and a watch. | The footballer was sideways kicking the ball. | It was right before I went downstairs in the airport | in Frankfurt | in 2015.
	LSC97	I was buying fruit in a convenience store. | It only cost €2.99 | and was when I took a break from working at my desk in my office. | The store was called SPAR | and it was in Dublin City University | in the middle of an afternoon on a Tuesday in September 2015.
	LSC95	I needed the hotel name and address so I was taking a photo of a phone screen. | The phone was being held by a lady | with a suitcase, | wearing blue/white shirt and black trousers. | We were in an airport after a flight. | It was in China on 24th May 2021.
B	LSC103	Buying Glenisk yogurt in a SPAR supermarket, | right after passing by Tasty Deli. | There were two people in front of me at the counter. | The man on the left side had a dark blue jacket and gray trousers. | It was in 2015 | on the 10th of March.

Continued on next page

Table A.3 – continued from previous page

Batch	ID	Description
	LSC76*	I was in a room with two large blue paintings on the wall. | The paintings were of the sea with islands and sky. | I think there was also a speaker on a stand and a large screen in the room also. | It was some sort of ceremony. | I remember having coffee before and a sandwich after the event. | It was in 2018, on the 9th of May.
	LSC91	It was an orange suitcase, | a ride-on suitcase | one with a face. | I remember it was in a store selling cars and bicycle equipment. | It was in the afternoon | before I drove home via a supermarket.
	LSC105	Eating mandarins | and an apple, | while working on a paper on my laptop. | The paper is about Quantified Self. | It was in 2016 | in early September.
	LSC81	I was going into Northside Shopping Centre. | I was there to get new keys. | I drove to the shopping center from work | and then I drove home. | It was in 2015 | in the morning time.
	LSC96	I was looking at small computer chips on rolls. | It was in a small university electronics laboratory in China. | There were at least 100 rolls of small computer chips. | It was part of a tour of computing and engineering facilities | and I was with a small delegation of people. | It was in May 2018.

Continued on next page

Table A.3 – continued from previous page

Batch	ID	Description
C	LSC90	I needed to buy a blood pressure monitor. | So I was looking in a pharmacy | that sold Omron and Braun devices. | Afterward, I waited for a long time in my dentist's office, | before getting a coffee/bagel and driving to my own office. | It was in 2016.
	LSC80	I was building a computer alone in the early morning on a Friday | at a desk | with a blue background. | Sometimes I needed to refer to the manual. | I remember some Chinese posters on the desk background. | I was at Dublin City University in 2015.
	LSC89	Telescope in the mirror before going to the airport. | I was able to see my telescope and a red flower vase in the mirror in my bedroom. | I also remember a white violin, | flowers, and a nice painting. | I was playing with my computer and phone at the time. | It was in 2015.
	LSC94	Colleague in my office; she was carrying a large paper envelope full of documents. | The envelope looked very heavy. | She was wearing red trousers, | a white shirt, and a polkadot top. | I remember my office door was open. | It was in September 2016. On the 27th I think, in the afternoon.
	LSC88	I was drinking coffee while waiting in a car repair/sales store. | The store was called Joe Duffy | and it sold both Volvo and Mazda cars. I drove to the store | and I got a taxi away from the store | and went directly to the airport. | It was in May 2018.

Continued on next page

Table A.3 – continued from previous page

Batch	ID	Description
	LSC93	Boarding pass for PVG. | I was going to Shanghai and queuing at the airport gate | in Dublin on a day with a nice blue sky. | My name is clearly visible on the boarding pass which was for the second leg of my flight | (from FRA to PVG) | in 2015.
D	LSC102	I was playing a retro car-racing game on a laptop in my office in the afternoon. | I was playing a single race at the beginner level | just for a few minutes before I returned to my normal work, | speaking with colleagues | and working on my iMac. | It was on a Friday in the summertime.
	LSC104	Birds in a cage, a yellow one on the lower left. | There was also one box with a small, GREEN old car (Beetle-like). | No, the car was BLUE! | It was in 2018 | in May. | I think it was a Sunday.
	LSC84	I was roasting marshmallows | on a BBQ | at home | before watching football on the TV. | After roasting the marshmallows I cooked some hamburgers on the same BBQ. | It was in the evening time in May 2018.
	LSC83	Planning a thesis/dissertation on a whiteboard with my Ph.D. student, | who was wearing a blue and black stripey top | in my office in 2016. | We were using blue, black, and green pens. | After this, I went back to work at my computer. | It was on the 27th September.

Continued on next page

Table A.3 – continued from previous page

Batch	ID	Description
	LSC87	I was lost and looking for directions. | on a street, | close to an Asian restaurant called Maple Leaf. | It was in the late afternoon or evening | and it was in Wexford. | I had driven there in 2015.
	LSC101	My MacBook Air had a broken key so I was learning to fix it. | I was watching a video on a monitor about replacing a broken Macbook key | and my Macbook Air was beside the monitor | on a desk at home | late at night | in 2018.
T	LSC75	I was taking a photo of a lake with a DSLR camera. | It was my Sony camera. | I was driving outside of Sheffield before | and after stopping at the lake. | It was in 2015 | on a Saturday.
	LSC77	Drinking a bottle of Budweiser beer at home. | This was during a BBQ | in the evening | in the summer of 2018. | I had driven back home in someone else car | before putting on the BBQ and getting the beer on a dull evening.
	LSC78	There was a Blue Air aircraft | after I got off a flight. | I was walking across the airport ground | before going through the airport building | and getting a taxi. | I had just arrived in Dublin in May 2018.
	LSC79	I was taking a photo of grandfather clocks | while shopping in the UK. | It was a Saturday | in an antique store | in March 2015. | I had driven a rental car to the store.

Bibliography

[1] Cathal Gurrin, Alan F. Smeaton, and Aiden R. Doherty. Lifelogging: Personal big data. *Foundations and Trends in Information Retrieval*, 8(1):1–125, 2014.

[2] Steve Hodges, Lyndsay Williams, Emma Berry, Shahram Izadi, James Srinivasan, Alex Butler, Gavin Smyth, Narinder Kapur, and Ken Wood. Sensecam: A retrospective memory aid. In *International Conference on Ubiquitous Computing*, pages 177–193. Springer, 2006.

[3] Hyowon Lee, Alan F Smeaton, Noel E O'connor, Gareth Jones, Michael Blighe, Daragh Byrne, Aiden Doherty, and Cathal Gurrin. Constructing a sensecam visual diary as a media process. *Multimedia systems*, 14:341–349, 2008.

[4] Aiden R Doherty, Katalin Pauly-Takacs, Niamh Caprani, Cathal Gurrin, Chris JA Moulin, Noel E O'Connor, and Alan F Smeaton. Experiences of aiding autobiographical memory using the sensecam. *Human–Computer Interaction*, 27(1-2):151–174, 2012.

[5] Gabriel de Oliveira Barra, Alejandro Cartas Ayala, Marc Bolaños, Mariella Dimiccoli, Xavier Giró Nieto, and Petia Radeva. Lemore: A lifelog engine for moments retrieval at the ntcir-lifelog lsat task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[6] Van-Tu Ninh, Tu-Khiem Le, Liting Zhou, Graham Healy, Minh-Triet Tran, Duc-Tien Dang-Nguyen, Sinead Smyth, and Cathal Gurrin. A baseline interactive retrieval engine for the nticr-14 lifelog-3 semantic access task. In *The Fourteenth NTCIR Conference (NTCIR-14)*, 2019.

[7] Liting Zhou, Luca Piras, Michael Riegler, Mathias Lux, Duc-Tien Dang-

Nguyen, Cathal Gurrin, et al. An interactive lifelog retrieval system for activities of daily living understanding. In *CLEF (working notes)*, 2018.

[8] Nguyen-Khang Le, Dieu-Hien Nguyen, Vinh-Tiep Nguyen, and Minh-Triet Tran. Lifelog moment retrieval with advanced semantic extraction and flexible moment visualization for exploration. In *CLEF (Working Notes)*, 2019.

[9] Aaron Duane, Cathal Gurrin, and Wolfgang Huerst. Virtual reality lifelog explorer: lifelog search challenge at acm icmr 2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, pages 20–23, 2018.

[10] Bernd Münzer, Andreas Leibetseder, Sabrina Kletz, Manfred Jürgen Primus, and Klaus Schoeffmann. lifexplore at the lifelog search challenge 2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, pages 3–8, 2018.

[11] Cathal Gurrin, Klaus Schoeffmann, Hideo Joho, Andreas Leibetseder, Liting Zhou, Aaron Duane, Dang Nguyen, Duc Tien, Michael Riegler, Luca Piras, et al. Comparing approaches to interactive lifelog search at the lifelog search challenge (lsc2018). *ITE Transactions on Media Technology and Applications*, 7(2):46–59, 2019.

[12] Omar Shahbaz Khan, Björn Þór Jónsson, Jan Zahálka, Stevan Rudinac, and Marcel Worring. Exquisitor at the lifelog search challenge 2019. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 7–11, 2019.

[13] Soumyadeb Chowdhury, Philip J McParlane, Md Sadek Ferdous, and Joemon Jose. " my day in review" visually summarising noisy lifelog data. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 607–610, 2015.

[14] Martin Dodge and Rob Kitchin. 'outlines of a world coming into existence': pervasive computing and the ethics of forgetting. *Environment and planning B: planning and design*, 34(3):431–445, 2007.

[15] Marc Bolanos, Mariella Dimiccoli, and Petia Radeva. Toward storytelling from visual lifelogging: An overview. *IEEE Transactions on Human-Machine Systems*, 47(1):77–90, 2016.

[16] Cathal Gurrin, Rami Albatal, Hideo Joho, and Kaori Ishii. A privacy by design approach to lifelogging. 2014.

[17] Gordon Bell and Jim Gemmell. A digital life. *Scientific American*, 296(3):58–65, 2007.

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

[19] Jim Gemmell, Gordon Bell, Roger Lueder, Steven Drucker, and Curtis Wong. Mylifebits: fulfilling the memex vision. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 235–238, 2002.

[20] Jim Gemmell, Gordon Bell, and Roger Lueder. Mylifebits: a personal database for everything. *Communications of the ACM*, 49(1):88–95, 2006.

[21] Kiyoharu Aizawa, Yuto Maruyama, He Li, and Chamin Morikawa. Food balance estimation by using personal dietary tendencies in a multimedia food log. *IEEE Transactions on multimedia*, 15(8):2176–2185, 2013.

[22] Mark Hughes, Eamonn Newman, Alan F Smeaton, and Noel E O'Connor. A lifelogging approach to automated market research. 2012.

[23] Tu-Khiem Le, Manh-Duy Nguyen, Ly-Duyen Tran, Van-Tu Ninh, Cathal Gurrin, and Graham Healy. Dcu team at the ntcir-15 micro-activity retrieval task. In *Proceedings of the NTCIR-15 Conference*, 2020.

[24] L. Signal, M. Smith, M. Barr, J. Stanley, T. Chambers, Jiang Zhou, A. Duane, G. Jenkin, A. Pearson, C. Gurrin, A. Smeaton, J. Hoek, and C. Ni Mhurchu. Kids'cam: An objective methodology to study the world in which children live. *American Journal of Preventive Medicine*, 53 3:e89–e95, 2017.

[25] Olga Gelonch, Mireia Ribera, Núria Codern-Bové, Sílvia Ramos, Maria Quintana, Gloria Chico, Noemí Cerulla, Paula Lafarga, Petia Radeva, and Maite Garolera. Acceptability of a lifelogging wearable camera in older adults with mild cognitive impairment: a mixed-method study. *BMC geriatrics*, 19(1):110, 2019.

[26] Abigail J Sellen and Steve Whittaker. Beyond total capture: a constructive critique of lifelogging. *Communications of the ACM*, 53(5):70–77, 2010.

[27] Jonathon Keats. *You belong to the universe: Buckminster Fuller and the future*. Oxford University Press, 2016.

[28] Duc-Tien Dang-Nguyen, Luca Piras, Michael Riegler, Giulia Boato, Liting Zhou, and Cathal Gurrin. Overview of imageclef lifelog 2017: lifelog retrieval and summarization. 2017.

[29] Duc-Tien Dang-Nguyen, Luca Piras, Michael Riegler, Liting Zhou, Mathias Lux, and Cathal Gurrin. Overview of imagecleflifelog 2018: daily living understanding and lifelog moment retrieval. 2018.

[30] Duc Tien Dang Nguyen, Luca Piras, Michael Riegler, Liting Zhou, Mathias Lux, Minh Triet Tran, Tu-Khiem Le, Van-Tu Ninh, and Cathal Gurrin. Overview of imagecleflifelog 2019: solve my life puzzle and lifelog moment retrieval. CEUR Workshop Proceedings, 2019.

[31] Van-Tu Ninh, Tu-Khiem Le, Liting Zhou, Luca Piras, Michael Riegler, Pål Halvorsen, Minh-Triet Tran, Mathias Lux, Cathal Gurrin, and Duc-Tien Dang-Nguyen. Overview of ImageCLEF Lifelog 2020:Lifelog Moment Retrieval and Sport Performance Lifelog. In *CLEF2020 Working Notes*, CEUR Workshop Proceedings, Thessaloniki, Greece, September 22-25 2020. CEUR-WS.org <http://ceur-ws.org>.

[32] Kazuaki Kishida and Makoto P Kato. Overview of ntcir-12. In *NTCIR*, 2016.

[33] Makoto P Kato and Yiqun Liu. Overview of ntcir-13. In *NTCIR*, 2017.

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

[35] Cathal Gurrin, Hideo Joho, Frank Hopfgartner, Liting Zhou, and Rami Albatal. Ntcir lifelog: The first test collection for lifelog research. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 705–708, 2016.

[36] Cathal Gurrin, Xavier Giro-i Nieto, Petia Radeva, Mariella Dimiccoli, Håvard Johansen, Hideo Joho, and Vivek K Singh. Lta 2016: The first workshop on lifelogging tools and applications. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1487–1488, 2016.

[37] Stephen E Robertson and Karen Spärck Jones. Simple, proven approaches to text retrieval. Technical report, University of Cambridge, Computer Laboratory, 1994.

[38] Gobinda G Chowdhury. *Introduction to modern information retrieval*. Facet publishing, 2010.

[39] Ly-Duyen Tran, Manh-Duy Nguyen, Nguyen Thanh Binh, Hyowon Lee, and Cathal Gurrin. Myscéal: An experimental interactive lifelog retrieval system for lsc'20. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, pages 23–28, 2020.

[40] Ly-Duyen Tran, Manh-Duy Nguyen, Binh Thanh Nguyen, and Cathal Gurrin. An experiment in interactive retrieval for the lifelog moment retrieval task at imagecleflifelog2020. In *CLEF2020 Working Notes*, CEUR Workshop Proceedings, Thessaloniki, Greece, 2020. CEUR-WS.org <http://ceur-ws.org>.

[41] Luca Rossetto, Ralph Gasser, Silvan Heller, Mahnaz Amiri Parian, and Heiko Schuldt. Retrieval of structured and unstructured data with vitrivr. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 27–31, 2019.

[42] Ly-Duyen Tran, Manh-Duy Nguyen, Nguyen Thanh Binh, Hyowon Lee, and Cathal Gurrin. Myscéal 2.0: A revised experimental interactive lifelog retrieval system for lsc'21. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 11–16. 2021.

[43] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[44] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[45] Cyrus Rashtchian, Peter Young, Micah Hodosh, and Julia Hockenmaier. Collecting image annotations using amazon's mechanical turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's Mechanical Turk*, pages 139–147, 2010.

[46] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 15–29. Springer, 2010.

[47] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded compositional semantics for finding and describing

images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218, 2014.

[48] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

[51] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[52] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[53] Sijin Wang, Ruiping Wang, Ziwei Yao, Shiguang Shan, and Xilin Chen. Cross-modal scene graph matching for relationship-aware image-text retrieval. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1508–1517, 2020.

[54] Botian Shi, Lei Ji, Pan Lu, Zhendong Niu, and Nan Duan. Knowledge aware semantic concept expansion for image-text matching. In *IJCAI*, volume 1, page 2, 2019.

[55] Aaron Duane. *Visual access to lifelog data in a virtual environment.* PhD thesis, Dublin City University, 2019.

[56] Jim Gemmell, Aleks Aris, and Roger Lueder. Telling stories with mylifebits. In *2005 IEEE International Conference on Multimedia and Expo*, pages 1536–1539. IEEE, 2005.

[57] Aiden R. Doherty, Alan F. Smeaton, Keansub Lee, and Daniel P. W. Ellis. Multimodal segmentation of lifelog data. In *Large Scale Semantic Access to Content (Text, Image, Video, and Sound)*, RIAO '07, page 21–38, Paris, FRA, 2007.

[58] Marti A Hearst and Christian Plaunt. Subtopic structuring for full-length document access. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 59–68, 1993.

[59] Cathal Gurrin, Hyowon Lee, Niamh Caprani, ZhenXing Zhang, Noel O'connor, and Denise Carthy. Browsing large personal multimedia archives in a lean-back environment. In *Advances in Multimedia Modeling: 16th International Multimedia Modeling Conference, MMM 2010, Chongqing, China, January 6-8, 2010. Proceedings 16*, pages 98–109. Springer, 2010.

[60] Aiden R Doherty, Niamh Caprani, Ciarán Ó Conaire, Vaiva Kalnikaite, Cathal Gurrin, Alan F Smeaton, and Noel E O'Connor. Passively recognising human activities through lifelogging. *Computers in human behavior*, 27(5):1948–1958, 2011.

[61] Yang Yang, Hyowon Lee, and Cathal Gurrin. Visualizing lifelog data for

different interaction platforms. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, pages 1785–1790. 2013.

[62] Liting Zhou, Cathal Gurrin, Graham Healy, Hideo Joho, Thanh-Binh Nguyen, Rami Albatal, Frank Hopfgartner, and Duc-Tien Dang-Nguyen. Overview of the ntcir-16 lifelog-4 task. In *Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies*, pages 130–135. National Institute of Informatics, 2022.

[63] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678, 2014.

[64] Bahjat Safadi, Philippe Mulhem, Georges Quénot, and Jean-Pierre Chevallet. Lig-mrim at ntcir-12 lifelog semantic access task. In *12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[65] Hsiang Lun Lin, Tzu-Chieh Chiang, Liang-Pu Chen, and Ping-Che Yang. Image searching by events with deep learning for ntcir-12 lifelog. In *NTCIR*, 2016.

[66] Long Xia, Yufeng Ma, and Weiguo Fan. Vtir at the ntcir-12 2016 lifelog semantic access task. In *NTCIR*, 2016.

[67] Harrisen Scells, Guido Zuccon, and Kirsty Kitto. Qut at the ntcir lifelog semantic access task. In *NTCIR*, 2016.

[68] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[69] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew

Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[70] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[71] Bahjat Safadi and Georges Quénot. A factorized model for multiple svm and multi-label classification for large scale multimedia indexing. In *2015 13th International workshop on content-based multimedia indexing (CBMI)*, pages 1–6. IEEE, 2015.

[72] Paul Over. Trecvid 2013–an overview of the goals, tasks, data, evaluation mechanisms and metrics. 2013.

[73] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[74] Judy Hoffman, Sergio Guadarrama, Eric S Tzeng, Ronghang Hu, Jeff Donahue, Ross Girshick, Trevor Darrell, and Kate Saenko. Lsda: Large scale detection through adaptation. *Advances in neural information processing systems*, 27, 2014.

[75] George A Miller. *WordNet: An electronic lexical database*. MIT press, 1998.

[76] Mathias Lux. Content based image retrieval with lire. In *Proceedings of the 19th ACM international conference on Multimedia*, pages 735–738, 2011.

[77] Liting Zhou, Aaron Duane, Duc-Tien Dang-Nguyen, and Cathal Gurrin. Dcu at the ntcir-13 lifelog-2 task. NTCIR, 2017.

[78] Duc-Tien Dang-Nguyen, Luca Piras, Giorgio Giacinto, Giulia Boato, and Francesco GB DE Natale. Multimodal retrieval with diversification and relevance feedback for tourist attraction images. *ACM Transactions on*

Multimedia Computing, Communications, and Applications (TOMM), 13(4):1–24, 2017.

[79] Jie Lin, Ana Garcia Del Molino, Qianli Xu, Fen Fang, Vigneshwaran Subbaraju, Joo-Hwee Lim, Liyuan Li, and Vijay Chandrasekhar. Vci2r at the ntcir-13 lifelog-2 lifelog semantic access task. In *NTCIR*, 2017.

[80] Gemma Roig, Xavier Boix, Roderick De Nijs, Sebastian Ramos, Koljia Kuhnlenz, and Luc Van Gool. Active map inference in crfs for efficient semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2312–2319, 2013.

[81] Nguyen-Khang Le, Dieu-Hien Nguyen, Trung-Hieu Hoang, Thanh-An Nguyen, Thanh-Dat Truong, Duy-Tung Dinh, Quoc-An Luong, Viet-Khoa Vo-Ho, Vinh-Tiep Nguyen, and Minh-Triet Tran. Hcmus at the ntcir-14 lifelog-3 task. In *Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies*, pages 48–60, 2019.

[82] Min-Huan Fu, Chia-Chun Chang, Hen-Hsen Huang, and Hsin-Hsi Chen. Incorporating external textual knowledge for life event recognition and retrieval. In *Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies*, pages 61–71, 2019.

[83] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[84] Ana Garcia del Molino, Bappaditya Mandal, Jie Lin, Joo-Hwee Lim, Vigneshwaran Subbaraju, and Vijay Chandrasekhar. Vc-i2r@ imageclef2017: Ensemble of deep learned features for lifelog video summarization. In *CLEF (Working Notes)*, 2017.

[85] Mihai Dogariu and Bogdan Ionescu. A textual filtering of hog-based hierarchical clustering of lifelog data. In *CLEF (Working Notes)*, 2017.

[86] Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. *arXiv preprint cmp-lg/9406033*, 1994.

[87] Ergina Kavallieratou, Carlos R del Blanco, Carlos Cuevas, and Narciso García. Retrieving events in life logging. In *CLEF (Working Notes)*, 2018.

[88] Fatma Ben Abdallah, Ghada Feki, Mohamed Ezzarka, Anis Ben Ammar, and Chokri Ben Amar. Regim lab team at imageclef lifelog moment retrieval task 2018. In *CLEF (Working Notes)*, 2018.

[89] Minh-Triet Tran, Thanh-Dat Truong, Tung Dinh Duy, Viet-Khoa Vo-Ho, Quoc-An Luong, and Vinh-Tiep Nguyen. Lifelog moment retrieval with visual concept fusion and text-based query expansion. In *CLEF (Working Notes)*, 2018.

[90] Minh-Son Dao, Anh-Khoa Vo, Trong-Dat Phan, and Koji Zettsu. Bidal@ imagecleflifelog2019: The role of content and context of daily activities in insights from lifelogs. In *CLEF (Working Notes)*, 2019.

[91] Pengfei Zhou, Cong Bai, and Jie Xia. Zjutcvr team at imagecleflifelog2019 lifelog moment retrieval task. In *CLEF (Working Notes)*, 2019.

[92] Youguo Li and Haiyan Wu. A clustering method based on k-means algorithm. *Physics Procedia*, 25:1104–1109, 2012.

[93] Cathal Gurrin and ACM Special Interest Group on Multimedia. *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*. ACM, 2018.

[94] Cathal Gurrin. *Proceedings of the ACM Workshop on Lifelog Search Challenge*. Association for Computing Machinery, 2019.

[95] Cathal Gurrin, Tu-Khiem Le, Van-Tu Ninh, Duc-Tien Dang-Nguyen, Björn Þór Jónsson, Jakub Lokoč, Wolfgang Hürst, Minh-Triet Tran, and Klaus Schoeffmann. Introduction to the third annual lifelog search challenge (lsc'20).

In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pages 584–585, 2020.

[96] Cathal Gurrin, Björn Þór Jónsson, Klaus Schöffmann, Duc-Tien Dang-Nguyen, Jakub Lokoč, Minh-Triet Tran, Wolfgang Hürst, Luca Rossetto, and Graham Healy. Introduction to the fourth annual lifelog search challenge, lsc'21. In *Proceedings of the 2021 International Conference on Multimedia Retrieval*, pages 690–691, 2021.

[97] Cathal Gurrin, Liting Zhou, Graham Healy, Björn Þór Jónsson, Duc-Tien Dang-Nguyen, Jakub Lokoč, Minh-Triet Tran, Wolfgang Hürst, Luca Rossetto, and Klaus Schöffmann. Introduction to the fifth annual lifelog search challenge, lsc'22. In *Proc. International Conference on Multimedia Retrieval (ICMR'22). ACM, Newark, NJ*, 2022.

[98] Liting Zhou, Zaher Hinbarji, Duc-Tien Dang-Nguyen, and Cathal Gurrin. Lifer: An interactive lifelog retrieval system. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, pages 9–14, 2018.

[99] Ly-Duyen Tran, Manh-Duy Nguyen, Binh Nguyen, Hyowon Lee, Liting Zhou, and Cathal Gurrin. E-myscéal: Embedding-based interactive lifelog retrieval system for lsc'22. In *Proceedings of the 5th Annual on Lifelog Search Challenge*, pages 32–37. 2022.

[100] Manfred Jürgen Primus, Bernd Münzer, Andreas Leibetseder, and Klaus Schoeffmann. The itec collaborative video search system at the video browser showdown 2018. In *MultiMedia Modeling: 24th International Conference, MMM 2018, Bangkok, Thailand, February 5-7, 2018, Proceedings, Part II 24*, pages 438–443. Springer, 2018.

[101] Klaus Schoeffmann and Werner Bailer. Video browser showdown. *ACM SIGMultimedia Records*, 4(2):1–2, 2012.

[102] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[103] František Mejzlík, Patrik Veselý, Miroslav Kratochvíl, Tomáš Souček, and Jakub Lokoč. Somhunter for lifelog search. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 73–75, New York, NY, USA, 2020. Association for Computing Machinery.

[104] Jakub Lokoč, František Mejzlik, Patrik Veselỳ, and Tomáš Souček. Enhanced somhunter for known-item search in lifelog data. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 71–73. 2021.

[105] Jakub Lokoč, Tomáš Souček, and Gregor Kovalčík. Using an interactive video retrieval tool for lifelog data. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, pages 15–19, 2018.

[106] Adrià Alsina, Xavier Giró, and Cathal Gurrin. An interactive lifelog search engine for lsc2018. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, pages 30–32, 2018.

[107] Thanh-Dat Truong, Tung Dinh-Duy, Vinh-Tiep Nguyen, and Minh-Triet Tran. Lifelogging retrieval based on semantic concepts fusion. In *Proceedings of the 2018 ACM Workshop on The Lifelog Search Challenge*, LSC '18, page 24–29, New York, NY, USA, 2018. Association for Computing Machinery.

[108] Andreas Leibetseder, Bernd Münzer, Manfred Jürgen Primus, Sabrina Kletz, Klaus Schoeffmann, Fabian Berns, and Christian Beecks. lifexplore at the lifelog search challenge 2019. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 13–17, 2019.

[109] Jakub Lokoč, Tomáš Souček, Premysl Čech, and Gregor Kovalčík. Enhanced viret tool for lifelog data. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 25–26, 2019.

[110] Nguyen-Khang Le, Dieu-Hien Nguyen, Trung-Hieu Hoang, Thanh-An Nguyen, Thanh-Dat Truong, Duy-Tung Dinh, Quoc-An Luong, Viet-Khoa Vo-Ho, Vinh-Tiep Nguyen, and Minh-Triet Tran. Smart lifelog retrieval system with habit-based concepts and moment visualization. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, LSC'19, page 1–6, New York, NY, USA, 2019. Association for Computing Machinery.

[111] Radim Rehrek and Petr Sojka. Software framework for topic modelling with large corpora. 2010.

[112] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

[113] Tu-Khiem Le, Van-Tu Ninh, Duc-Tien Dang-Nguyen, Minh-Triet Tran, Liting Zhou, Pablo Redondo, Sinead Smyth, and Cathal Gurrin. Lifeseeker: Interactive lifelog search engine at lsc 2019. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 37–40, 2019.

[114] Chia-Chun Chang, Min-Huan Fu, Hen-Hsen Huang, and Hsin-Hsi Chen. An interactive approach to integrating external textual knowledge for multimodal lifelog retrieval. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 41–44, 2019.

[115] Son P Nguyen, Dien H Le, Uyen H Pham, Martin Crane, Graham Healy, and Cathal Gurrin. Vielens, an interactive search engine for lsc2019. In *Proceedings of the ACM Workshop on Lifelog Search Challenge*, pages 33–35, 2019.

[116] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[117] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 580–587, 2014.

[118] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[119] Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. Vse++: Improving visual-semantic embeddings with hard negatives. *arXiv preprint arXiv:1707.05612*, 2017.

[120] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[121] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[122] Yaxiong Wang, Hao Yang, Xueming Qian, Lin Ma, Jing Lu, Biao Li, and Xin Fan. Position focused attention network for image-text matching. *arXiv preprint arXiv:1907.09748*, 2019.

[123] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[124] Linchao Zhu, Zhongwen Xu, and Yi Yang. Bidirectional multirate reconstruction for temporal modeling in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2653–2662, 2017.

[125] Zhedong Zheng, Liang Zheng, Michael Garrett, Yi Yang, Mingliang Xu, and Yi-Dong Shen. Dual-path convolutional image-text embeddings with instance loss. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 16(2):1–23, 2020.

[126] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In *Proceedings of the European conference on computer vision (ECCV)*, pages 201–216, 2018.

[127] Zihao Wang, Xihui Liu, Hongsheng Li, Lu Sheng, Junjie Yan, Xiaogang Wang, and Jing Shao. Camp: Cross-modal adaptive message passing for text-image retrieval. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5764–5773, 2019.

[128] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX*, pages 104–120. Springer, 2020.

[129] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXX 16*, pages 121–137. Springer, 2020.

[130] Siqi Sun, Yen-Chun Chen, Linjie Li, Shuohang Wang, Yuwei Fang, and Jingjing Liu. Lightningdot: Pre-training visual-semantic embeddings for real-time image-text retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 982–997, 2021.

[131] Antoine Miech, Jean-Baptiste Alayrac, Ivan Laptev, Josef Sivic, and Andrew Zisserman. Thinking fast and slow: Efficient text-to-visual retrieval with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9826–9836, 2021.

[132] Karan Desai and Justin Johnson. Virtex: Learning visual representations from

textual annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11162–11173, 2021.

[133] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[134] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[135] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021.

[136] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[137] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678, 2015.

[138] Yikang Li, Wanli Ouyang, Bolei Zhou, Kun Wang, and Xiaogang Wang. Scene graph generation from objects, phrases and region captions. In *Proceedings of the IEEE international conference on computer vision*, pages 1261–1270, 2017.

[139] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs:

Scene graph parsing with global context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5831–5840, 2018.

[140] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

[141] Chunxiao Liu, Zhendong Mao, Tianzhu Zhang, Hongtao Xie, Bin Wang, and Yongdong Zhang. Graph structured network for image-text matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10921–10930, 2020.

[142] Haiwen Diao, Ying Zhang, Lin Ma, and Huchuan Lu. Similarity reasoning and filtration for image-text matching. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 1218–1226, 2021.

[143] Luca Rossetto, Matthias Baumgartner, Narges Ashena, Florian Ruosch, Romana Pernischová, and Abraham Bernstein. Lifegraph: A knowledge graph for lifelogs. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 13–17, New York, NY, USA, 2020. Association for Computing Machinery.

[144] Paul Bruton, Joss Langford, Matthew Reed, and David Snelling. Classification of everyday living version 1.0.

[145] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[146] Tai-Te Chu, Chia-Chun Chang, An-Zi Yen, Hen-Hsen Huang, and Hsin-Hsi Chen. Multimodal retrieval through relations between subjects and objects in lifelog images. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 51–55, New York, NY, USA, 2020. Association for Computing Machinery.

[147] D Rajasekar and Rajesh Verma. *Research methodology*. Archers & Elevators Publishing House, 2013.

[148] John W Creswell. *Educational research: Planning, conducting, and evaluating quantitative and qualitative research*. Pearson Education, Inc, 2012.

[149] Ly-Duyen Tran, Manh-Duy Nguyen, Binh T Nguyen, and Liting Zhou. Myscéal: a deeper analysis of an interactive lifelog search engine. *Multimedia Tools and Applications*, pages 1–18, 2023.

[150] Liting Zhou, Duc-Tien Dang-Nguyen, and Cathal Gurrin. A baseline search engine for personal life archives. In *Proceedings of the 2nd Workshop on Lifelogging Tools and Applications*, pages 21–24, 2017.

[151] Clinton Gormley and Zachary Tong. *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.", 2015.

[152] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[153] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

[154] Giuseppe Amato, Paolo Bolettieri, Fabio Carrara, Fabrizio Falchi, and Claudio Gennaro. Large-scale image retrieval with elasticsearch. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 925–928, 2018.

[155] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[156] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[157] Andreas Leibetseder and Klaus Schoeffmann. Lifexplore at the lifelog search challenge 2020. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 37–42, New York, NY, USA, 2020. Association for Computing Machinery.

[158] Gregor Kovalčík, Vít Škrhak, Tomáš Souček, and Jakub Lokoč. Viret tool with advanced visual browsing and feedback. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 63–66, New York, NY, USA, 2020. Association for Computing Machinery.

[159] Xirong Li, Chaoxi Xu, Gang Yang, Zhineng Chen, and Jianfeng Dong. W2vv++ fully deep learning for ad-hoc video search. In *Proceedings of the 27th ACM international conference on multimedia*, pages 1786–1794, 2019.

[160] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[161] Silvan Heller, Mahnaz Amiri Parian, Ralph Gasser, Loris Sauter, and Heiko Schuldt. Interactive lifelog retrieval with vitrivr. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 1–6, New York, NY, USA, 2020. Association for Computing Machinery.

[162] Omar Shahbaz Khan, Mathias Dybkjær Larsen, Liam Alex Sonto Poulsen, Björn Þór Jónsson, Jan Zahálka, Stevan Rudinac, Dennis Koelma, and Marcel

Worring. Exquisitor at the lifelog search challenge 2020. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 19–22, New York, NY, USA, 2020. Association for Computing Machinery.

[163] Aaron Duane, Björn Þór Jónsson, and Cathal Gurrin. Vrle: Lifelog interaction prototype in virtual reality: Lifelog search challenge at acm icmr 2020. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 7–12, New York, NY, USA, 2020. Association for Computing Machinery.

[164] Ahmed Alateeq, Mark Roantree, and Cathal Gurrin. Voxento: A prototype voice-controlled interactive search engine for lifelogs. In *Proceedings of the Third Annual Workshop on Lifelog Search Challenge*, LSC '20, page 77–81, New York, NY, USA, 2020. Association for Computing Machinery.

[165] Tu-ninh Van, Tu-Khiem Le, Liting Zhou, Luca Piras, Michael Riegler, Mathias Lux, Minh-Triet Tran, Cathal Gurrin, and Duc Tien Dang Nguyen. Lifer 2.0: Discovering personal lifelog insights using an interactive lifelog retrieval system. CLEF, 2019.

[166] Fatma Ben Abdallah, Ghada Feki, Anis Ben Ammar, and Chokri Ben Amar. Big data for lifelog moments retrieval improvement. In *CLEF (Working Notes)*, 2019.

[167] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[168] Ricardo F Ribeiro, Júlio Silva, Alina Trifan, José Luis Oliveira, and António JR Neves. Ua. pt bioinformatics at imageclef 2020: Lifelog moment retrieval web based tool. In *CLEF (Working Notes)*, 2020.

[169] Matthew Honnibal and Ines Montani. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7(1):411–420, 2017.

[170] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.

[171] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[172] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[173] Thao-Nhu Nguyen, Tu-Khiem Le, Van-Tu Ninh, Minh-Triet Tran, Nguyen Thanh Binh, Graham Healy, Annalina Caputo, and Cathal Gurrin. Lifeseeker 3.0: An interactive lifelog search engine for lsc'21. 2021.

[174] Emil Knudsen, Thomas Holstein Qvortrup, Omar Shahbaz Khan, and Björn Þór Jónsson. Xqc at the lifelog search challenge 2021: Interactive learning on a mobile device. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 89–93. 2021.

[175] Wei-Hong Ang, An-Zi Yen, Tai-Te Chu, Hen-Hsen Huang, and Hsin-Hsi Chen. Lifeconcept: An interactive approach for multimodal lifelog retrieval through concept recommendation. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 47–51. 2021.

[176] Luca Rossetto, Matthias Baumgartner, Ralph Gasser, Lucien Heitz, Ruijie Wang, and Abraham Bernstein. Exploring graph-querying approaches in

lifegraph. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 7–10. 2021.

[177] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26, 2013.

[178] Naushad Alam, Yvette Graham, and Cathal Gurrin. Memento: A prototype lifelog search engine for lsc'21. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 53–58. 2021.

[179] Ahmed Alateeq, Mark Roantree, and Cathal Gurrin. Voxento 2.0: a prototype voice-controlled interactive search engine for lifelogs. In *Proceedings of the 4th Annual on Lifelog Search Challenge*, pages 65–70. 2021.

[180] Ly-Duyen Tran, Manh-Duy Nguyen, Duc-Tien Dang-Nguyen, Silvan Heller, Florian Spiess, Jakub Lokoč, Ladislav Peška, Thao-Nhu Nguyen, Omar Shahbaz Khan, Aaron Duane, et al. Comparing interactive retrieval approaches at the lifelog search challenge 2021. *IEEE Access*, 11:30982–30995, 2023.

[181] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018.

[182] Ning Xu, An-An Liu, Jing Liu, Weizhi Nie, and Yuting Su. Scene graph captioner: Image captioning based on structural visual representation. *Journal of Visual Communication and Image Representation*, 58:477–485, 2019.

[183] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *European conference on computer vision*, pages 382–398. Springer, 2016.

[184] Manh-Duy Nguyen, Binh T Nguyen, and Cathal Gurrin. Graph-based indexing and retrieval of lifelog data. In *MultiMedia Modeling: 27th International Conference, MMM 2021, Prague, Czech Republic, June 22–24, 2021, Proceedings, Part II 27*, pages 256–267. Springer, 2021.

[185] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

[186] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5831–5840, 2018.

[187] Yikang Li, Wanli Ouyang, Bolei Zhou, Kun Wang, and Xiaogang Wang. Scene graph generation from objects, phrases and region captions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1261–1270, 2017.

[188] Yikang Li, Wanli Ouyang, Bolei Zhou, Jianping Shi, Chao Zhang, and Xiaogang Wang. Factorizable net: an efficient subgraph-based framework for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 335–351, 2018.

[189] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei Liu. Learning to compose dynamic tree structures for visual contexts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6619–6628, 2019.

[190] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[191] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[192] Yunsheng Bai, Hao Ding, Yang Qiao, Agustin Marinovic, Ken Gu, Ting Chen, Yizhou Sun, and Wei Wang. Unsupervised inductive graph-level representation learning via graph-graph proximity. *arXiv preprint arXiv:1904.01098*, 2019.

[193] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

[194] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[195] Hui Chen, Guiguang Ding, Xudong Liu, Zijia Lin, Ji Liu, and Jungong Han. Imram: Iterative matching with recurrent attention memory for cross-modal image-text retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12655–12663, 2020.

[196] Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. Multi-modality cross attention network for image and sentence matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10941–10950, 2020.

[197] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 357–366, 2021.

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

[199] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[200] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

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

[202] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*, 2020.

[203] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

[204] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

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

[206] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[207] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[208] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[209] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. *Advances in Neural Information Processing Systems*, 33:6616–6628, 2020.

[210] Dongxu Li, Junnan Li, Hung Le, Guangsen Wang, Silvio Savarese, and Steven C. H. Hoi. Lavis: A library for language-vision intelligence. *arXiv preprint arXiv:2209.09019*, 2022.

[211] Aniket Agarwal, Ayush Mangal, et al. Visual relationship detection using scene graphs: A survey. *arXiv preprint arXiv:2005.08045*, 2020.

[212] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, Michael Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. 2016.

[213] Desmond Elliott and Frank Keller. Image description using visual dependency representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, 2013.

[214] Hao Wu, Jiayuan Mao, Yufeng Zhang, Yuning Jiang, Lei Li, Weiwei Sun, and Wei-Ying Ma. Unified visual-semantic embeddings: Bridging vision and language with structured meaning representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6609–6618, 2019.

[215] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D. Manning. Generating semantically precise scene graphs from

textual descriptions for improved image retrieval. In *Workshop on Vision and Language (VL15)*, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[216] Ly-Duyen Tran, Naushad Alam, Yvette Graham, Linh Khanh Vo, Nghiem Tuong Diep, Binh Nguyen, Liting Zhou, and Cathal Gurrin. An exploration into the benefits of the clip model for lifelog retrieval. In *Proceedings of the 19th International Conference on Content-based Multimedia Indexing*, pages 15–22, 2022.

[217] Naushad Alam, Yvette Graham, and Cathal Gurrin. Memento 2.0: An improved lifelog search engine for lsc'22. In *Proceedings of the 5th Annual on Lifelog Search Challenge*, pages 2–7. 2022.

[218] Manh-Duy Nguyen, Binh T Nguyen, and Cathal Gurrin. Hada: A graph-based amalgamation framework in image-text retrieval. In *Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part I*, pages 717–731, 2023.

[219] Van-Tu Ninh, Manh-Duy Nguyen, Sinéad Smyth, Minh-Triet Tran, Graham Healy, Binh T. Nguyen, and Cathal Gurrin. An improved subject-independent stress detection model applied to consumer-grade wearable devices. In Hamido Fujita, Philippe Fournier-Viger, Moonis Ali, and Yinglin Wang, editors, *Advances and Trends in Artificial Intelligence. Theory and Practices in Artificial Intelligence*, pages 907–919, Cham, 2022. Springer International Publishing.

[220] Susana Brás, Jacqueline HT Ferreira, Sandra C Soares, and Armando J Pinho. Biometric and emotion identification: An ecg compression based method. *Frontiers in psychology*, 9:467, 2018.

[221] Taehyun Kim, Dongil Shin, and Dongkyoo Shin. Towards an emotion recognition system based on biometrics. In *2009 International Joint Conference*

on Computational Sciences and Optimization, volume 1, pages 656–659. IEEE, 2009.

[222] Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic vision transformers: Combining improvements in metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7409–7419, 2022.