Context-aware data-to-text generation.

UPADHYAY, A.

2024

Context-Aware Data-to-Text Generation

Ashish Upadhyay

A thesis submitted in partial fulfilment
of the requirements of
Robert Gordon University
for the degree of Doctor of Philosophy

May 2024

Abstract

Data-to-Text Generation (D2T) is the subfield of Artificial Intelligence (AI) and Natural Language Processing (NLP) that aims to build systems capable of summarising non-linguistic structured data into textual reports. D2T systems extract important insights from domain specific, mostly numerical, data and convey them in natural language reports which are more accessible to humans. This technology can help professionals in reducing their time spent on repetitive paperwork and allow them to focus on more important aspects of their jobs.

Literature presents two main approaches to building D2T systems: first, the rule-based approach, which uses domain specific rules with hand-crafted templates to produce the summaries; and second, the neural-based approach, which utilises a sequence-to-sequence learning method to learn the domain rules and text generation from parallel corpus of input data and output summaries. The rule-based systems are able to produce high quality accurate summaries but are difficult to scale and also produce monotonous summaries. Neural systems, on other hand, promise scalability across problems and domains (given the availability of parallel corpus for training) and are able to produce fluent, diverse and human looking texts. However, they struggle in maintaining high-quality accuracy and hallucinate with incorrect generations unsupported by input data.

Some D2T problems can be seen as a stream of time-stamped events with a textual summary written for each, for example, daily weather forecasts or regular sporting events. Human-authored event summaries often contain temporal contextual information where the presented information is derived from the data of other previously occurring events. While in other situations the summary content is influenced by the environmental context in which the events take place. Current state-of-the-art systems do not incorporate such contextual information while producing the event summaries, making these summaries less interesting and lacking detail compared to the human authored summaries.

In this thesis, we present several key methods for a D2T system pipeline that addresses temporal and environmental contextual problems. We first present a dynamic template method for D2T, called CBR-D2T, that helps in mitigating the accuracy and diversity trade-off between neural and rule-based systems. Empirical evaluations on a sports domain dataset suggest that CBR-D2T is able to achieve 6% better content accuracy than a neural benchmark while also maintaining better fluency and diversity than a template baseline. We then present a content type typology of D2T problems that is used to profile D2T datasets based on the different level of complexity present in the event summaries. This method uses the event representation to dynamically select a set of templates and organises them based on a pre-defined plan to generate accurate yet diverse event summary. We also present a CBR-based context aware content planning method, CBR-Plan, that uses the environmental context of an event to produce a content plan for an event summary. The content plans produced from this method more closely imitate the content plans in human-authored summaries in terms of the different types of information discussed, achieving 10% correlation with content plans in human-authored summaries than the content plans from summaries generated by neural benchmarks.

Finally, we present a context aware hybrid text generation method that utilises important temporal contextual data selected from past related events to produce a contextually-aware event summary. This method utilises the content plan produced from CBR-Plan to build an input sequence with important data from current as well as past events. The input sequence is then used by a state-of-the-art neural network to generate an initial contextual summary which is then post-edited with the CBR-D2T method to improve the accuracy of past-event information communicated in the summary. The user study conducted to measure the accuracy of our proposed method found that further post-editing the neural summary improves the content accuracy by more than double.

Keywords: Natural Language Generation, Data-to-Text Generation, Content Planning, Surface Realization, Case-Based Reasoning, Context Awareness

Declaration of Authorship

I declare that I am the sole author of this thesis and that all verbatim extracts contained in the thesis have been identified as such and all sources of information have been specifically acknowledged in the bibliography. Parts of the work presented in this thesis have appeared in the following publications:

- Upadhyay, A., Massie, S. and Clogher, S., 2020, June. Case-based approach to automated natural language generation for obituaries. In International Conference on Case-Based Reasoning (pp. 279-294). Springer, Cham.
  (**Chapters 3 and 4**)

- Upadhyay, A., Massie, S., Singh, R.K., Gupta, G. and Ojha, M., 2021, September. A Case-Based Approach to Data-to-Text Generation. In International Conference on Case-Based Reasoning (pp. 232-247). Springer, Cham. (Best poster award at SICSA PhD Conf 2022)
  (**Chapter 4**)

- Upadhyay, A. and Massie, S., 2022. Content-Type Profiling of Data-to-Text Generation Datasets. In International Conference on Computational Linguistics. COLING 2022
  (**Chapter 5**)

- Upadhyay, A. and Massie, S., 2022. A Case-Based Approach for Content Planning in Data-to-Text Generation. In International Conference on Case-Based Reasoning (pp. 380-394). Springer, Cham.
  (**Chapter 6**)

- Upadhyay, A. and Massie, S., 2023. Context-Aware Surface Realization for Data-to-Text Generation. In International Conference on Case-Based Reasoning 2023
  (**Chapter 7**)

Dissemination of Source-Code, Datasets, & Outputs The following GitHub repositories contain the source-code, data used and outputs produced in different experiments.

- **Chapter 3**: https://github.com/ashishu007/Obituary-Generation [1]
- **Chapter 4**: https://github.com/ashishu007/data2text-cbr
- **Chapter 5**: https://github.com/ashishu007/Content-Type-Profiling
- **Chapter 6**: https://github.com/ashishu007/data2text-cbr-plan
- **Chapter 7**: https://github.com/ashishu007/Context-Aware-D2T

[1] only the Obituary dataset

Acknowledgements

I extend my deepest gratitude to my esteemed supervisory team, Dr. Stewart Massie and Prof. Nirmalie Wiratunga of School of Computing at Robert Gordon University (RGU). Your unwavering support and invaluable guidance have been instrumental throughout my journey. I am profoundly appreciative of the meticulous and constructive supervision you have provided, shaping my research endeavours profoundly. Thank you sincerely for your dedication.

My heartfelt thanks extend to the AI-Research (AIR) Group, whose nurturing environment has allowed me to flourish not only as a researcher but also as a proficient communicator and writer. To all members, past and present, your camaraderie and insightful discussions have left an indelible mark on my academic pursuits. I am also indebted to the administrative and IT teams at the School of Computing for their steadfast support, ensuring a conducive environment for my research endeavours.

To my cherished friends and family, your unwavering support and encouragement have been my source of strength throughout this journey. I am profoundly grateful for your belief in me. Lastly, I express my gratitude to the CTML team at JPMorgan Glasgow for their continuous support and encouragement over the past year.

Contents

Abstract **ii**

Declaration of Authorship **iv**

Acknowledgements **vi**

1 Introduction **1**

1.1 Natural Language Processing (NLP) Landscape 3

1.1.1 Data-to-Text Generation 5

1.1.2 Commerical Data-to-Text Generation 7

1.2 Research Motivation 8

1.3 Research Question and Objectives 11

1.4 Contributions 12

1.5 Thesis Overview 14

2 Literature Review **17**

2.1 Natural Language Processing Tasks 17

2.1.1 Natural Language Understanding 18

2.1.2 Natural Language Generation 20

2.2 Neural Methods in Natural Language Processing 21

2.2.1 Data Representation and Word Embeddings 22

2.2.2 Neural Architectures for Text Processing 23

2.2.3 Pre-Trained Language Models and Transfer Learning 24

2.2.4 Large Language Models and Generative AI 27

2.3 Data-to-Text Generation Methods 29

2.3.1 Traditional D2T Methods 31

2.3.2 Neural D2T Methods 33

2.4 Context-Awareness in Data-to-Text Generation 36

2.5 Case-Based Reasoning 38

2.6 Conclusions 41

3 Datasets and Evaluation **42**

3.1 Formalizing D2T Concepts 42

3.2 Datasets 44

3.2.1 Obituary . . . 45
3.2.2 SumTime . . . 46
3.2.3 SportSett . . . 46
3.2.4 MLB . . . 47
3.3 Evaluation . . . 48
3.3.1 Accuracy . . . 49
3.3.2 Fluency . . . 51
3.4 Conclusion . . . 54

4 Case Based Approach to Data-to-Text Generation 55
4.1 Motivation . . . 55
4.2 Case-Based Methodology (CBR-D2T) . . . 56
4.2.1 Case-Base Creation . . . 58
4.2.2 Learning Parameters . . . 60
4.2.3 Summary Generation . . . 65
4.3 Experiment Setup . . . 66
4.3.1 Size of Case-Bases . . . 67
4.3.2 Baseline and Benchmark . . . 68
4.3.3 Evaluation . . . 68
4.4 Results and Discussion . . . 69
4.4.1 Comparison with Benchmark and Baseline . . . 70
4.4.2 Ablation Studies . . . 72
4.4.3 Qualitative Analysis . . . 73
4.5 Conclusion . . . 74

5 Content-Type Profiling of Datasets 75
5.1 Motivation . . . 75
5.2 Methodology . . . 77
5.2.1 Content Type Typology . . . 77
5.2.2 Building Content Type Classifier . . . 81
5.3 Experiment Setup . . . 83
5.3.1 Datasets . . . 83
5.3.2 Generation Systems . . . 83
5.3.3 Generation Systems' Accuracy Evaluation . . . 84
5.4 Results and Discussions . . . 84
5.4.1 Analysing Human Written Summaries . . . 84
5.4.2 Analysing System Generated Summaries . . . 86
5.4.3 Concept Drift in SportSett . . . 89
5.5 Conclusion . . . 89

6 Context-Aware Content Planning 95
6.1 Motivation . . . 95
6.2 Background . . . 97

6.3 Methodology . . . 100
6.3.1 Case Representation . . . 100
6.3.2 Environmental Features . . . 102
6.3.3 Content Planning Methodology (CBR-Plan) . . . 104
6.3.4 Entity Selection . . . 105
6.3.5 Author's Writing Style . . . 106
6.4 Experiment Setup . . . 107
6.4.1 Dataset . . . 107
6.4.2 Baseline and Benchmark Systems . . . 107
6.4.3 Evaluation Metrics . . . 108
6.5 Results and Discussion . . . 108
6.5.1 Content Selection and Ordering . . . 109
6.5.2 Content Type Distribution . . . 110
6.5.3 Content Diversity . . . 110
6.5.4 Author's Writing Style . . . 111
6.6 Conclusion . . . 112

7 Context-Aware Surface Realization **117**
7.1 Introduction . . . 117
7.2 Background . . . 119
7.3 Methodology . . . 120
7.3.1 Contextual Inter-Event Data Selection . . . 120
7.3.2 Surface Realisation . . . 125
7.3.3 A Working Example . . . 128
7.4 Experiment Setup . . . 130
7.4.1 Dataset . . . 130
7.4.2 Inter-Event Data Selection Models . . . 131
7.4.3 Surface Realisation Systems . . . 132
7.4.4 Evaluation Metrics . . . 132
7.5 Human Evaluation Setup for Accuracy . . . 133
7.6 Results . . . 135
7.6.1 Inter-Event Data Selection . . . 135
7.6.2 Surface Realisation . . . 137
7.6.3 Human Evaluation of Generated Summaries . . . 141
7.7 Conclusion . . . 142

8 Conclusion and Future Work **143**
8.1 Objectives Revisited . . . 144
8.1.1 O1: Review Literature . . . 144
8.1.2 O2: Profiling Datasets . . . 145
8.1.3 O3: CBR-D2T . . . 146
8.1.4 O4: Context-Aware Content Planning . . . 146
8.1.5 O5: Context-Aware Surface Realisation . . . 146

8.1.6 O6: Evaluation 147
8.2 Limitations and Future Works 148

List of Tables

2.1 NLI Example . . . 19
2.2 Some major neural-D2T methods . . . 33

3.1 Statistics of different datasets . . . 46

4.1 CBR-D2T against other systems on two datasets (Gold is human-authored summaries) . . . 70
4.2 Effect of training set size on CBR and Neural systems . . . 71
4.3 Ablation study results . . . 71
4.4 A summary generated from our $\mathbf{CBR}_{allS}$ system . . . 73

5.1 Content-Type classifiers performance on different datasets . . . 82
5.2 Statistics of different datasets . . . 83
5.3 Number of sentences from different categories manually annotated for error-rate evaluation . . . 85
5.4 System-wise error-rates of different datasets categorised by content-types (lower is better; ↓) . . . 92
5.5 Automated metric scores of different systems (↑, higher is better) . . . 93
5.6 Extractive Evaluation results of systems developed for MLB and SportSett dataset (↑, higher is better) . . . 94

6.1 CBR-Plan versus baseline and benchmark systems . . . 109
6.2 Effect of different environmental contextual themes . . . 109
6.3 Correlation between concept frequency of different systems versus gold . . . 110
6.4 Different authors' writing style . . . 111

7.1 Dataset stats for building theme classifiers . . . 131
7.2 Performance of Theme Classifiers for inter-event data selection . . . 136
7.3 Extractive Evaluation results with BLEU score (↑, higher is better) . . . 138
7.4 Extractive evaluation results using Information Extraction metrics . . . 138
7.5 Concept selection ability of different systems . . . 140
7.6 Human evaluation results divided into two inter-across themes . . . 141

List of Figures

1.1 Data-to-Text Generation table 3
1.2 Data-to-Text Generation summary 4
1.3 Natural Language Processing landscape 5
1.4 Excerpt of a human authored Data-to-Text summary and its corresponding input data table 8
1.5 Contextual information needed for summary generation in complex data-to-text generation domain 9
1.6 Context-Aware Data-to-Text Generation 14

2.1 Different tasks in NLP divided into two major subfields 18
2.2 Word Embeddings for words - king, queen, man, woman 22
2.3 LSTM 24
2.4 Transformer model 25
2.5 Prediction of next word using a Language Model 26
2.6 Two approaches to neural language modelling (a) Masked; (b) Causal Language Modelling 27
2.7 D2T subtasks 30
2.8 Traditional D2T architecture 31
2.9 CBR system working cycle (Aamodt and Plaza, 1994) 40

3.1 Excerpt of a human authored D2T summary with input data from sports domain (basketball) 43
3.2 Data-to-Text Datasets 45
3.3 Obituary dataset 46
3.4 SumTime dataset 47
3.5 SportSett dataset 48
3.6 MLB dataset 49

4.1 Excerpt of a basketball game's summary corresponding to its statistics table 56
4.2 Creating Case-Base for a component 58
4.3 A component's Case-Base from the proposed CBR-D2T system 59
4.4 Case alignment of a CBR system 62
4.5 Generating final summary 66
4.6 Ranking retrieved templates using GPT-2 LM 67

5.1 Different types of content in D2T summaries 76
5.2 Different content-types in a D2T summary 80
5.3 Generating the Content-Type Profile of a dataset 82
5.4 Distribution of content types in human authored summaries 86
5.5 Profile of Automated Systems on MLB Dataset 87
5.6 Profile of Automated Systems on SportSett Dataset 87
5.7 Profile of Automated Systems on SumTime Dataset 87
5.8 Profile of Automated Systems on Obituary Dataset 88
5.9 SportSett summaries' distribution by years 90
5.10 SportSett summaries' distribution by authors 91

6.1 Knowledge sources contributing to a basketball summary. 96
6.2 Excerpt of a basketball game's summary corresponding to its statistics table 98
6.3 Different types of content in a D2T summary 99
6.4 Content-plan of a summary taken from SportSett dataset 100
6.5 Frequency of concepts in the content planning case base of SportSett dataset 101
6.6 Problem-side representation (R_i) of Content Planning case-base 103
6.7 CBR-Plan methodology 113
6.8 Content type distribution of content plans on SportSett dataset 114
6.9 Proportion of different concepts in different systems 115
6.10 Content-Type distribution of summaries from different authors 116

7.1 Excerpt of a basketball game's summary from SportSett dataset corresponding to its statistics table 118
7.2 Content types in D2T summaries (as defined in Chapter 5) 119
7.3 Content Plan of an event summary 120
7.4 Context-Aware Surface Realisation 121
7.5 Feature correlation of Player Average Theme Classifier 125
7.6 Feature correlation of Team Streak Theme Classifier 125
7.7 Event input data: a basketball game statistics table 127
7.8 Inter event database and inter event theme classifier 128
7.9 Final input sequence for neural surface realiser 129
7.10 Output from neural surface realiser after 'Input Augmentation' phase . . . 130
7.11 Final event summary after post-editing 131
7.12 Home page of human evaluation website 135
7.13 Data page showing input data and output sentences from the D2T systems 136
7.14 The form to submit the data for counted supporting and contradicting facts 137
7.15 Content-Type distribution of summaries from different systems 139

List of Algorithms

1 **Calculate the loss value for each candidate generated in PSO algorithm** 65

Chapter 1

Introduction

Professionals across various fields find themselves encumbered by the overwhelming burden of routine paperwork, greatly limiting their capacity to engage in more vital and creative endeavours. For instance, a recent study revealed that General Practitioners (GPs) allocate less than 30% of their time to direct patient care, while dedicating over 50% of their work hours to desk tasks such as updating patient records (Lee, 2016). This is common beyond the medical profession, affecting other professionals including educators, engineers, and more, as they grapple with spending more time on clerical duties than on their core activities. This pervasive issue of excessive paperwork stifles innovation and hinders the fulfilment of professionals true potential.

With the emergence of internet and cloud computing, many businesses have started applying Artificial Intelligence (AI) and Natural Language Processing (NLP) techniques to improve their daily processes. Many business processes, such as Supply Chain Management, Sales, Accounting generate huge amount of structured non-linguistic data from automated systems often in the form of domain specific attributes-value pairs. Businesses often hire Data Analysts to analyse the structured data in order to take better business decisions (Dale, 2020).

In general, computers can understand structured data very well, while humans respond better to natural language text (Reiter and Dale, 2000). Data Analysts generally use Business Intelligence (BI) tools to extract patterns from the structured data and visualise it through charts and graphs on dashboards. However, they often still need to write an in-person custom textual report depending on the requirements of end-user or department.

Automating this task of custom reporting will help Data Analysts save time and thus focus more on exploration and discovery (Dale, 2020). This task of automatically summarising structured non-linguistic data into textual documents is known as Data-to-Text Generation (D2T) (Gatt and Krahmer, 2018; Reiter, 2007a; Wiseman et al., 2017).

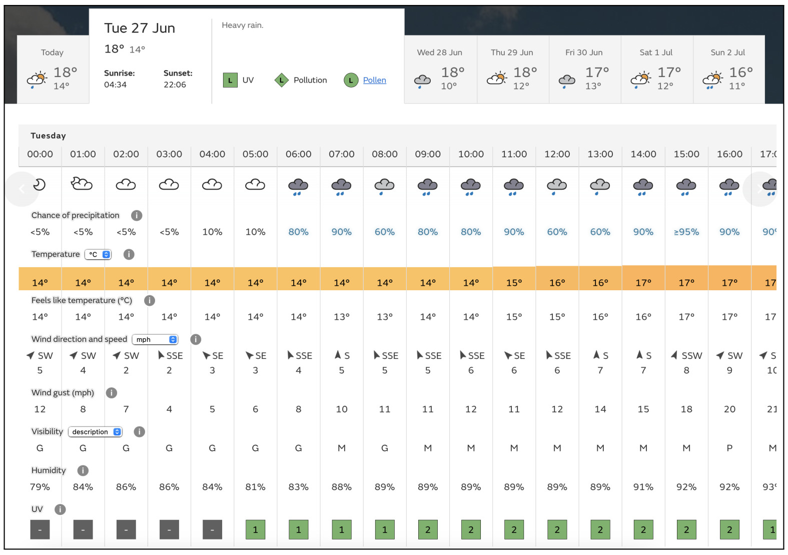
A D2T task of weather reporting is shown in Figures 1.1 and 1.2. The table in Figure 1.1 contains the statistics of weather forecasts for the week (26 June - 2 July 2023) with changes in the condition per hour (for 27 June 2023). The weather data is taken from MetOffice [1] and is for the Glasgow area in Scotland, UK. The corresponding weather report is shown in Figure 1.2 which describes the important information about the region's weather for the next few days.

A D2T system must perform several sub-tasks in order to generate such a textual summary of the data from the table Gatt and Krahmer (2018); Reiter (2007a); Wiseman et al. (2017). It must select and derive important attributes from the table, plan and organise the flow of information and then should coherently present them into textual format including all the domain specific keywords. Sometimes, to make the weather reports more interesting, a comparison maybe made between current or future conditions with past conditions in similar context (such as comparing end of June's weather from this year to last year). In that case, the D2T system must be able to understand the temporal relation of the data and be capable of processing it in order to extract interesting interpretations and comparisons.

There are many other use-cases in different domains that fit such requirements. For example, the compliance reporting in many industries such as manufacturing or aviation, where routine maintenance checks are required. An engineer might be required to investigate the conditions of different components and then provide a comprehensive report. A D2T system that takes in the collected investigative data, and provides a textual summary of important insights extracted from it can be very useful for the engineer as an initial draft of their report.

In the next section, the Natural Language Processing (NLP) landscape is discussed which is the general topic covering many natural language related tasks for AI, including D2T. Major developments in NLP and relating areas of AI are discussed, leading up to recent neural network related breakthroughs. This is followed with a brief discussion on different tasks in NLP and where D2T fits-in.

[1]https://www.metoffice.gov.uk



Today	Tue 27 Jun		Wed 28 Jun	Thu 29 Jun	Fri 30 Jun	Sat 1 Jul	Sun 2 Jul
18° 14°	18° 14° Sunrise: 04:34 Sunset: 22:06	Heavy rain. L UV, L Pollution, L Pollen	18° 10°	18° 12°	17° 13°	17° 12°	16° 11°

Tuesday

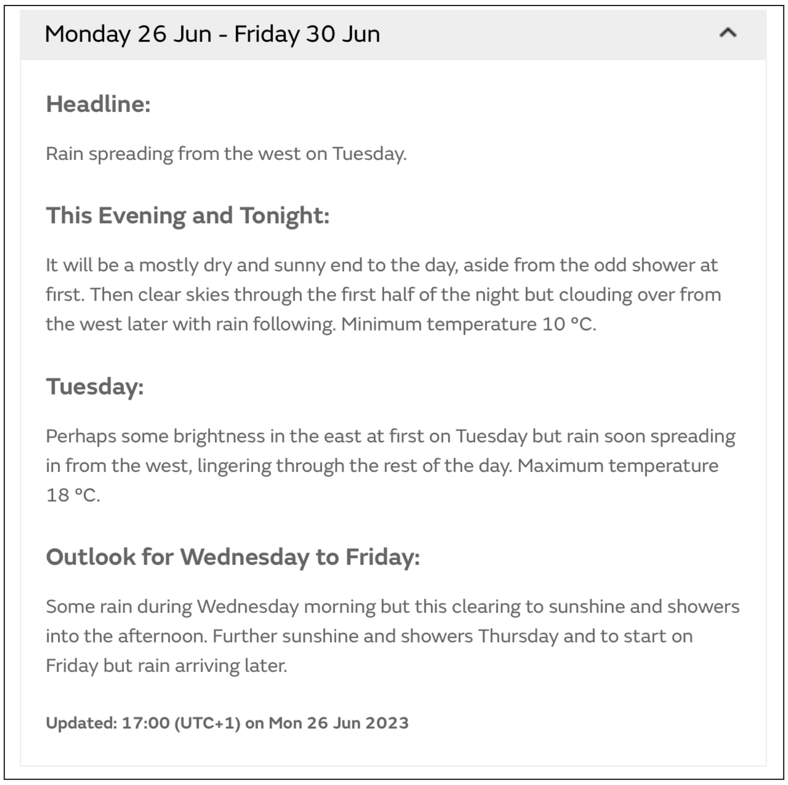
	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:0
Chance of precipitation	<5%	<5%	<5%	<5%	10%	10%	80%	90%	60%	80%	80%	90%	60%	60%	90%	≥95%	90%	90
Temperature °C	14°	14°	14°	14°	14°	14°	14°	14°	14°	14°	14°	15°	16°	16°	17°	17°	17°	17
Feels like temperature (°C)	14°	14°	14°	14°	14°	14°	14°	13°	13°	14°	14°	15°	15°	16°	16°	17°	17°	17
Wind direction and speed mph	SW 5	SW 4	SW 2	SSE 2	SE 3	SE 3	SSE 4	S 5	SSE 5	SSE 5	SSE 6	SE 6	SSE 6	S 7	S 7	SSW 8	SW 9	S 10
Wind gust (mph)	12	8	7	4	5	6	8	10	11	11	12	11	12	14	15	18	20	21
Visibility description	G	G	G	G	G	G	G	M	G	M	M	M	M	M	M	M	P	M
Humidity	79%	84%	86%	86%	84%	81%	83%	88%	89%	89%	89%	89%	89%	89%	91%	92%	92%	93
UV	-	-	-	-	-	1	1	1	1	2	2	2	2	2	2	2	2	1

Figure 1.1: D2T input table showing weather forecasts for the week

1.1 Natural Language Processing (NLP) Landscape

Natural Language Processing (NLP) is the subfield of Artificial Intelligence (AI) that aims to bridge the linguistic barrier between humans and machines by helping machines understand and generate natural language. NLP has been an active research field for decades with the earliest research dating back to the 1950s. In 1954, Georgetown University and IBM developed the first NLP system under the famous '*Georgetown-IBM experiment*' that translated about 60 Russian sentences to English (Garvin, 1968). This was a rule-based system that primarily used a set of six hand-crafted grammar rules along-with about 250 lexicon items in vocabulary to perform the translation.

Until the 1980s, the research focused on using **rule-based approaches** with computational models based on linguistic theories to process natural language. These approaches achieved limited success in real-world applications, mostly in database querying and semantic question-answering tasks. The rule-based NLP systems were highly reliant on hand-crafted knowledge and were difficult to scale across domains. A decade later, in the 1990s/early 2000s, the research witnessed a shift towards data-driven n-gram based **statistical approaches** that allowed machines to learn the rules of language processing from data itself. With the help of machine learning algorithms and feature engineering, the NLP systems were now scalable to different domains if an ample amount of labelled data was available. This approach also helped in building different text-analytics based


Monday 26 Jun - Friday 30 Jun

Headline:

Rain spreading from the west on Tuesday.

This Evening and Tonight:

It will be a mostly dry and sunny end to the day, aside from the odd shower at first. Then clear skies through the first half of the night but clouding over from the west later with rain following. Minimum temperature 10 °C.

Tuesday:

Perhaps some brightness in the east at first on Tuesday but rain soon spreading in from the west, lingering through the rest of the day. Maximum temperature 18 °C.

Outlook for Wednesday to Friday:

Some rain during Wednesday morning but this clearing to sunshine and showers into the afternoon. Further sunshine and showers Thursday and to start on Friday but rain arriving later.

Updated: 17:00 (UTC+1) on Mon 26 Jun 2023

Figure 1.2: D2T output summary with weather report

applications such as information extraction and sentiment analysis. However, there was still a reliance on hand-crafted feature engineering and manual data annotation that could be resource intensive and costly (Ruder, 2019).

The next decade, the 2010s, with the advancements in computing hardware and availability of data in the internet age, NLP research also moved towards **neural-based approaches** to utilise self-supervised and transfer learning methods in most cases (Ruder, 2019). The self-supervised methods in NLP allow Language Models (LMs) or Word-Embeddings (WEs) to learn general features of natural language by pre-training on large amount of raw text data without needing to manually label any samples (Devlin et al., 2019; Howard and Ruder, 2018; Mikolov et al., 2013; Pennington et al., 2014). The neural networks are capable to perform range of NLP tasks with fewer amount of labelled data and almost no feature engineering (Devlin et al., 2019; Howard and Ruder, 2018; Liu et al., 2019b). In recent years, Large Language Models (LLMs) such as BLOOM, GPT-3/4, PaLM-1/2 [2] (Brown et al., 2020; Chowdhery et al., 2022; Scao et al., 2023)

[2]GPT-4 and PaLM-2 have were only released with a technical report rather than a full paper like

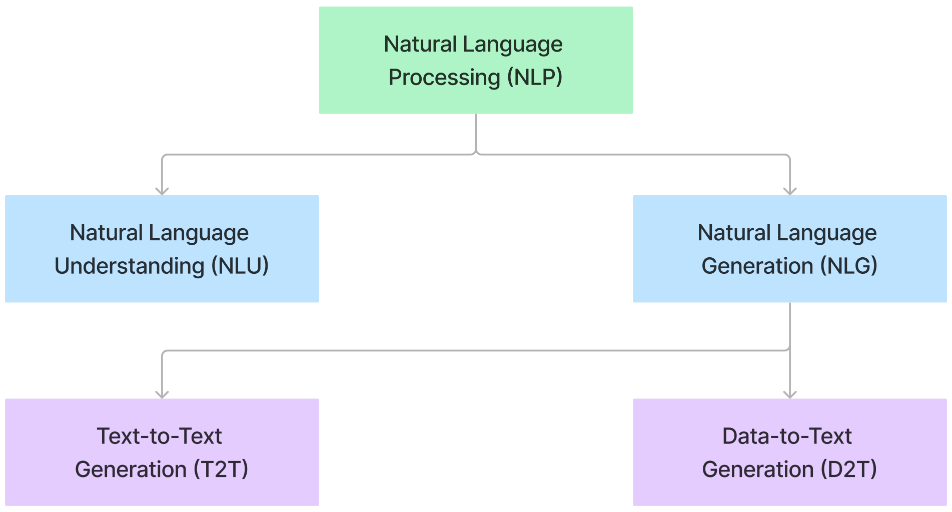

Natural Language Processing (NLP)
Natural Language Understanding (NLU)
Natural Language Generation (NLG)
Text-to-Text Generation (T2T)
Data-to-Text Generation (D2T)

Figure 1.3: Natural Language Processing landscape

have demonstrated on-par human level abilities on different NLP tasks, also in few-shot setting (Brown et al., 2020; Scao et al., 2023).

NLP as a general topic is often categorised in two subfields: Natural Language Understanding (NLU) and Natural Language Generation (NLG). As names suggest, the subfield of NLU aims to make machines better understand natural language by using computational methods, while the subfield of NLG works supports machines to automatically generate natural language. NLG problems, based on the input data, are usually divided in two types: Text-to-Text Generation (T2T), where the task is to generate textual output from textual input; and Data-to-Text Generation (D2T), where textual output is generated from non-linguistic structured data as input (Gatt and Krahmer, 2018; Reiter and Dale, 2000).

1.1.1 Data-to-Text Generation

Tasks such as automated report generation or robo-journalism are considered D2T problems with the input as mostly structured non-linguistic data while the required output is a textual summary. D2T combines data analytics with text generation where domain specific data is analysed to extract useful actionable insights and information, which is then summarised in textual form for better understanding by humans. The approaches

their earlier counterparts. GPT-4: https://openai.com/research/gpt-4; PaLM-2: https://blog.google/technology/ai/google-palm-2-ai-large-language-model/

to data-to-text generation can be classified into two types: traditional methods, systems based on carefully crafted rules and features; and neural methods, systems learning features and rules from training data.

Most traditional methods for D2T take a **pipeline-based** modular approach to solving the problem. The mainstream architecture for traditional D2T systems has four modules: Signal Analysis, Data Interpretation, Document Planning, Surface Realization (Reiter, 2007a). The Signal Analysis module is responsible for analysing the input data and looking for patterns in them. Data Interpretation module takes the those patterns and detects domain-specific messages and causal relations between them. The Document Planning module is responsible for selecting and arranging the messages into a textual document. This module must also decide the overall structure of the output textual document including sentence and paragraph breaks. Finally, the Surface Realization module uses the document plan with selecting messages and generates a textual document by coherently describing them.

The traditional approaches require explicit rule and template designing for each domain, thus require considerable amount of work when applying the same approach to similar but different domains. Neural approaches, on the other hand take an **end-to-end** approach where they learn the domain specific rules and language generation at the same time through representation learning (Wiseman et al., 2017). Sequence-to-Sequence (seq2seq) models are used for representation learning in deep learning methods (Vaswani et al., 2017). A standard seq2seq model has an **encoder** network that encodes the input which is then decoded into text using a **decoder** network. In the recent years, seq2seq models have become state-of-the-art in various text generation tasks, including data-to-text generation (Lewis et al., 2020; Puduppully et al., 2022, inter alia).

Although, most neural D2T systems unify the text generation task in an end-to-end setting, some have also embraced the pipeline based approach Castro Ferreira et al. (2019); Puduppully et al. (2019a). The most common neural pipeline based approach divides the task of summary generation into two sub-tasks: **content planning**, deciding *what to say* in the summary from input data; and **surface realisation**, tackling *how to say* the decided content into textual format Moryossef et al. (2019b); Puduppully and Lapata (2021). Both of these tasks are solved with separate modules, however, in some cases the learning method may share the loss calculation between the two modules (Puduppully et al., 2022). Studies have shown modular neural systems to perform better than their end-to-end counterparts (Castro Ferreira et al., 2019).

The emergence of Large Language Models (LLMs) and their use in '*Generative AI*' has gathered lots of press attention. LLM powered chatbot technologies [3] seem to be great at generating really good quality of human-like texts. Clark et al. (2021) also found that human annotators are unable to distinguish GPT3 (a precursor of ChatGPT) generated texts from human written ones. Suggesting GPT3 may be capable of passing Turing test and in possessing human-level capabilities at language generation tasks. Despite their success at general language generation tasks, hallucination seems to be a big problem with neural models, even LLMs. Recent examples have shown LLMs, such as ChatGPT, generate false and incorrect text despite looking human-like [4] (Ji et al., 2023).

1.1.2 Commerical Data-to-Text Generation

There have been many commercial players in the space of automated text generations. The very first company to commercialise D2T application was CoGenText which build a weather forecasting system called FoG, or Forecast Generator, in early 1990s [5] (Goldberg et al., 1994). FoG was used to generate weather forecasts in Canada (both in English and French language). The next decade saw some more names such as Narrative Sciences [6], Automated Insights [7], Data2Text Ltd. [8] and Yesop [9] commercialising D2T applications, mostly in sports and finance domain (Allen et al., 2010) [10]. Most of these started as small-scale startups before being acquired by big corporations, such as Tableu and Salesforce. The D2T products and services of these companies have been using traditional rule/template-based systems which are delivered either through cloud or on-premise software platforms. The generation of accurate and hallucination-free text has been the main USP of these companies Dale (2020, 2023).

With the rise of neural methods and generative AI technologies, commercial players in D2T continue to use the traditional methods but with plans to embrace the new technologies [11]. The rise of new technology has provided these commercial players an

[3]such as ChatGPT & Bard

[4]https://www.theguardian.com/technology/2023/jun/23/two-us-lawyers-fined-submitting-fake-court

[5]https://www.arria.com/blog/fog-the-birth-of-commercial-nlg/

[6]Acquired by Salesforce https://en.wikipedia.org/wiki/Narrative_Science

[7]https://automatedinsights.com/

[8]https://www.arria.com/

[9]https://yseop.com/

[10]https://web.archive.org/web/20100503074247/http://www.businessweek.com/magazine/content/10_19/b4177037188386.htm

[11]https://www.arria.com/blog/openai-gpt-system-what-does-it-do/
https://yseop.com/blog/what-is-generative-ai/
https://en.ax-semantics.com/gpt-3-vs-data-to-text-whats-the-right-content-generation-technology/

TEAM	WIN	LOSS	PTS	REB	AST	...
Pacers	4	6	99	40	17	···
Celtics	5	4	105	47	22	···

PLAYER	H/V	AST	REB	PTS	CITY	···
Myles Turner	H	1	8	17	Indiana	···
Thaddeus Young	H	3	8	10	Indiana	···
Isaiah Thomas	V	5	0	23	Boston	···
Kelly Olynyk	V	4	6	16	Boston	···
···						

The Boston Celtics defeated the host Indiana Pacers 105-99 at Bankers Life Fieldhouse on Saturday. It was the **second victory** over Pacers for the Celtics this season after emerging victorious in Boston 91-84 on Nov. 16. ... Isaiah Thomas led the team in scoring, totaling 23 points and five assists on 4–of–13 shooting. Kelly Olynyk got a rare start and finished second on the team with his 16 points, six rebounds and four assists. ... Boston will return to action on Monday against the New Orleans Pelicans.

Figure 1.4: Excerpt of a human authored Data-to-Text summary and its corresponding input data table

extended list of customer enquiries. In most cases, the customers still need to understand the differences between the two methods, traditional and neural approaches. But so far, the message seems to be clear that the neural technologies can only be adopted in the creative space while hallucination-free, robust text generation systems still requires traditional methods.

There are typically two types of end-user who use D2T systems in real-world applications. First, users who want to use such systems as a starting point and post-edit the generated summaries. For example, a business analyst preparing a finance report might want to use a D2T system to build an initial draft of their report. In this case, some error in the automatically generated text may be acceptable, but still should be low. On the other hand, the second type of user, who doesn't want to post-edit the generated summaries. For example, direct publishing of sports summaries, these summaries must be error-free and should not consist of any inaccurate generations. However, reporters - who write these sports reports, may be use a D2T system as a starting point and be fine with some inaccuracies before post-editing.

1.2 Research Motivation

Traditional systems, built with carefully-crafted rules and templates acquired from domain knowledge, are very good at producing high-quality textual summaries with accurate information. However, they lack diversity by generating monotonous texts and are difficult to scale across domains, since new domain will require new set of rules. On the other hand, neural systems are capable of learning such rules from parallel training

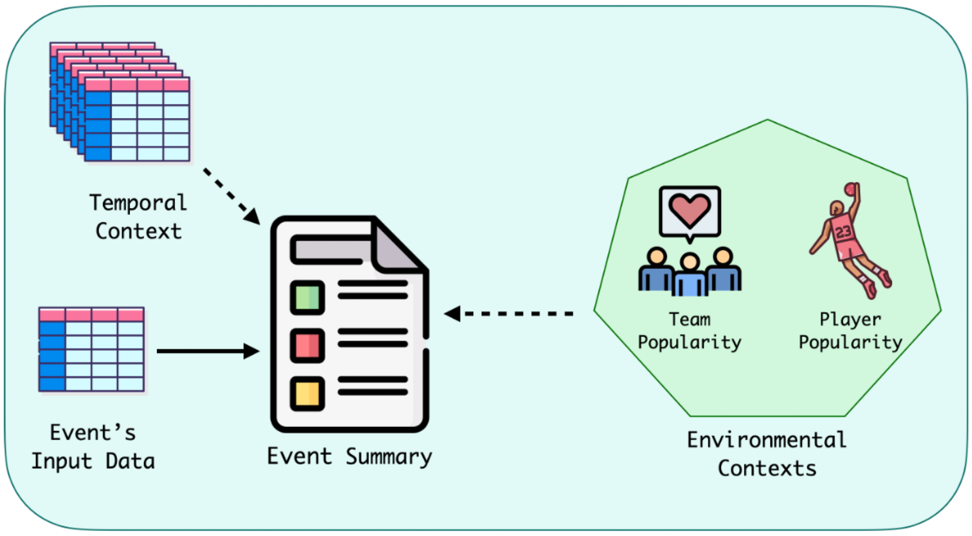


Figure 1.5: Contextual information needed for summary generation in complex data-to-text generation domain

data and are capable of producing diverse and fluent summaries while offering better scalability across domains, given ample training data is available (Puduppully, 2022). However, they often hallucinate by generating inaccurate summaries that are unfaithful to the input data (Ji et al., 2023). The hallucination problem is also observed in LLMs.

Apart from the accuracy and diversity trade-off, the current systems can also lack contextual awareness. D2T problems present different challenges in terms of the information presented in the textual summaries. Some are **easier D2T problems** with short summaries of only a few sentences that may require small amounts of input data to be processed. On the other hand, there are **complex D2T problems** that contain longer summaries with many sentences (often more than ten), and very rich domain specific vocabulary. These problems are likely to require the automated systems to process and analyse large amount of input data. In most cases, automated systems are also expected to derive new information from input data, which is not explicitly mentioned.

The events in D2T problems are not always independent. They are interrelated in the way that a reasonable size of D2T summaries contain information derived from other events as well. Often relationship is temporal with related events taking place over time. The context in which an event takes place is important and some understanding of the context is required in order to generate good summaries. Humans can use their knowledge of the domain to identifying specific related information hidden in the big volume of data. However, the current D2T systems find this challenging. For example, a basketball summary is shown in Figure 1.4 along-with its input data on left. In the second line of the summary it mentions that it was the Celtics' **second victory** over the Pacers this season (**bold-faced**). This type of information can only be derived by considering data from past events in addition to the current event. This type of content is common in human authored summaries and thus a D2T system should also be capable of introducing this type of content in their summaries, which they struggle to do (Thomson et al., 2020b; Wang, 2019).

Some D2T problems can be seen as a stream of time-stamped events with a textual summary of each event presenting the insights. An event is an activity happening in a time-period of interest for which a textual summary is written. For example, in sports reporting - a game played between two teams can be an event; whereas in weather forecasting - the time period and location for which the forecast is written can be considered one full event.

The human authored summaries of these events are a mix of different types of content. In simpler D2T problems, the summary content is mostly verbatim of records from input data. On the other hand, in complex D2T problems, the summary content is also complex and often describes information derived from multiple records of one or many events in the stream. For a D2T system to produce such complex content, it needs to be contextually aware by being able to process data from not just input from the current event but also the surrounding context.

In complex D2T problems, the contextual awareness can be of different types. In this research, we consider two types (as shown in Figure 1.5):

- **Environmental Context**: the general information of the world and environment in which an event take place, such as the player's or team's popularity in sports domain; and
- **Temporal Context**: the information from the input data of past events in the event-stream;

Current D2T systems tend not to be contextually aware and often fail to utilise contextual information effectively in the summary generation process. Even, LLMs have been shown to struggle in understanding the temporal aspects of NLP problems, thus struggling to generate summaries with accurate contextual information derived from past events. Event summaries produced by such systems are dissimilar to human authored summaries in terms of the different types of content. For example, system generated summaries contain very small amount of content derived from past events as compared to their human-authored counterparts. Thus, the current system generated summaries are less insightful and so less interesting to the end-user.

1.3 Research Question and Objectives

Current approaches to data-to-text generation suffer from two main issues:

- **Accuracy vs Diversity Trade-off**: current approaches to D2T struggle with a trade-off where traditional systems produce accurate but less diverse summaries while neural systems generate more diverse and fluent but inaccurate summaries;
- **Contextual Awareness**: current systems fail to fully consider contextual information when generating an event summary, making the summary less insightful and interesting than that of human authored ones;

Thus in this thesis, we investigate two research questions:

RQ1 How to build a D2T system that can effectively mitigate the accuracy and diversity trade-off in its generated summaries?

RQ2 How to build a D2T system that can effectively process the contextual information about the event and generate summaries with contextually aware insights?

In order to answer these research questions, we identify the following objectives:

O1 Review the Natural Language Generation literature specially towards Data-to-Text Generation

- Identify the current state-of-the-art in research methods for D2T problems within the broader context of NLG and NLP.
- Analyse the already existing work on contextual-awareness in D2T methods.

O2 Build a method to identify the mix of content types in the human-authored D2T summaries
Understanding the mix of content types in human-authored summaries will help in building a better understanding domain requirements. This will allow developers to curate their systems according to needs of different types of content.

- Identify a typology of content types in D2T summaries.
- Build a method to generate a profile of D2T dataset based on the mix of content types in their event summaries.

O3 Develop a data-driven methodology for D2T to mitigate the accuracy vs diversity trade-off in neural and template based D2T systems
Template-based D2T systems are accurate but produce monotonous summaries while neural D2T systems produce diverse and fluent summaries but hallucinate with inaccurate generations

- Develop a Case-Based Reasoning (CBR) methodology to dynamically select templates based on the event data, allowing summaries to be diverse yet accurate.

O4 Develop a Context-Aware Content Planning method to use environmental context in building the content plan for an event summary

- Design a method to utilise environmental context from a problem domain to build the content plan for a D2T summary with similar summary organisation as human written ones.

O5 Develop a Context-Aware Surface Realization methodology to use temporal context for generating final event summary

- Build a method to use temporal context for D2T summary generation in order to produce system generated summaries that have similar mix of different content types as of human-authored ones.

O6 Evaluate the proposed methods on complex real-world dataset

- Demonstrate the effectiveness of proposed methods on at-least one complex dataset that resembles real-world D2T problems

1.4 Contributions

This thesis makes four main contributions in D2T research, which are outlined below:

C1 CBR-D2T: A Case-Based Reasoning Approach to Data-to-Text Generation
Our first contribution is a CBR approach for D2T, called CBR-D2T, (Chapter 4)

that dynamically selects a set of templates to produce an event summary. A bank of templates is collected using a semi-automated process which is used to build a case-base. For each event, the case-base is queried to retrieve a set of appropriate templates based on the event's representation, which are filled with correct information from input to generate the event summary. The templates allow the summaries to be accurate and their dynamic selection based on event's representation helps the system in generating diverse summaries. An empirical evaluation is carried out on a sports domain dataset that demonstrates the effectiveness of CBR-D2T. It achieves better content accuracy than a neural benchmark system while also maintaining better fluency and richer generation vocabulary than a template-based baseline.

C2 Content Type Profiling of Data-to-Text Generation Datasets
The next contribution proposes a novel typology of different content types in D2T summaries (Chapter 5) based on their information source from the event stream. We also propose a text-classification method to profile a D2T dataset based on mix of different types of content in their event summaries. We used this method to profile different D2T datasets from various domain using their human-authored summaries and analyse the domain requirements. We identify that more than half of the content in human-authored summaries in sports domain is of complex types which is derived using multiple attributes. The profiles also suggest that more-than one-third of the content is derived from temporal context collected from past events. We also use this method to profile the summaries generated from different state-of-the-art neural methods in this domain and found that neural methods struggle in generating content of complex types.

C3 Context Aware Content Planning
The third contribution of this thesis is a context aware content planning method, CBR-Plan (Chapter 6), that utilises the wider environmental context of the event in the content planning process. Through our domain knowledge we identify some features that represent environmental context for an event, which is used to extend event's representation. Then a CBR method is used to produce a content plan for the event based on its extended representation. Empirical evaluation on a sports domain dataset demonstrates the benefit of CBR-Plan against several neural benchmarks and a template-based baseline. The content plan produced from CBR-Plan is more closer to content plans present in human-authored summaries as compared to other systems.

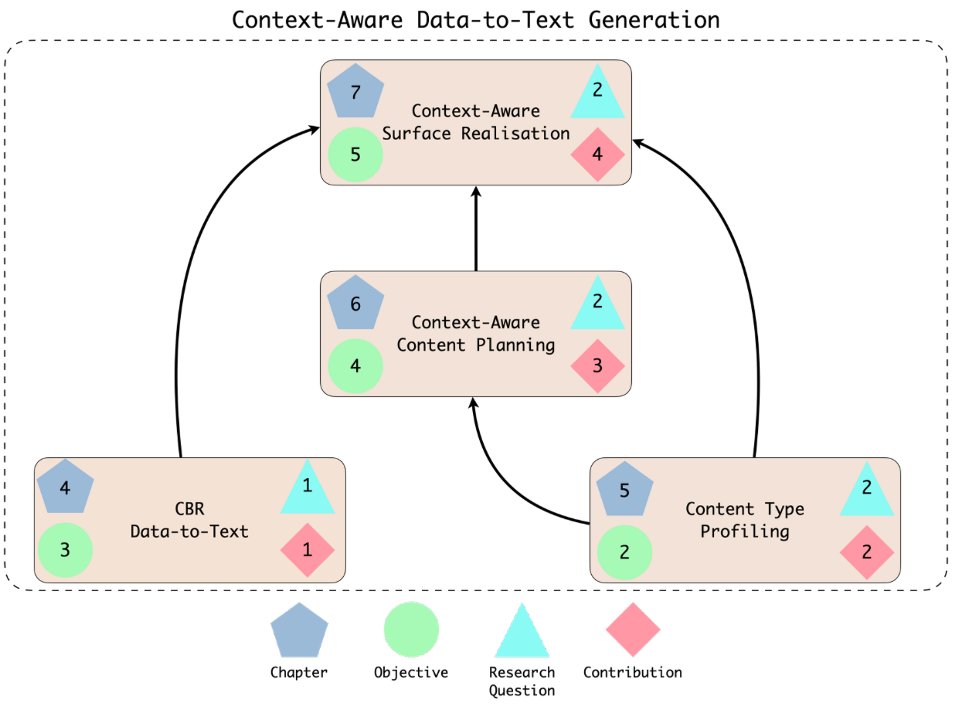

Context-Aware Data-to-Text Generation
7
Context-Aware Surface Realisation
2
5
4
6
Context-Aware Content Planning
2
4
3
4
CBR Data-to-Text
1
3
1
5
Content Type Profiling
2
2
2
Chapter
Objective
Research Question
Contribution

Figure 1.6: Context-Aware Data-to-Text Generation

C4 Context Aware Surface Realisation
The final contribution of this thesis is a context aware surface realisation method that uses temporal context for event summary generation (Chapter 7). The contribution is three-fold: first, a data selection method is used to select important historical data from past related events as temporal context; second, this data is arranged with current event's data according to the content plan produced by CBR-Plan which is used by a state-of-the-art neural D2T method to produce a draft contextual summary; and finally, this contextual summary is post-edited by using the CBR-D2T method to improve the content accuracy of temporal information in the event summary. Both automated and human evaluation of the summaries generated from the proposed system establish the improved content accuracy while maintaining similar fluency over different neural D2T benchmarks.

1.5 Thesis Overview

The rest of the thesis is organised as follows; also see Figure 1.6 for an overview.

Chapter 2 provides the background of D2T research field in broader context of NLP and

NLG. It starts with brief discussion on different NLP tasks leading into the recent deep learning based neural advancements in NLP. This sets the scene for major neural based advancements in D2T while also highlighting challenges in using Large Language Models based Generative AI approach in D2T. The chapter then discusses different approaches to D2T starting with established traditional rule-based methods and then diving into neural-based methods. The chapter also discusses literature around the contextual awareness in D2T.

Next, Chapter 3, provides a background into different types of publicly available D2T datasets and identifies the datasets that follow time-stamped events-based setup. This chapter also discusses the different types of evaluation measures used in literature for assessing quality of system generated summaries in terms of content accuracy and fluency.

Chapter 4 presents a CBR approach for D2T to dynamically select a set of templates for generating an event summary that mitigates the accuracy vs diversity trade-off between neural and template based systems. The chapter also presents an empirical evaluation of this method performed on a sports domain dataset.

In Chapter 5, we dive deeper to understand the nature of different D2T problems from various domains. Thus in this chapter, we propose a novel typology of content types in D2T summaries which is used to profile different D2T datasets from based on their human-authored summaries. We also use the same method to profile the system generated summaries from various state-of-the-art neural systems on these datasets and identify their struggle in generating content of complex types.

Chapter 6 presents a CBR approach for contextually aware content planning in D2T which utilises context from wider environment in which the events take place. This method is able to produce content plans for the an event summary which are more closer to human authored ones.

The final contribution, Chapter 7, we propose a methodology for context aware surface realisation that utilises information from past event's input data to derive contextually-aware event summaries. The proposed approach first selects historical contextual information for entities in the event using a machine learning approach. This information, along-with current event's input data and content plan (from previous chapter), is used by a CBR-assisted pipeline based neural surface realiser to generate the final event summary.

We conclude the thesis in Chapter 8 with a summary of our contributions and a review of the extent to which we met our research objectives. We also outline the limitations of the work presented in this thesis and considerations for future extensions.

Chapter 2

Literature Review

This chapter reviews the literature around neural-NLP and D2T. We start by discussing where D2T fits in the broader NLP landscape and recent advancements in neural-NLP. This helps in understanding the background for neural methods in D2T. Then we discuss the different approaches to D2T by classifying them in two categories: traditional approaches from pre-neural era that include both rule based and statistical method; and then neural-D2T, which utilise neural methods. Context awareness in D2T is also discussed before finishing this chapter with a brief discussion on Case-Based Reasoning (CBR), which is at the core of approaches developed in this thesis.

2.1 Natural Language Processing Tasks

Natural Language Processing (NLP) is the subfield of Artificial Intelligence (AI) that aims to bridge the gap between machines and humans. The field studies methods that can help machines better understand the natural language used by humans and act upon them. On a broad level, the NLP landscape is divided into two subfields: Natural Language Understanding (NLU), and Natural Language Generation (NLG). NLU aims to solve tasks that require the understanding of natural language and provides methods to extract structured information from free-form text. NLG, on the other hand, works the other way and aims to solve tasks that require structured information to be summarised into free-form text. However, in the past few years, tasks that require machines to process textual input in order to generate textual input (such as Machine Translation, Summarisation), are also considered part of NLG landscape (Gatt and Krahmer, 2018).

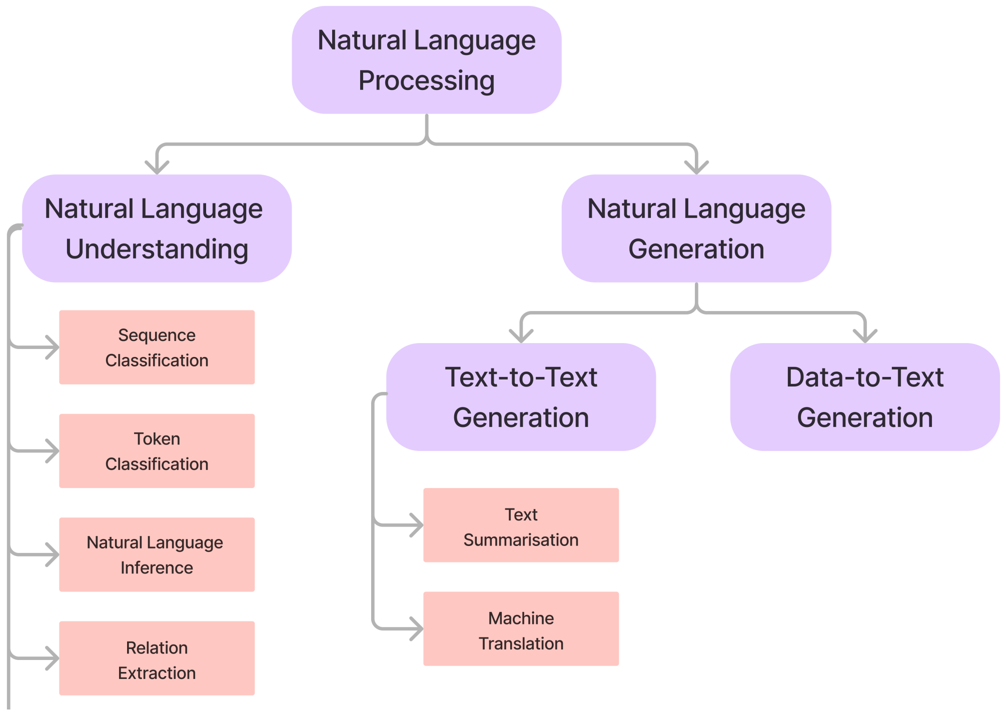


Figure 2.1: Different tasks in NLP divided into two major subfields

In the next subsections, we will discuss different tasks in NLP from by categorising them into these two subfields. Figure 2.1 provides an outline.

2.1.1 Natural Language Understanding

NLU tasks involve taking textual data as input and providing structured information as output. The output format of the structured information can vary from task to task ranging from just a single label to whole document, to JSON formatted fine-grained information with involved entities and their roles in the text. Here, we discuss some of the common NLU tasks.

Sequence Classification

The most common task in NLU is sequence classification where an algorithm is expected to categorize a piece of natural language text into some predefined class. As natural language text is often a sequence of tokens (words, characters or subwords), classifying such input is referred as sequence classification.

There are several real-world use-cases of sequence classification task, such as sentiment analysis, or document classification. In sentiment analysis, the goal is to identify the polarity of a text sequence based on its content. For example, using customer's review on an online platform to judge their likeliness of the product. On the other hand, document classification can be used for automatically categorising news articles into several topics (political, religious or technological).

Token Classification

Another common NLU task is token classification where every token of the textual sequence is assigned a label. Token classification is used in cases where fine-grained information is required from the text. Subtasks for token classification include Named Entity Recognition (NER), and Parts of Speech Tagging (PoST). NER methods can be used to identify specific entities from the text such as Person, Organisation or Dates while PoST methods can be used to identify the words that are noun, verbs, etc in the text.

This information about entities and parts-of-speech involved in the text can be very useful in many use-cases. For example, extracting different entities from invoices can help big organisations improve their processes.

Natural Language Inference

In Natural Language Inference (NLI) tasks, the goal is to predict if an hypothesis is supported by the premise or not. In general terms, NLI methods try to establish if a statement is supported or contradicted by another one. For example, the hypothesis in table 2.1 contradicts the premise. Real-world applications of NLI include scenarios where a given statement is matched against provided evidence including argumentation mining, or automatic auditing.

Table 2.1: NLI Example

Premise	**Label**	**Hypothesis**
A man inspects the uniform of a figure in some East Asian country.	contradiction	The man is sleeping.

Relation Extraction

The task of identifying semantic relationships between entities of text is known as relation extraction. Usually the relations are defined between two or more entities like person, location or organisation and fall under various pre-defined semantic categories. The ability to identify the relation between different entities allows for fine-grained information extraction grounded in the text documents.

In some domains, such as sentiment analysis, it might be possible to identify a certain event (positive or negative) using sequence classification methods, but the fine-grained information that causes the event is still grounded in the text. In these scenarios, relation extraction can help in identifying the cause of the event by predicting the relation between entities. For example, a negative event about an organisation can be identified by classifying news articles about it. However, what may have constituted for the negative event still remains grounded in text and relation extraction can help in getting more fine-grained information about it.

2.1.2 Natural Language Generation

Traditionally, NLG tasks have involved taking structured data as input and producing textual output describing that data. But more recently, tasks which process textual input to produce textual output are also considered part of NLG. However, based on the input data, these tasks can be divided into two types: Text-to-Text Generation (T2T), where input data is in textual format; and Data-to-Text Generation (D2T), where input data is in non-linguistic structured format.

Text-to-Text Generation

There are many use-cases where the requirement is to produce a piece of text based on some another text. For example, summarising a long document into a short report; or translating a sentence from one language to another. In these cases, the task is to take some text as input and process it to produce some another text as output. Two most common tasks in T2T are:

- **Summarization**: requires an automated system to produce an accurate summary of a long textual document. Summarisation can usually be of two types: abstractive, where new piece of text is generated using the input data by learning abstract concepts from it; and extractive, where important snippets and excerpts from input data are extracted and arranged to produce the final summary.
- **Machine Translation**: requires a system to produce the same input text in a

different language as output. Several real-world applications such as Google Translate [1] and Microsoft Translator [2] are used by millions of users for quick language translation.

Data-to-Text Generation

The task of summarising structured non-linguistic data into a textual report is termed as Data-to-Text Generation (D2T). The textual report in D2T problems is expected to summarise the important insights extracted from the input data. Use-cases for D2T can be found in may real-world applications ranging from the health domain to weather reporting. D2T systems take domain specific, mostly numeric data, and convert them into textual summaries with the important insights and information. Textual summaries allows the domain-specific data to be accessed by wider audience and help humans understand the insights better than when explained by other graphical methods (Dale, 2020; Gkatzia et al., 2016; REITER and DALE, 1997). In this thesis, the focus is on D2T tasks.

2.2 Neural Methods in Natural Language Processing

The advancements in computing technologies and availability of large datasets has allowed neural methods to achieve state-of-the-art performance across several tasks including NLP (Manning, 2015). Neural methods, through representation learning, offer a promise of learning domain and task specific rules from training corpus, without any need of explicit rule/feature engineering. Whilst, even with challenges, especially in higher computing budget for training and difficulty in gathering representative training data for the given task in a domain - neural methods are mostly preferred across many NLP tasks.

However, in D2T, neural methods still struggle with major hallucination problems and thus are not the preferred approach for real-world system development (Dale, 2023). Recent advancements in neural D2T methods have tried to borrow concepts from traditional D2T approaches such as separating planning from realization and have improved performance (Castro Ferreira et al., 2019; Moryossef et al., 2019b; Puduppully and Lapata, 2021). In this thesis, we also take a similar approach and utilise a modular architecture with separate planning and selection phases. Modules for these phases prepare a final

[1]https://translate.google.com/
[2]https://www.microsoft.com/en-us/translator/

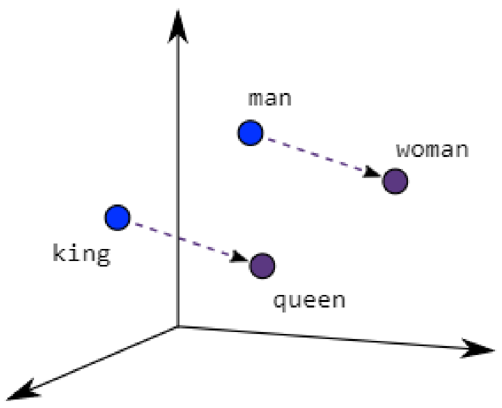


Figure 2.2: Word Embeddings for words - king, queen, man, woman

input sequence for a neural D2T system, whose output is then post-edited by a dynamic template system to improve accuracy.

In the following subsections, we briefly discuss the different key advancements in neural methods for NLP that have lead to the development of current state-of-the-art D2T systems. Following this, in next section, we will discuss the advancements in D2T methods, both traditional and neural.

2.2.1 Data Representation and Word Embeddings

Word Embeddings (WEs) are the fundamental blocks of neural methods in NLP. They represent textual data in a distributed representation using n-dimensional dense vectors, allowing neural methods to process the natural language very easily. WEs usually do not seem meaningful but exhibit similarity-property, i.e., embeddings of similar words are also similar. For example, the distance and direction between embeddings for word queen and king are same as the distance and direction between the embeddings of words woman and man Figure 2.2 [3].

Basic word embeddings are static in nature, i.e., they are pre-trained and stored for later use. Some commonly used static embeddings are: Word2Vec (Mikolov et al., 2013); or GloVe (Pennington et al., 2014). These embeddings are learned using a method similar to language model (Section 2.2.3) in self-supervised manner. In this setting, the language model tries to predict the next word based on some context and in this process learns

[3]Figure taken from: `https://www.analyticsvidhya.com`

the semantic relationships between words through back-propagation.

Even though these embeddings are able to provide semantic meaning to the words, they loose contextual information because of their static nature. For example, the embedding for the word 'bank' in sentences: 'Aberdeen is on the bank of River Dee' and 'Barclays is the most common high-street bank in UK', is same, despite having different senses. Recent advancements in word embeddings processed through pre-trained language models takes contextual information into account as well and provides the dynamic embeddings (Devlin et al., 2019; Peters et al., 2018; Radford et al., 2019).

2.2.2 Neural Architectures for Text Processing

NLP tasks, in terms of input and output mappings can be classified into two types: first, many-to-one mapping, where a sequence of input words (or tokens) are mapped to one output label as in sequence classification or NLI; and second, many-to-many mapping, where input sequence of many tokens is mapped to an output sequence of many tokens as in token classification or text generation (NLG) tasks.

To solve the many-to-one mapping type of tasks, encoder only architecture is used. The encoder architectures take a sequence of vectors as input (each vector corresponds to a token) and process them to create an encoded vectors of those vectors through some functions. These encoded vectors are then processed further to map them on to one class. On the other hand, the many-to-many tasks are solved using an encoder-decoder (often referred as sequence-to-sequence) architecture. In this architecture, an encoder model is used to create an encoded representation of input sequence which is then used by a decoder model to generate a sequence of tokens. All NLG tasks, including D2T, require a textual sequence as output, the primary approach for D2T with neural methods is encoder-decoder, or sequence-to-sequence, architectures.

Most neural methods for D2T make use of either Long Short Term Memory (Hochreiter and Schmidhuber, 1997) or Transformer (Vaswani et al., 2017) models. These methods are briefly discussed here:

Long Short-Term Memory Networks LSTM neural network models try to 'remember' the important information from a sequence while at the same time trying to forget irrelevant information through their series of gates. All gates are the functions of previous hidden state and current input and interact with the something called a cell state as they unroll over sequence of input (Figure 2.3). LSTMs can be used for both

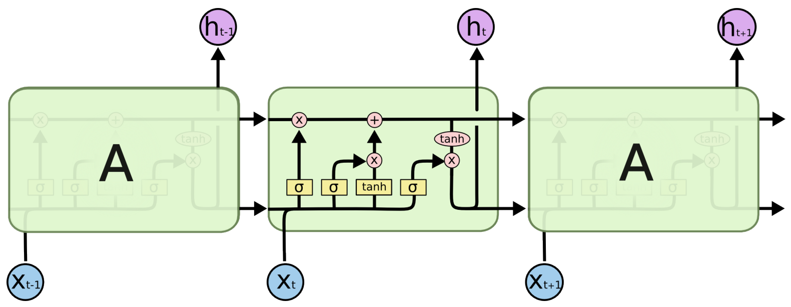


Figure 2.3: LSTM

encoder-only as well as encoder-decoder tasks. In encoder tasks, the final state h_t can be used as encoded representation for output class mapping. While in encoder-decoder settings, another LSTM is used to generate sequence of tokens step-by-step by using the encoded representation.

Attention Vanilla encoder-decoder models provide equal importance to all the parts of input sequence. Also, the input sequence is compressed into a single context vector for decoding, creating a bottleneck problem - where long information is compressed in a small representation. Attention mechanism (Bahdanau et al., 2015) provides a solution to this problem by aligning the output at each decoding step with whole input sequence. This allows the model to learn the most important part of the input aligning with the current step output.

Transformers Transformers models (Vaswani et al., 2017) make use of attention mechanism to create an encoded representation of the input sequence without any recurrence unit (as needed in LSTMs). This allows transformer models to scale in size and train much faster than LSTMs. Similar to LSTMs, transformer models can also be used in both encoder-only as well as encoder-decoder settings (Figure 2.4).

2.2.3 Pre-Trained Language Models and Transfer Learning

Language Modelling is the most fundamental task in NLP which requires a probabilistic model to predict the probabilities of a series of words based on the context (see Figure 2.5). The models used for solving this task are known as Language Models (LMs), and are the most fundamental building block in today's NLP research and applications.

LMs do not have many direct real-world use-cases [4], but through language modelling

[4] Although, with the advancements in Large Language Models, the LMs themselves are being used in

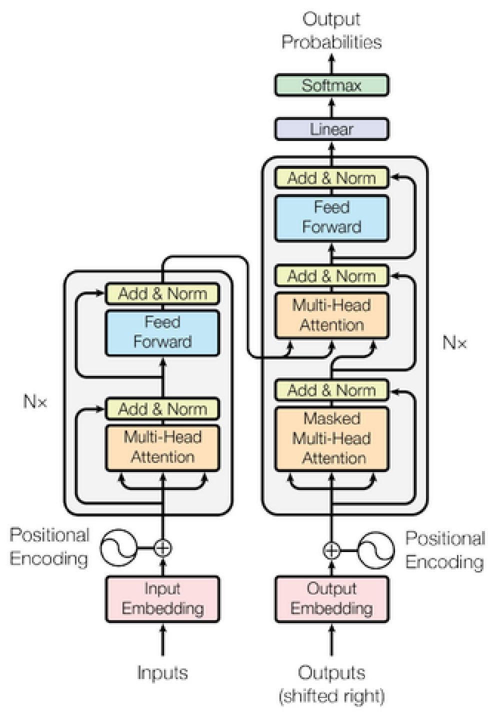


Figure 2.4: Transformer model

task, they learn the inherent features of natural language using self-supervised training. This learning is then transferred into a task specific model with transfer learning using domain and task specific data. There are two main approaches of training LMs: causal, where the LM is trained to predict next word based on the context of previous words; and masked, where the LM is asked to predict the masked word based on the context surrounding it (Figure 2.6).

Howard and Ruder (2018) first demonstrated the benefit of computer-vision style transfer learning (general pre-training, then task-specific fine-tuning) in NLP. They present a transfer learning method called Universal Language Model Fine-Tuning (ULMFiT), where can be used to pre-train a language model on large unlabelled text corpora and then fine-tuned on different NLP tasks with domain and task specific data. The success of this approach combined with advancements in transformer models allowed the development of more and better transfer learning approaches in NLP.

Generative Pre-trained Transformers (GPT) (Radford et al., 2019), a precursor to current

many real-world tasks through Generative-AI techniques

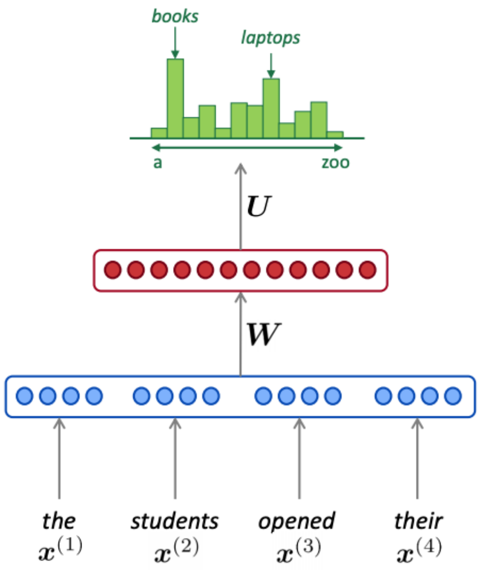


Figure 2.5: Prediction of next word using a Language Model

popular ChatGPT [5], used a Transformer architecture for Language Modelling instead of LSTMs as used in ULMFiT. The pre-training used the Causal Language Modelling task which was trained to predict the next token for a given sequence.

Causal Language Models albeit outperforming previous methods, are limited to just one-directional context. However, the nature of natural language dictates to have bi-directional contextual understanding in order to fully grasp the meaning of a sentence. Masked Language Modelling techniques help overcome this drawback and introduce the bi-directional understanding by asking the model to predict the value of a masked token somewhere in between the sequence of tokens.

Language Models such as BERT, or Bi-directional Encoder Representations from Transformers (Devlin et al., 2019), are an example of MLM. The model is pre-trained to predict a masked word in the sequence based on surrounding context. Several incremental improvements have been made in these LMs including RoBERTa, DeBERTa, etc (He et al., 2023; Liu et al., 2019b, inter alia). A family of domain-specific LMs are created by fine-tuning these LMs on domain-specific data in domains such as health and science. SciBERT, MedBERT (Beltagy et al., 2019; Rasmy et al., 2021, inter alia) are a few examples.

[5]`https://chat.openai.com/`

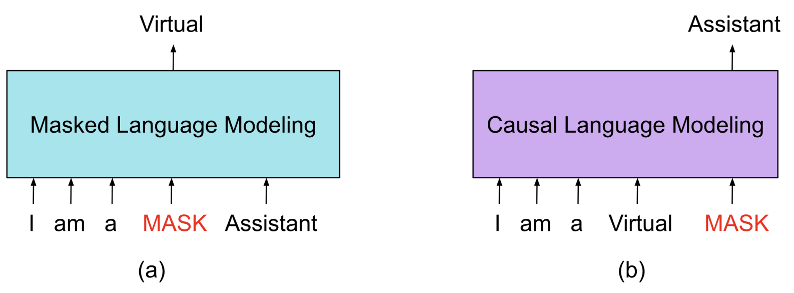


Figure 2.6: Two approaches to neural language modelling (a) Masked; (b) Causal Language Modelling

A family of pre-trained language models, called sequence-to-sequence LMs, utilise encoder-decoder architecture. LMs such as BART (Lewis et al., 2020), PEGASUS (Zhang et al., 2020a), T5 (Raffel et al., 2020) are pre-trained on many-to-many mapping tasks where a sequence of input tokens is mapped to a sequence of output tokens. These language models are often used with non event-based D2T simpler datasets (Nan et al., 2021).

Use of transformer architecture allowed the faster training, meaning more layers could be stacked up for a much deeper model. This allowed the models to be trained on even bigger datasets and eventually perform better on many language related tasks. Where LSTM based LMs such as ULMFiT only demonstrated their capabilities on only a handful of NLP tasks, the transformer based LMs have become state-of-the-art on many NLP tasks now [6]. In complex D2T problems, since the data availability is still scarce due to the complex nature of domains, pre-training of LMs is difficult. Thus, models trained from scratch with LSTMs are still prevalent in neural approaches (Puduppully et al., 2022; Wiseman et al., 2017), along with some transformer based methods as well (Narayan et al., 2020; Rebuffel et al., 2020).

2.2.4 Large Language Models and Generative AI

Transformer neural network has allowed horizontal scaling of Language Models by stacking multiple hidden layers making them very large in size and number of parameters. This type of horizontal scaling was an issue with RNN/LSTM based models because they struggled with the issue of vanishing and exploding gradients. Such larger models are usually referred as Large Language Models (LLMs) and exhibit many multi-task

[6]https://rajpurkar.github.io/SQuAD-explorer/

learning abilities and perform reasonably well even in few-shot settings (Brown et al., 2020; Chowdhery et al., 2022; Scao et al., 2023).

LLMs with their textual interface provide an easier approach for humans to interact with them and allows them to solve many NLP tasks. API based services such as ChatGPT [7] and Bard [8] has become very popular recently and allow general users to use the LLM capabilities in their real-world tasks. However, there usefulness only remains limited to open-ended creative tasks such as drafting emails.

Luccioni and Rogers (2023) set a threshold on pre-training data (1-billion tokens) used by a Language Model for it to be considered an LLM. This allows earlier transformer-based language models such as BERT and GPT-2 to be considered an LLM, even though they only have about hundred million parameters. However, these models are not useful in 'Generative AI' context and are fine-tuned on domain specific data based on the down-stream task. On the other hand, models such as GPT-3, PaLM and BLOOM, with more than hundred billion parameters, can perform many tasks without further fine-tuning. These models however, do use a few samples to understand the domain and task specifics by making use of in-context learning paradigm (colloquially referred as prompt-engineering) (Dong et al., 2023). However, there isn't wide evidence suggesting these larger LMs with high parameter count to achieve state-of-the-art performance across tasks and domains (Luccioni and Rogers, 2023).

LLMs have demonstrated that scaling of causal language models in terms of parameters and pre-training tokens can improve the generalisability of the language models. Recent works have also demonstrated the trade-off between parameter count and pre-training data size in order to achieve similar performance (Hoffmann et al., 2022). However, usefulness of LLMs in a challenging real-world task still needs to be established.

Despite the good promise, there are many concerns with LLMs:

- Hallucinations: the issue of hallucination and inaccurate generation is prevalent with LLMs. With bigger size, the inaccurate generations have become ever stronger, making the inaccurate texts sound true.
- Maintainability: as a single model is trained and tuned for many tasks, the danger of catastrophic forgetting may arise. Tuning the model for certain group of tasks may result into degraded performance in other (Chen et al., 2023).

[7]https://chat.openai.com/
[8]https://bard.google.com

- Data Leakage: since many API based models are closed-source and their pre-training data is unclear, its difficult to understand how these LLMs were trained. In many cases, as in ChatGPT, the model's interaction with users is used to re-train it, thus allowing a major possibility of data leakage.

A recent news reported that a lawyer firm in the US was fined for using inaccurate court citations generated by ChatGPT [9]. It is common for LLMs to make such errors and thus making them difficult to use in real-world use cases without proper guard-rails and human supervision.

2.3 Data-to-Text Generation Methods

In order to produce the textual summary, a D2T system must solve six fundamental sub-tasks (Gatt and Krahmer, 2018; REITER and DALE, 1997; Reiter and Dale, 2000):

1. **Content Determination**: deciding the information from input data to be included in the final text summary;
2. **Text Structuring**: deciding the ordering of information in the final summary;
3. **Sentence Aggregation**: deciding individual sentences for different information;
4. **Lexicalisation**: finding the correct domain-specific words and phrases to express the information;
5. **Referring Expression Generation (REG)**: the task of identifying domains objects through words and phrases; and
6. **Linguistic Realisation**: producing well formed sentences by combining all the selected words and phrases

An example of D2T summary generation process from neonatal intensive care domain is shown in Figure 2.7. The time-series graph (a) represents the input data for which a textual summary needs to be produced. The first task is (a) content determination, ie, the D2T system must decide what are the important aspects in the data. In this case, the low heart rate (bradycardias) is the important aspect of presented data. Next task is (b) text structuring, where the system needs to decide the ordering in which the selected important data should be presented to the reader. Then, as part of (c) aggregation, lexicalisation and REG, the system needs to arrange and express these selected data into

[9]https://www.theguardian.com/technology/2023/jun/23/two-us-lawyers-fined-submitting-fake-court

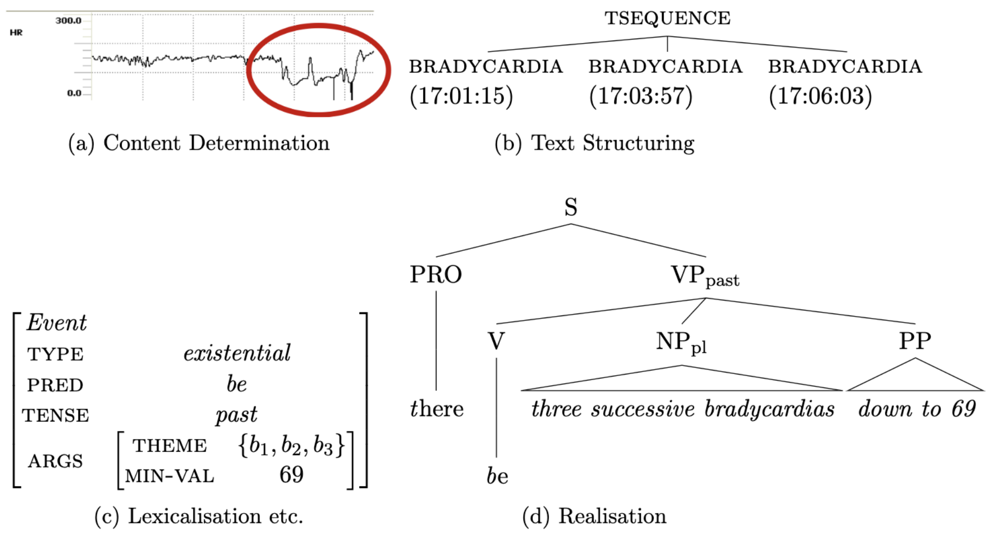


Figure 2.7: D2T subtasks (figure taken from Gatt and Krahmer (2018))

individual sentence plans that will form the final text summary. Finally, the linguistic realisation (d) happens resulting into final textual summary.

Most traditional methods take modular approach to D2T. They utilise domain specific rule engineering to build separate modules for initial subtasks and then templates or language grammars for the surface realiser (Goldberg et al., 1994; Portet et al., 2009; Sripada et al., 2003). However, there are some examples from pre-neural era that utilise statistical methods for some subtasks and templates for surface realisation (Adeyanju et al., 2010; Angeli et al., 2010; Kondadadi et al., 2013).

Recent neural methods to D2T, try to learn the domain specific and language generation rules using parallel corpora of training data. Neural methods, through encoder-decoder architectures, generate textual summaries without explicit templates or grammar. There are also both types of approaches: unified end-to-end, where a single sequence-to-sequence model is used for summary generation (Lebret et al., 2016; Puduppully et al., 2019b; Wiseman et al., 2017); as well as modular approaches, where different modules are used to solve different subtasks separately (Moryossef et al., 2019b; Puduppully and Lapata, 2021). However, in all cases, the final text generation in neural methods happens with a sequence-to-sequence.

In the next subsections, we will discuss the different approaches to D2T in literature. We divide our discussion in two broad categories: first, discussing traditional rule-based

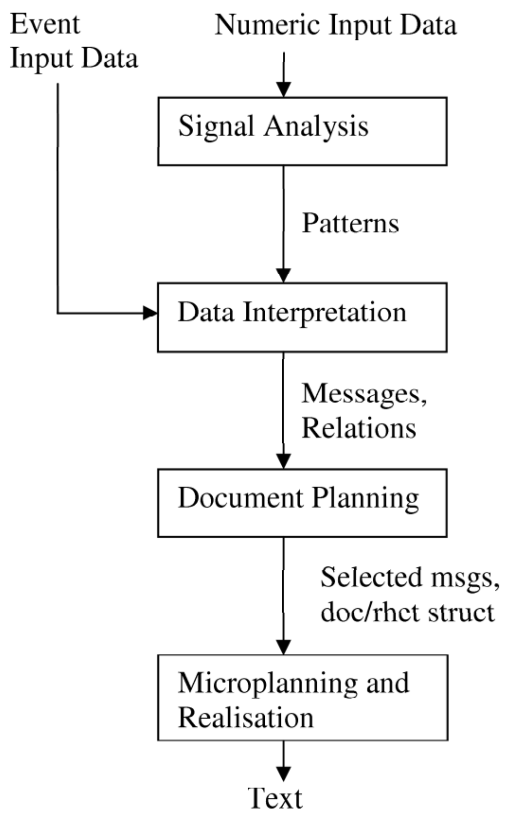


Figure 2.8: Traditional D2T architecture (figure taken from Reiter (2007a))

approaches which will also include some of the statistical approaches from pre-neural era; and second, the recent advancements in neural NLG for D2T.

2.3.1 Traditional D2T Methods

D2T research dates back to 1980s, Kukich (1983) build an automated system to generate natural language reports from databases in finance domain. This system applied a knowledge-based expert system to generate stocks report from stocks quote database. Following decades saw some commercial use of D2T research as well, mostly in weather reporting and some in health domain. Goldberg et al. (1994) presented a weather forecasting system, FoG, deployed in real-world that generated weather forecasts in two languages. Similar weather forecasting applications were developed by Sripada et al. (2003), focusing on special type of forecasts written for offshore engineers. There are few examples from health domain as well where D2T systems have been used for assisting doctors and nurses in communicating information to patients with natural language reports. Reiter et al. (2001) presented a D2T system used generate personalised smoking-cessation letters. Portet et al. (2009) built D2T system to generate textual reports from neonatal intensive care data tailored towards nurses and patient's family.

Most traditional D2T methods have been using a modular approach where several modules are used in a pipeline to solve the individual D2T subtasks. The most common architecture has been the one outlined in Reiter (2007a) which has four major steps (Figure 2.8):

- **Signal Analysis**: responsible for analysing the input data and finding patterns in them;
- **Data Interpretation**: responsible for using those identified patterns to detect domain-specific messages and causal relations between them;
- **Document Planning**: responsible for selecting and arranging the messages into a textual document which also includes deciding the overall structure of the output text; and
- **Surface Realisation**: responsible for generating the textual summary by coherently describing the selected messages in according to the document plan

While most works described above utilise engineered rules for each module, there have been some methods that utilise statistical approaches to learn some rules from data itself. These statistical methods also acknowledge the different subtasks and mostly use same modules to generate the summaries (Angeli et al., 2010; Konstas and Lapata, 2013; Liang et al., 2009).

Liang et al. (2009) present a hierarchical method for text generation using hidden semi-Markov models to first select the important facts from data and then describe them by selecting appropriate words. Angeli et al. (2010) utilise the alignment of facts and words from Liang et al. (2009) in a unified content selection and surface realisation model. They model the generation process into a series of local decisions and generate the final summary using templates. Konstas and Lapata (2013) use global context-free grammar to represent document plans and use them for text generation in an end-to-end manner.

Work	**Method**	**Approach**	**Data**
Lebret et al. (2016)	Seq2seq model bi-directional LSTM encoder and LSTM decoder	Unified end-to-end	Wikipedia infobox summaries
Wiseman et al. (2017)	Seq2Seq model with MLP encoder and LSTM with copy mechanism	Unified end-to-end	Basketball game summaries
Moryossef et al. (2019b)	Graph based content planner with seq2seq realiser	Modular	Wikipedia RDF summaries (Colin et al., 2016)
Puduppully et al. (2019a)	Content selection planning of table attributes with seq2seq model for surface realisation	Modular	Basketball game summaries (Wiseman et al., 2017)
Chen et al. (2020)	BiLSTM encoder with pre-trained GPT-2 decoder	Unified end-to-end	Meaning Representations, RDFs, Wiki infobox
Puduppully et al. (2019b)	BiLSTM encoder with entity-specific gate and LSTM decoder	Unified end-to-end	Basketball and baseball summaries
Rebuffel et al. (2020)	Seq2seq with hierarchical transformer encoder and LSTM decoder	Unified end-to-end	Basketball game summaries
Puduppully and Lapata (2021)	Separate macro planner for content selection and planning with LSTM-seq2seq surface realiser	Modular	Basketball and baseball summaries

Table 2.2: Some major neural-D2T methods

2.3.2 Neural D2T Methods

Initials works in using neural methods for D2T started by adapting an end-to-end setting. Mei et al. (2016) proposed a sequence-to-sequence framework using LSTM encoder and decoder for text generation from structured data. The model achieved 59% relative improvement compared to then state-of-the-art non-neural method on WeatherGov dataset (Liang et al., 2009). This achievement led to the exploration of developing better neural techniques and bigger training datasets for data-to-text generation. Authors in (Lebret et al., 2016) proposed a conditional language model to generate biographies of famous people from their Wikipdeia infobox. This method outperformed a template-based baseline with 15 BLEU score. Further improvements to this work applied structural biases in designing different encoders and decoders (Liu et al., 2018; Nema et al., 2018). Recent works applying transfer learning using massive pre-trained language models such as GPT2 (Radford et al., 2019) and T5 (Raffel et al., 2020) on this domain also shows huge improvement in generation quality while using just few hundred training samples (Chen et al., 2020; Kale, 2020).

Earlier datasets such as RoboCup and WeatherGov were not enough in terms of quantity and complexity to evaluate the performance of deep learning based neural networks. This led the development of shared tasks such as E2E (Novikova et al., 2017) and WebNLG (Colin et al., 2016) challenges. E2E challenge introduced a shared task to generate restaurant's descriptions based on the structured meaning representations, while WebNLG challenge introduced another dataset to generate entity descriptions from RDF triples. Initial deep learning based works to these challenges modelled the input data as a sequence of tokens, encoding them using LSTMs (Hochreiter and Schmidhuber, 1997) or Transformers (Vaswani et al., 2017), completely ignoring the underlying data structure (Dušek et al., 2018; Gardent et al., 2017). Although these methods were able to get good result, further improvements proved that exploiting the underlying data structure, by using special encoders like graph neural nets (Schlichtkrull et al., 2018), can be very beneficial (Beck et al., 2018; Liu et al., 2019a; Marcheggiani and Perez-Beltrachini, 2018; Zhao et al., 2020).

Despite showing improved performance in data-to-text generation, initial deep learning methods were able to generate only few sentence long documents. On the other-hand, real-world data-to-text generation tasks require longer document generations as well as selecting the important parts of input data (or content selection). Thus authors in (Wiseman et al., 2017) used a standard sequence-to-sequence model with copy mechanism to generate multiple lines game summary from NBA matches' box- and line- score data. They also open-sourced a data-to-text generation dataset known as RotoWire containing expert authored game summaries of NBA matches. The task was specially challenging because only a small portion from the input data needs to be selected in order to generate the summary. This allowed models to be evaluated in a better real-world scenario compared to previous ones, which only required text generation describing all the entities of input. In the same work, Wiseman et al. (2017) also demonstrated that despite the recent success of neural models in data-to-text generation, they failed to convincingly approximate the human summaries. Sometimes, they were outperformed by even simple templates methods in automated evaluations.

Improvements to this work again tried to encode different structural biases to the model, mainly in terms of tuning the encoder of seq2seq model, and embedding the entities in different manner, exhibiting better results (Puduppully et al., 2019b; Rebuffel et al., 2020). Puduppully et al. (2019b) presented an entity-focused model which learns a separate entity-focused representation for each entity in the data during training. This helps the model in updating the input records according to the entity they belong during generation process.

Rebuffel et al. (2020) presented a hierarchical model for D2T which uses a transformer model to encode the input data on two levels. On the first level, the transformer encoder encodes the entities separately using their attribute values. Then on next level, it encodes the pre-encoded entities through self-attention mechanism to create the final encoded representation of input table. This encoded input table is then used by a standard LSTM based decoder to generate the final text summary.

Works discussed above employed an unified architecture to text generation unlike the traditional modular approaches. However, unified neural methods often hallucinate by generate factually inaccurate textual summaries (Puduppully et al., 2019a; Wiseman et al., 2017). Recent works started using deep learning based systems with different components for different subtasks (Laha et al., 2020; Moryossef et al., 2019a,b; Puduppully et al., 2019a; Werlen et al., 2019). They mainly employed two modules: **content planning** - selecting, ordering and structuring the important information from input data; and **surface realisation** - generating fluent summary of the information in natural language. Having explicit components for various subtasks, as in traditional approaches, increases the adequacy and accuracy of deep learning models (Castro Ferreira et al., 2019; Moryossef et al., 2019a,b).

These pipeline systems (separating content planning from surface realisation) often utilise both hand-crafted rules and neural networks for the different components. In works (Laha et al., 2020; Moryossef et al., 2019a,b), authors adapted different rule based approaches to content selection and planning while completely separating text generation from content plan using neural models. The modules in these works are developed independently, following the traditional approach to data-to-text generation. On the other hand, Castro Ferreira et al. (2019); Puduppully et al. (2019a) separate the content planning phase from generation without sacrificing the end-to-end setting. This is done by sharing same loss function across two different encoder-decoder frameworks.

Puduppully et al. (2019a) presented a neural method with separate planning and selection from realisation phase. This model has a content planning module that learns to identify key attributes from the input data table and effectively organize them for presentation in the final summary. This selected and planned input is then presented a surface realization module to generate final textual summary, which is also a neural based sequence-to-sequence model.

Further additions in modular approaches to D2T included Puduppully and Lapata (2021) where a planning module is used to select and plan the input data on the entity level, referred as macro-plan. Each component of the macro-plan is referred as paragraph plan which corresponds to a sentence in the textual summary. This macro-plan is then used by a standard sequence-to-sequence surface realisation model for final text summary generation.

Although being scalable to different domains through training data, producing hallucination free accurate texts is still a challenging task with deep learning systems.

2.4 Context-Awareness in Data-to-Text Generation

In the previous sections, we discussed the advancements and current state-of-the-art in Data-to-Text Generation literature. We discussed two main types of systems for D2T, rule-based traditional systems, and learning-based neural systems. Both types of systems have their own benefits and drawbacks where rule based systems produce accurate but non-diverse texts, neural systems produce fluent and diverse texts but hallucinate with inaccurate generations (Pauws et al., 2019).

Human written D2T summaries contain different types of information. The easier types are the ones which can be copied or derived from just the current event's data. However, in practice much of the summaries content is derived from two types of contextual information: first, temporal context, where the context from past events is needed to derive the information in current events summary; and second, environmental context, where the general knowledge of the world where events take place is used to influence the content of the summary. Current D2T systems, lack such contextual awareness in their system design. This makes the system generated summaries less insightful and less interesting for the end-user compared to their human-written counterparts.

There has been some work that consider the temporal aspects of event based D2T problems. Few earlier works, both in neural as well as rule based systems, have tried to include some form of temporal context in final event summary (Gong et al., 2019a; Robin and

McKeown, 1996). Robin and McKeown (1996) presented a revision-based method for surface realisation that iteratively revises D2T summaries with more information. A type of the information they used for revision was 'historical background' which can be considered as the temporal context. However, they mainly focus on building the revision method with some pre-calculated temporal information as one of the scenarios.

Gong et al. (2019b) suggest that taking the third dimension as time into account can be helpful for generation from complex time-series data. They used a multi-dimensional encoder to encode the records from previous events and then use a standard LSTM decoder to generate the text. However, adding a new dimension doesn't help neural systems as they now need to process even more numerical data and neural methods find arithmetic operations very challenging (Nie et al., 2018). Wang (2019) also identify the presence of temporal information in D2T summaries, but in order to measure the impact of better alignment between input data and output summary, they remove the parts of summary that contain such temporal information. This type of approach, whilst producing more accurate summaries, will be deemed ecologically invalid as they do not reflect the content of their human-authored counterparts (de Vries et al., 2020; Thomson et al., 2020b).

There has been some study in D2T to incorporate end-user information in order to tailor summaries for their requirements (Gkatzia et al., 2014; Portet et al., 2009; Reiter et al., 2001). STOP system presented in Reiter et al. (2001) was used to write personalised summaries to create awareness against smoking. The system used a questionnaire from end-users to personalise the final summary accordingly. In similar approach, the BT project (Hunter et al., 2011; Mahamood and Reiter, 2011) generated multiples summaries of same events in Neonatal Intensive Care Unit (NICU) domain. Hunter et al. (2011) generate the summaries for nurses while Mahamood and Reiter (2011) generate the summary for parents of infants in order to help manage their stress-level. Gkatzia et al. (2014) present a multi-adaptive approach to summary generation using Principal Component Regressor (PCR) that can adapt its content according to the user preference in time-series data.

However, in many cases, the environment in which the events take place can have an impact on the summary content as well. For example, a sports summary for a game towards the end of the season might decide to include more comparisons of players performance over the season and teams chances of qualifying for play-offs. Similarly, a weather report for Aberdeen, UK might describe 20 degrees temperature in June as normal while same temperature near equator in June will be considered unusual. This suggests that general knowledge of the world such as the time or location of events can have a major impact on the summary content.

Some previous works have identified the need of such external knowledge in D2T and have tried to include them in system development (Chen et al., 2019; Thomson et al., 2020b). Thomson et al. (2020b) identify that D2T summaries in basketball domain often mention the names of city and stadium where game takes place. They collected such information from external sources [10] and include them in the event data, improving the accuracy of system generated summaries considerably. Similarly, Chen et al. (2019) demonstrate external knowledge can improve summaries in biography domain. Apart from parallel input data and output summary, they use entity-specific knowledge collected from Wikipedia to improve the alignment, in turns improving system generated summaries.

In this thesis, we present methods for utilising both temporal and environmental context in the summary generation process. The environmental context is used to produce a better content plan that is more close to human written summaries as compared to neural benchmarks. The content plan presents an organisation of summary content in terms of different types of information to be described in each sentence. While the temporal context is utilised by the surface realisation module to include the important data selected from the previous events in the final event summary.

2.5 Case-Based Reasoning

Case Based Reasoning (CBR) is a sub-field of AI problem solving that works on the principle of "similar problems have similar solution" (Aamodt and Plaza, 1994; Richter and Weber, 2013). This is synonymous to humans everyday problem solving where to solve a new target problem, solution of similar problems from past experiences are reused and adapted.

CBR systems work in a four-step process known as 4Rs of CBR, as explained here:

1. **Retrieve**: retrieve the problems similar to target problem from case-base;
2. **Reuse**: reuse the solution of these retrieved problems to propose a new solution for the target problem;

[10] `https://www.basketball-reference.com/`

3. **Revise**: revise the proposed solution to adapt to the new target problem; and
4. **Retain**: finally retain the revised solution with its respective problem in the case-base for future use;

CBR systems use four knowledge containers to store the knowledge of world and domain that is used for problem solving. The four containers are:

1. **Case-Base**: Case-Base (CB) acts as the database of a CBR system by storing previous experiences in the form of cases. Each case has two components: problem-side, used to represent the previous problems; and solution-side, used to represent the solution to that problem.
2. **Vocabulary**: Vocabulary of the CBR system is used to represent the problem-side of a case. In terms of a classification problem, the vocabulary can be considered as the features of the samples.
3. **Similarity Metric**: Similarity metric is used to measure the similarity between the target problem and previous cases for retrieval.
4. **Adaptation Knowledge**: Adaptation knowledge is used to modify the proposed solution according to the specific needs of target problem.

A CBR system works by storing previous experiences in CB. When a new target problem arises, the CBR system retrieves most-similar cases using the similarity metric and then uses their solution to propose a new solution. This solution is then adapted accordingly and also retained in the case-base. The full CBR process is shown in Figure 2.9. In a CBR system design, it is also possible to trade-off knowledge between the four containers (Ganesan and Chakraborti, 2018).

CBR methods have also been used in many D2T works (Adeyanju et al., 2010; Dubey et al., 2018; Kondadadi et al., 2013; van der Lee et al., 2018). The core-idea revolves around building a case-base of templates and using CBR process to dynamically select appropriate templates for textual summary. The problem-side representation of a case in the case-base is either full or part of input data and solution-side is a template with placeholders to describe the data. Different methods are used to retrieve and arrange the templates in order to generate the final summary.

Adeyanju et al. (2010) use a CBR approach to align templates of review text with a likert scale rating in order to generate a textual review of hotels. Kondadadi et al. (2013)

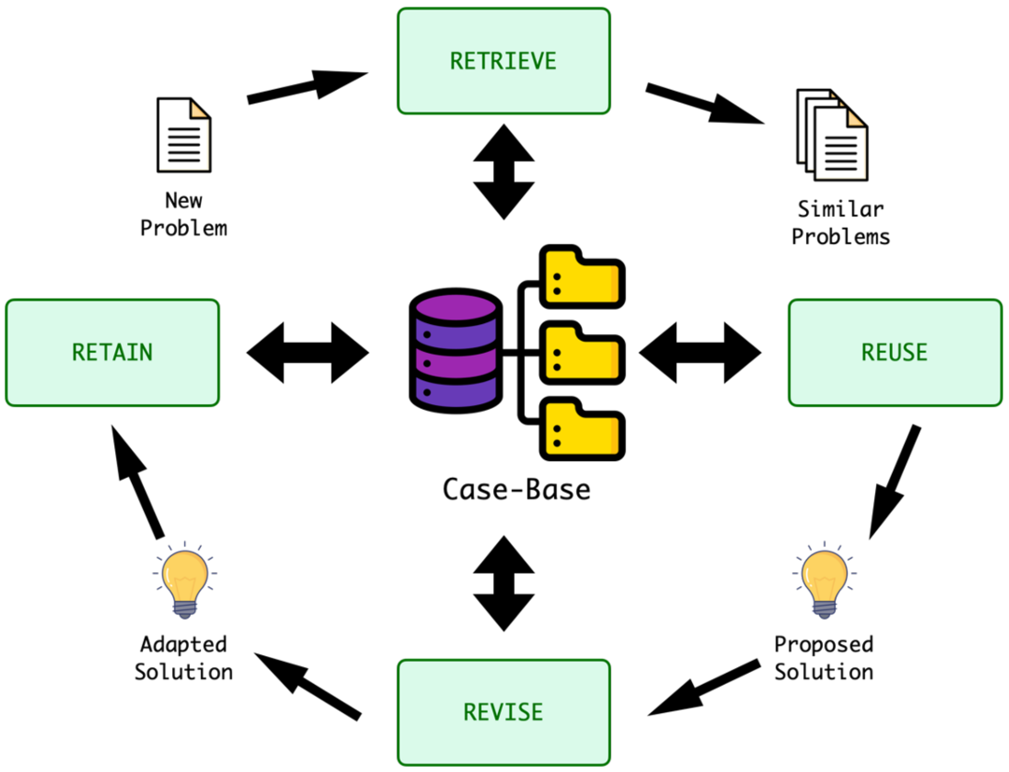


Figure 2.9: CBR system working cycle (Aamodt and Plaza, 1994)

propose an aggregated planning and realization method to dynamically select templates to generate textual summaries in weather and biography domains. van der Lee et al. (2018) study different methods of learning to either retrieve or generate templates (using neural sequence-to-sequence method) to generate weather and sports reports.

However, most approaches have focused on smaller and less complex datasets with few sentence summaries. Use of CBR methods for dynamically selecting and organising templates in complex domains with longer summaries is majorly unexplored. This could be because of the uneven alignment between attributes of input data table and output summary sentences. In this thesis, we propose a CBR method to dynamically select a set of templates and organise them to produce an event summary in a complex sports domain where average summary length is 15 sentences.

2.6 Conclusions

This chapter reviewed the literature around different types of D2T methods and contextual awareness in D2T. Some other key topics such as CBR and advancements in neural-NLP are also discussed briefly.

We identified how traditional approaches break the process of D2T summary generation into several key subtasks. A great deal of manual effort is always involved in building components for these subtasks. Neural methods aim to reduce this manual effort by offering to learn the subtasks from training data. Both unified end-to-end and pipeline-based modular approaches with neural methods are evident in literature, with latter often offering better quality of output and control over process. But the neural methods still struggle with hallucinations, which is still prevalent with larger models.

In this thesis, we build a modular approach for D2T. Separate modules are build for: Content Planning (Chapter 6), that uses environmental context to provide the higher-level organisation of a summary in terms of different types of content to discuss (identified in Chapter 5); Content Selection, that selects important data from previous events (Chapter 7); and Surface Realisation, which uses a neural method to produce an initial draft of summary and then post-edits them using dynamically selected templates (Chapter 7). The dynamic template method provides a semi-automated way to commission high quality templates and uses a CBR method to dynamically retrieve appropriate templates based on the event's input data. This system is separately evaluated (Chapter 4) before using it in post-editing of neural generated summary.

In the next chapter, we will discuss different publicly available D2T datasets and evaluation metrics used in literature for assessing the quality of system generated D2T summaries.

Chapter 3

Datasets and Evaluation

In this chapter, we will provide a background on standard D2T datasets publicly available in the research field. We also discuss the various evaluation methodologies adopted in the field.

3.1 Formalizing D2T Concepts

D2T literature showcases a variety of datasets representing different types of real-world problems. These datasets are a collection of events with a textual summary written for each. An event is a time-frame or activity which is represented by structured data, mostly numeric and in tabular form. In sports domains, a game can be considered a single event where the event is represented by the statistics of players & teams performance along with a textual summary written to describe the event. Similarly, in weather domain, the time-frame for which the weather report is written would be considered an event.

These D2T datasets can be distinguished into different categories based on the relation between their events. In stand-alone D2T applications, the events are time-stamped and occur in a sequence. For example, games in a league will follow a temporal pattern. On the other hand, D2T datasets are also used in general-purpose LLM benchmarking or building a module in an AI pipeline such as ChatBots. For example, an answering module in a chatbot can use a D2T system to formulate the answer in textual form. Such datasets contain events that are randomly arranged and are not related to each other (Gardent et al., 2017; Novikova et al., 2017).

Within datasets with time-stamped events, there can be two types. First are the type

TEAM	WIN	LOSS	PTS	REB	AST	...
Pacers	4	6	99	40	17	...
Celtics	5	4	105	47	22	...

PLAYER	H/V	AST	REB	PTS	CITY	...
Myles Turner	H	1	8	17	Indiana	...
Thaddeus Young	H	3	8	10	Indiana	...
Isaiah Thomas	V	5	0	23	Boston	...
Kelly Olynyk	V	4	6	16	Boston	...
...						

The Boston Celtics defeated the host Indiana Pacers 105-99 at Bankers Life Fieldhouse on Saturday. It was the second victory over Pacers for the Celtics this season after emerging victorious in Boston 91-84 on Nov. 16. ... Isaiah Thomas led the team in scoring, totaling 23 points and five assists on 4–of–13 shooting. Kelly Olynyk got a rare start and finished second on the team with his 16 points, six rebounds and four assists. ... Boston will return to action on Monday against the New Orleans Pelicans.

Figure 3.1: Excerpt of a human authored D2T summary with input data from sports domain (basketball)

of datasets whose summaries require contextual awareness, i.e, the summary of an event may need some external information other than the event's input data (Thomson et al., 2020a). This external information can be: either temporal context - information from past or concurrent events; or environmental context - information about the general world in which event takes place. Second types of datasets are the one which do not require any contextual information to write the event summary, ie, the event's input data contains all the necessary information for writing the event summary (Sripada et al., 2003) (also described in Figure 3.2).

In this thesis, we focus on the time-stamped event based D2T problems and thus study the datasets that present these types of problems. More specifically, we study the event based D2T problems with contextual awareness and provide contributions to improve the event summary's quality in terms of improving their contextual awareness. An example event from an event-based D2T dataset, with input data and output summary as shown in Figure 3.1. We formalize event-based D2T problems with the following notations.

A data instance in an event-based **D2T dataset** $\mathcal{DB}$ is an **event** $\mathcal{E}_i$ with a **data structure** $\mathcal{D}_i$ for which a **textual summary** $\mathcal{S}_i$ is written summarising the insights and information of the event. A data structure consists of multiple **entities** $\mathcal{O}_{i,o}$ that are the objects involved in the event, and each entity is described by multiple **features** $\mathcal{F}_{i,o,f}$ which are the attributes of those entities.

$$\mathcal{DB} = [\mathcal{E}_{i-e}, \cdots, \mathcal{E}_i, \cdots, \mathcal{E}_{i+e}] \tag{3.1}$$
$$\mathcal{E}_i = \{\mathcal{D}_i, \mathcal{S}_i\} \tag{3.2}$$
$$\mathcal{D}_i = \{\mathcal{O}_{i,1}, \mathcal{O}_{i,2}, \cdots, \mathcal{O}_{i,o}\} \tag{3.3}$$
$$\mathcal{O}_{i,o} = \{\mathcal{F}_{i,o,1}, \mathcal{F}_{i,o,2}, \cdots, \mathcal{F}_{i,o,f}\} \tag{3.4}$$

When building a text generation system g for a D2T task, the summary of an event is the function of the current event as well as other events in stream:

For problems that do not require contextual awareness:

$$\mathcal{S}_i = g(\mathcal{D}_i) \tag{3.5}$$

For problems that require contextual awareness:

$$\mathcal{S}_i = g(\mathcal{D}_i, \mathcal{DB}) \tag{3.6}$$

A value $\mathcal{R}_{i,o,f}$ will be recorded for each feature $\mathcal{F}_{i,o,f}$ of an entity $\mathcal{O}_{i,o}$ that can be considered a record in the data structure. So if an input data structure is flattened into a sequence of records (mostly for training a neural generation system), it looks as:

$$\begin{multline} \mathcal{D}_i = \{\mathcal{R}_{i,1,1}, \mathcal{R}_{i,1,2}, \cdots, \mathcal{R}_{i,1,f}, \cdots \\ \mathcal{R}_{i,2,1}, \mathcal{R}_{i,2,2}, \cdots, \mathcal{R}_{i,2,f}, \cdots \\ \mathcal{R}_{i,o,1}, \mathcal{R}_{i,o,2}, \cdots, \mathcal{R}_{i,o,f}\} \end{multline} \tag{3.7}$$

3.2 Datasets

A taxonomy of different D2T datasets is shown in Figure 3.2. As described above, this thesis focuses on event-based D2T datasets and thus the four common event-based D2T

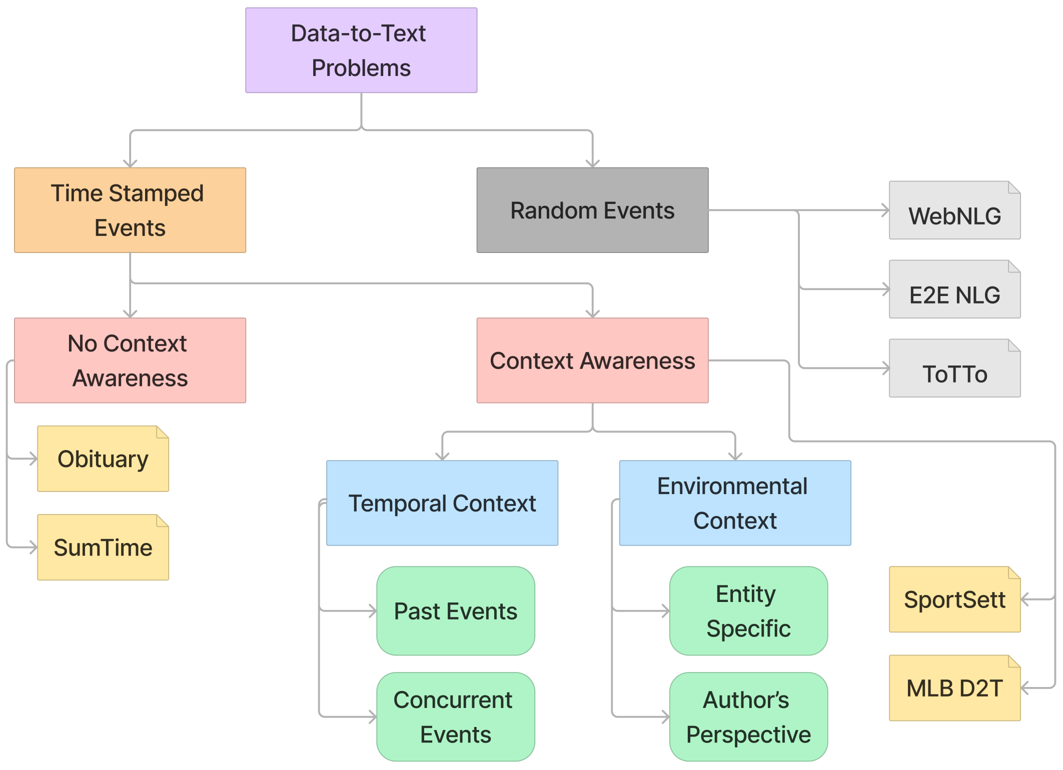


Figure 3.2: D2T Datasets: yellow and grey leaf boxes are several datasets from literature

datasets are described below. The basic statistics of these datasets is shown in Section 3.2.

3.2.1 Obituary

This dataset contains a sample of 850 obituaries aligned with their personal information. In this dataset, an Obituary (or a death) is the event; where the deceased person is the entity; and information related to their personal life and funeral are the different features. The dataset contains 80/20/20 obituaries in train/valid/test set respectively. This dataset is curated as part of this thesis, where the obituaries were scrapped from a funeral website in UK [1]. The structured information from these obituaries about the deceased person is manually extracted by authors. This structured information acts as the input data for the event, while the obituary is the output summary. An example event from Obituary dataset is shown in Figure 3.3 [2].

[1] https://funeral-notices.co.uk/
[2] Dataset available at https://github.com/ashishu007/Obituary-Generation

	Obituary	SumTime	SportSett	MLB
Train set	80	793	3690	22821
Valid set	20	99	1230	1739
Test set	20	100	1230	1744
Avg. Input Attributes	20	500	500	600
Avg. Sentences	3	12	15	20

Table 3.1: Statistics of different datasets

Feature	Value
Name	Joya Duncan-Parker
Demise Date	27th August 2018
Demise Place	home
Nick Name	Joyce Parker
Age	69 years
Home Town	Clarencefield
Gender	Female
Children	Paul and Lee
Funeral Guest List	Private

DUNCAN-PARKER JOYA On the 27th August 2018, suddenly at home, Joya Duncan-Parker (Joyce Parker), aged 69 years, of Clarencefield, formerly of Dumfries, loving mum of Paul and Lee, and a much loved nana. Funeral Private.

Figure 3.3: Obituary dataset: input structured data about deceased person's life on the left, and corresponding textual output (Obituary) on the right

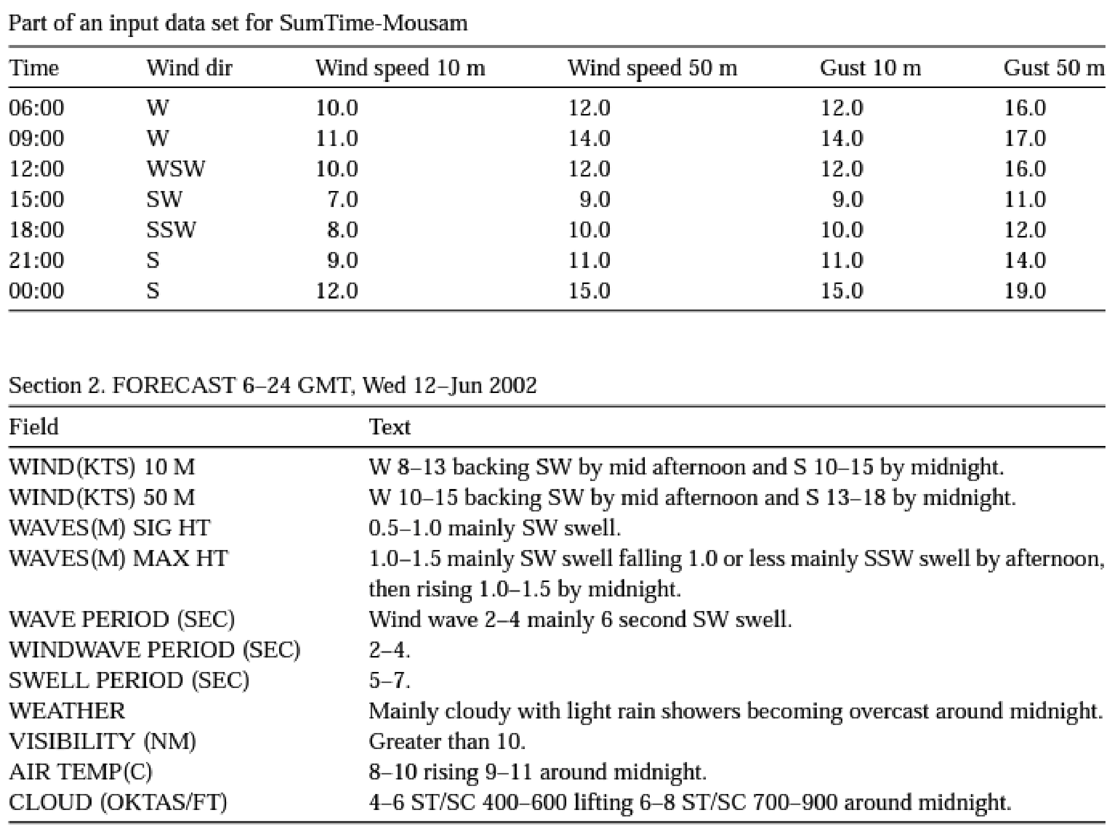
3.2.2 SumTime

This dataset contains human written weather forecasts for a day written for oil and gas offshore engineers in Aberdeen, UK. The forecasts are usually written from two types of NWP data: wave data; and mmo data (please refer to Sripada et al. (2003) for a detailed discussion on the data organisation). The representation of SumTime on different dimensions is as follows: each sample in the dataset covers a forecast during 12-hours time-period (AM forecast or PM forecast). The time period for which the forecast is written is considered an event; the different entities are the elements described in the forecast such as wind or wave; and the hours of the day for which the readings are taken for those elements are the features. The total number of samples in train/valid/test sets are 793/99/100. An example event from SumTime dataset is shown in Figure 3.4.

3.2.3 SportSett

This dataset contains box- & line- scores from NBA (basketball) games aligned with human-written summaries describing the respective game [3]. The games are from season

[3]collected from www.rotowire.com



Part of an input data set for SumTime-Mousam

Time	Wind dir	Wind speed 10 m	Wind speed 50 m	Gust 10 m	Gust 50 m
06:00	W	10.0	12.0	12.0	16.0
09:00	W	11.0	14.0	14.0	17.0
12:00	WSW	10.0	12.0	12.0	16.0
15:00	SW	7.0	9.0	9.0	11.0
18:00	SSW	8.0	10.0	10.0	12.0
21:00	S	9.0	11.0	11.0	14.0
00:00	S	12.0	15.0	15.0	19.0

Section 2. FORECAST 6–24 GMT, Wed 12–Jun 2002

Field	Text
WIND(KTS) 10 M	W 8–13 backing SW by mid afternoon and S 10–15 by midnight.
WIND(KTS) 50 M	W 10–15 backing SW by mid afternoon and S 13–18 by midnight.
WAVES(M) SIG HT	0.5–1.0 mainly SW swell.
WAVES(M) MAX HT	1.0–1.5 mainly SW swell falling 1.0 or less mainly SSW swell by afternoon, then rising 1.0–1.5 by midnight.
WAVE PERIOD (SEC)	Wind wave 2–4 mainly 6 second SW swell.
WINDWAVE PERIOD (SEC)	2–4.
SWELL PERIOD (SEC)	5–7.
WEATHER	Mainly cloudy with light rain showers becoming overcast around midnight.
VISIBILITY (NM)	Greater than 10.
AIR TEMP(C)	8–10 rising 9–11 around midnight.
CLOUD (OKTAS/FT)	4–6 ST/SC 400–600 lifting 6–8 ST/SC 700–900 around midnight.

Figure 3.4: SumTime dataset: the structured data with weather forecast for the day on top, while corresponding weather report in textual format at the bottom (Sripada et al., 2003)

2014 to 2018, out of which 2014, 2015 & 2016 are used as train split, 2017 for valid split, and 2018 for test split. The number of samples in train/valid/test sets is 4745/1228/1229 respectively. Each game in the dataset is an event; the players teams from the game are entities; and stat-types are the features. An example event from SportSett dataset is shown in Figure 3.5 [4].

3.2.4 MLB

This dataset contains stats from MLB games aligned with their summaries written by human authors [5]. Each sample in the dataset contains the box score and play-by-play record of the of the game on the input side, which is aligned with a textual summary of

[4]Dataset available at https://huggingface.co/datasets/GEM/sportsett_basketball

[5]collected from www.espn.com

TEAM	WIN	LOSS	PTS	REB	AST	...
Pacers	4	6	99	40	17	···
Celtics	5	4	105	47	22	···

PLAYER	H/V	AST	REB	PTS	CITY	···
Myles Turner	H	1	8	17	Indiana	···
Thaddeus Young	H	3	8	10	Indiana	···
Isaiah Thomas	V	5	0	23	Boston	···
Kelly Olynyk	V	4	6	16	Boston	···
...						

The Boston Celtics defeated the host Indiana Pacers 105-99 at Bankers Life Fieldhouse on Saturday. It was the second victory over Pacers for the Celtics this season after emerging victorious in Boston 91-84 on Nov. 16. ... Isaiah Thomas led the team in scoring, totaling 23 points and five assists on 4–of–13 shooting. Kelly Olynyk got a rare start and finished second on the team with his 16 points, six rebounds and four assists. ... Boston will return to action on Monday against the New Orleans Pelicans.

Figure 3.5: SportSett dataset: the structured data containing game statistics on the left, with corresponding game summary excerpt in textual format on the right

around 20 sentences long. In this dataset, games are considered as the events; players, teams and the plays as entities of the event; and different stat-types as the features of an entity. The dataset is split for train/valid/test sets, containing 22821/1739/1744 samples each. An example event from MLB dataset is shown in Figure 3.6 [6].

3.3 Evaluation

Evaluation of any text generation system is a non-trivial and challenging task. There are many dimensions for assessing the quality of automatically generated text. Based on the domain and end-user requirements, a dimension may be more important than other. However, a paramount requirement of the D2T summary is being accurate and providing faithful information which is free of any unsupported claims.

Other dimensions of evaluation include text fluency and diversity. Diversity can be both lexical and content-based where the system is not just expected to use different vocabulary but also present different insights. Taking an example from sports domain, a D2T system is expected to produce event summaries that discuss multiple aspects of the game without repeating only a few things in all of their summaries. This is usually referred as the content selection ability of the system which measures how well the system can identify and select important information from the input data.

Apart from these, aspects such as content ordering is also important which translates as the ability of a system to order the information presented in the summary. Another

[6] Dataset available at https://huggingface.co/datasets/GEM/mlb_data_to_text

TEAM	Inn1	Inn2	Inn3	Inn4	...	R	H	E	...
Orioles	1	0	0	0	...	2	4	0	...
Royals	1	0	0	3	...	9	14	1	...

BATTER	H/V	AB	R	H	RBI	TEAM	...
C. Mullins	H	4	2	2	1	Orioles	...
J. Villar	H	4	0	0	0	Orioles	...
W. Merrifield	V	2	3	2	1	Royals	...
R. O'Hearn	V	5	1	3	4	Royals	...
...	...	...	...	...	...	...	

PITCHER	H/V	W	L	IP	H	R	ER	BB	K	...
A. Cashner	H	4	13	5.1	9	4	4	3	1	...
B. Keller	V	7	5	8.0	4	2	2	2	4	...
...	...	...	...	...	...	...				

KANSAS CITY, Mo. – Brad Keller kept up his recent pitching surge with another strong outing. Keller gave up a home run to the first batter of the game – Cedric Mullins – but quickly settled in to pitch eight strong innings in the Kansas City Royals' 9–2 win over the Baltimore Orioles in a matchup of the teams with the worst records in the majors. ··· Ryan O'Hearn homered among his three hits and drove in four runs, Whit Merrifield scored three runs, and Hunter Dozier and Cam Gallagher also went deep to help the Royals win for the fifth time in six games on their current homestand. ··· The Royals answered in the bottom of the inning as Gallagher hit his first home run of the season...

Figure 3.6: MLB dataset: the structured data containing game statistics on the left, with corresponding game summary excerpt in textual format on the right

important aspect of D2T systems is the distribution of different types of content produced in their summaries. That is, the summaries should contain similar amount of contextually aware information as in human authored summaries.

In the next subsections, we will define the automated metrics and human judgement methods used for evaluating content accuracy, selection, ordering and diversity. Next chapter (Chapter 5) defines a method for measuring the distribution of content types in summaries produced by different D2T systems.

3.3.1 Accuracy

Automated Metrics

The most common method of automated accuracy evaluation of generated texts is the family of extractive evaluation (EE) metrics proposed by Wiseman et al. (2017). The EE method uses a standard information extraction model to extract relation-tuples from the system generated summaries. These relation tuples are then matched against the input data as well as similar relation-tuples extracted from human authored summaries.

Each relation-tuple extracted from textual summaries contain three elements are in the form of: $< \mathcal{E}, m, \mathcal{F} >$, where $\mathcal{E}$ and m are the name of an entity and numerical value extracted from the text respectively, while F is the predicted relation between the entity and the value. For example, consider the first sentence from the summary in Figure 3.1.

We can extract the entity '*Boston Celtics*' and the value '*105*'. The information extraction can predict the relation between them as '*PTS*'.

Several such relation-tuples are then extracted from the textual summaries (both human authored and system generated) and compared to calculate the following metrics:

- **Relation Generation**: measures the ability of text generation systems to produce summaries that are faithful to the input data. This is measured as the precision of relation tuples extracted from generated summaries matched against the input data.
- **Content Selection**: measures the ability of systems to select content closer to human authors. This is measured as the precision and recall between the relation tuples extracted from system generated summaries and the tuples extracted from human-authored summaries.
- **Content Ordering**: measures the ability of systems to order the content of summaries which is measured as the Normalized Damerau-Levenshtein Distance.

Human Judgement

Despite the efficiency of automated metrics for accuracy evaluation they fall short on several occasions. Automated metrics, especially the trained ones are also limited to the goodness of the learned model. EE metrics being trained for IE tasks are prone to inaccuracies themselves and thus human judgement is still preferred to validate the findings of automated evaluations.

Thus, in this thesis, we also use human judgement to evaluate the system generated summaries. Following most work in the literature, we use the count method to evaluate accuracy of summaries, which counts number of supporting and contradicting claims from a sentence Puduppully et al. (2019a); Rebuffel et al. (2020); Wiseman et al. (2017). For example, in the first sentence of Figure 3.1, there are at-least two claims made: Boston Celtics scored 105 points; and Indiana Pacers scored 99 points. The human evaluator, when provided with the correct input data, can verify if these claims are supported by the input data or contradict it. Thus the task for the human evaluator becomes to count the number of such supporting and contradicting claims from the generated summaries. These statistics about average number of supporting and contradicting claims can be used to asses the accuracy capability of a generation system.

Thomson et al. (2023) propose a gold standard methodology for evaluating accuracy in D2T. This method presents a typology of different accuracy errors in D2T summaries. The error categories are:

- **Name**: incorrect people, place, organisation names including days of the week
- **Number**: incorrect numbers, also includes spelled out digits
- **Word**: an incorrect word or phrase not belonging to above
- **Context**: incorrect inference caused by a word or phrase due to its surrounding context
- **Not Checkable**: a statement that cannot be checked because its time-consuming or proper information is unavailable
- **Other**: any other inaccuracy which doesn't belong to above (for example, illogical phrases)

In this method, human annotators are asked to annotate the inaccurate generations in a summary and assign them one of these six categories. However, this method is very resource intensive and takes about 30-45 minutes to annotate one basketball summary (Thomson and Reiter, 2020). We still use this method for evaluating both system generated and human authored D2T summaries from different datasets of various domains in Chapter 5.

3.3.2 Fluency

Another important expectation of D2T systems is to generate summaries which are well-written in terms of natural, human-readable language. Thus its important for system generated summaries to: be grammatically correct; and have a natural flow of information in them. For this, we use the following metrics to evaluate.

Automated Metrics

There are multiple automated metrics that measure the fluency of generated summaries. Most of these are based on the n-gram overlap of the system generated summaries and human authored summaries. These metrics require atleast one human authored reference summary per system generated summary to perform the evaluation. Most of these metrics evolved in machine translation research and have been adopted in the NLG literature as well.

BLEU short for Bilingual Evaluation Understudy (Papineni et al., 2002), is a precision oriented metric that compares the n-gram overlap between candidate output and a reference to produce a score between 1 to 100. It is defined as the precision of overlapping n-grams as the ratio of the number of overlapping n-grams to the total number of n-grams in the candidate output. This n-gram precision is calculated for 1- to 4-grams which are combined with the following formula:

$$BLEU = BP \cdot exp\left(\sum_{n=1}^{N} W_n \log p_n\right) \tag{3.8}$$

where W_n are the weights of the different n-gram precisions, such that $\sum_{n=1}^{N} W_n = 1$, p_n is the precision for each n-gram, and BP is the brevity penalty which is used to penalise the shorter candidate outputs.

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation Lin (2004), measures the recall of n-gram overlaps between candidate output and reference; and also provides a score between 1 to 100 like BLEU.

$$\text{ROUGE} = \frac{\sum\limits_{s_r \in \text{references}} \sum\limits_{n\text{-gram} \in s_r} Count_{overlap}(n\text{-gram})}{\sum\limits_{s_r \in \text{references}} \sum\limits_{n\text{-gram} \in s_r} Count(n\text{-gram})} \tag{3.9}$$

METEOR or Metric for Evaluation of Translation with Explicit Ordering, measures a *relaxed*-F score between the n-gram overlaps of candidate output and reference. It first does an exact word mapping, followed by stemmed word mapping, and finally paraphrase mapping to compute the F-score as:

$$P(Precision) = \frac{\#mapped\_ngrams}{\#ngrams\_in\_candidate}, R(Recall) = \frac{\#mapped\_ngrams}{\#ngrams\_in\_reference} \tag{3.10}$$

$$Fscore = \frac{10PR}{R + 9P} \tag{3.11}$$

It also applies a penalty to reward longer contiguous overlaps and calculates the final

METEOR-score as follows:

$$\text{Penalty} = 0.5 * \left[\frac{\#chunks}{\#unigrams\_matched}\right]^3 \tag{3.12}$$

$$\text{METEOR Score} = Fscore * (1 - Penalty) \tag{3.13}$$

chrF++ is a character-based n-gram overlap metric which measures the F-score between the character-, unigram- and bigram- based overlap of candidate output with reference as follows:

$$chrF_\beta = (1 + \beta^2)\frac{chrP.chrR}{\beta^2.chrP + chrR} \tag{3.14}$$

where β indicates the more weightage given to recall than precision, $chrP$ is the overlap precision, and $chrR$ is the overlap recall.

BERT-Score measures the cosine similarity between the tokens of candidate output and reference texts. These tokens are embedded using a pre-trained language model such as BERT and uses a greedy matching approach to calculate the F-score as follows:

$$\text{BERTscore} = F_{BERT} = 2\frac{P_{BERT}.R_{BERT}}{P_{BERT} + R_{BERT}} \tag{3.15}$$

$$R_{BERT} = \frac{1}{|r|}\sum_{i \in r} \max_{j \in p} \vec{i}^T \vec{j} \ , P_{BERT} = \frac{1}{|p|}\sum_{j \in p} \max_{i \in r} \vec{i}^T \vec{j} \tag{3.16}$$

Human Judgement

We also conduct human evaluations to measure the fluency of generated texts. This was done by presenting a few pairs of system generated summaries to some human evaluators and asking them to pick the best out of two on the following three dimensions:

- **Grammaticality**: is the generated summary written in well-formed, grammatically correct English?

- **Coherence**: does the summary have a natural ordering of information and seems properly structured?
- **Conciseness**: are the unnecessary repetitions, of words, phrases or sentences, avoided?

3.4 Conclusion

In this chapter, we discussed different types of real-world D2T problems and publicly available datasets representative of them. We differentiate them: first based on the relation between their events, and then based on the contextual awareness in their summaries. We also discussed the common practice for evaluating D2T systems in literature and define the automated metrics and human judgement methods used for evaluation.

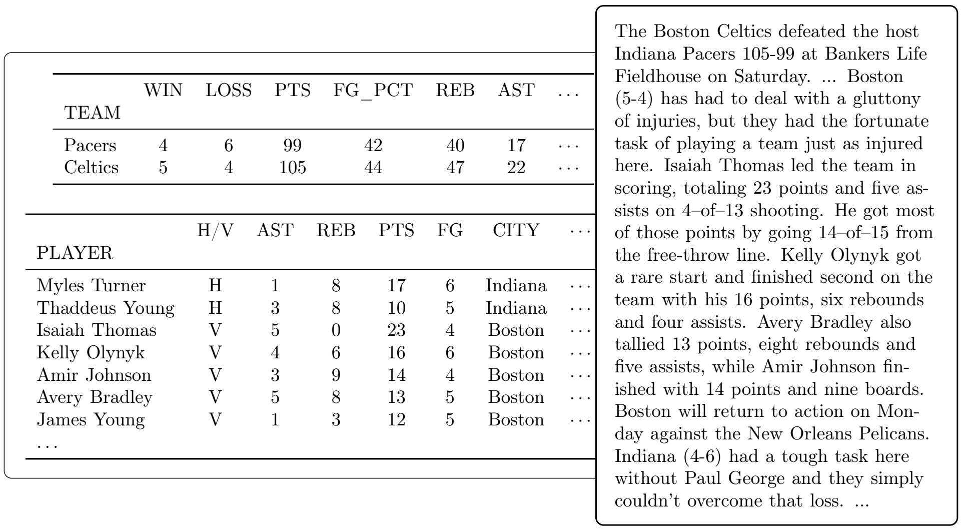
Chapter 4

Case Based Approach to Data-to-Text Generation

There are mainly two approaches to building a Data-to-Text Generation system: first, traditional rule-based, which uses a set of templates and rules to generate the summaries; and second, recent neural-based, which use a learning approach to learn the task of summary generation from historical data. Rule-based systems produce accurate but monotonous summaries, while neural systems are good fluent and diverse but inaccurate summaries. In this chapter, we propose a CBR approach to Data-to-Text Generation that mitigates the issue of accuracy vs diversity by dynamically selecting templates to write an event summary based on the knowledge of previous events.

4.1 Motivation

Neural systems demonstrate greater fluency and diversity in generated textual summaries but often *hallucinate* by producing inaccurate information that is not supported by the input data. One of the main reasons for hallucination is that the systems have to learn multiple domain specific rules to make sense of input data as well as learn how to verbalise that data (Puduppully et al., 2019a). Rule-based systems however are able to produce high quality texts in terms of accuracy but at the expense of diversity in the generated texts. Although, in real-world data-to-text applications, accuracy is usually much more important than fluency and diversity (Dale, 2023), often making rule-based systems the preferred choice in real-world applications.



TEAM	WIN	LOSS	PTS	FG_PCT	REB	AST	...
Pacers	4	6	99	42	40	17	...
Celtics	5	4	105	44	47	22	...

PLAYER	H/V	AST	REB	PTS	FG	CITY	...
Myles Turner	H	1	8	17	6	Indiana	...
Thaddeus Young	H	3	8	10	5	Indiana	...
Isaiah Thomas	V	5	0	23	4	Boston	...
Kelly Olynyk	V	4	6	16	6	Boston	...
Amir Johnson	V	3	9	14	4	Boston	...
Avery Bradley	V	5	8	13	5	Boston	...
James Young	V	1	3	12	5	Boston	...
...							

The Boston Celtics defeated the host Indiana Pacers 105-99 at Bankers Life Fieldhouse on Saturday. ... Boston (5-4) has had to deal with a gluttony of injuries, but they had the fortunate task of playing a team just as injured here. Isaiah Thomas led the team in scoring, totaling 23 points and five assists on 4–of–13 shooting. He got most of those points by going 14–of–15 from the free-throw line. Kelly Olynyk got a rare start and finished second on the team with his 16 points, six rebounds and four assists. Avery Bradley also tallied 13 points, eight rebounds and five assists, while Amir Johnson finished with 14 points and nine boards. Boston will return to action on Monday against the New Orleans Pelicans. Indiana (4-6) had a tough task here without Paul George and they simply couldn't overcome that loss. ...

Figure 4.1: Excerpt of a basketball game's summary from SportSett dataset corresponding to its statistics table.

The summaries in data-to-text generation datasets can be broken down into multiple components, each trying to convey a different type of information about different entities. As seen in the Figure 4.1, the first sentence of the summary describes general information about the game, while next few sentences discuss teams' performance followed by individual player's performances. We utilise this information to separate summaries into different components and present a system that can generate the summary for each component separately and then combine them to produce a final event summary.

We propose a Case-Based Reasoning (CBR) approach that learns to dynamically select relevant templates from the case-base based on the event's data to generate its summary. Our model learns to choose important entities from the data and then verbalises them via templates extracted from the training corpus. We run experiments to evaluate our proposed method on a sports domain dataset, SportSett (Thomson et al., 2020a). We demonstrate that our proposed method produces better quality texts than neural systems while also maintaining better diversity than rule based systems.

4.2 Case-Based Methodology (CBR-D2T)

We present a Case-Based approach to dynamically select templates for generating an event summary, called CBR-D2T. The input to a D2T system is a set of records organised by entities and features. Based on these records, an output summary needs to be generated that describes the event to which the given entities belong.

The D2T summaries can be broken into different components. These components try to describe certain types of information in certain ways. For example, in the sports domain, the summaries typically describe the winner of the game and then describe teams and players information. Similarly in the obituary domain, the summaries describe the deceased personal information followed by their family information, and then funeral arrangements.

Our CBR-D2T method uses separate case base for each component. The problem-side of a case in each component's case base is an entity's representation and solution-side is a template describing that entity. The CBR-D2T method first selects some important entities using an *entity importance* classifier and then retrieves relevant templates for each entity. These templates are filled with respective values and arranged in a pre-defined order to produce the final summary.

In this chapter, we use a sports domain dataset (SportSett) to evaluate our method. The task in this dataset is to generate textual summary of basketball games based on match data. We divide the summaries into four components:

1. **winning-team**: a sentence to describe the winning team and game's overall statistics;
2. **teams-performance**: a sentence to describe the comparative stats of both participating teams;
3. **players-performance**: a few sentences describing some important players contributions for each team;
4. **next-game**: describing the next game fixtures of both teams;

Since the sentences in winning-team and next-game components don't vary much, we use a set of 10 manually crafted templates to describe them. For the other two components, teams-performance and players-performance, we use our CBR-D2T method to sentences.

In the following sections, we describe: first, the methodology to create a case-base of templates from a textual corpus; second, an intermediate step of learning some parameters including feature weights and entity importance classifier; and third, the process of generating final event summary using the templates.

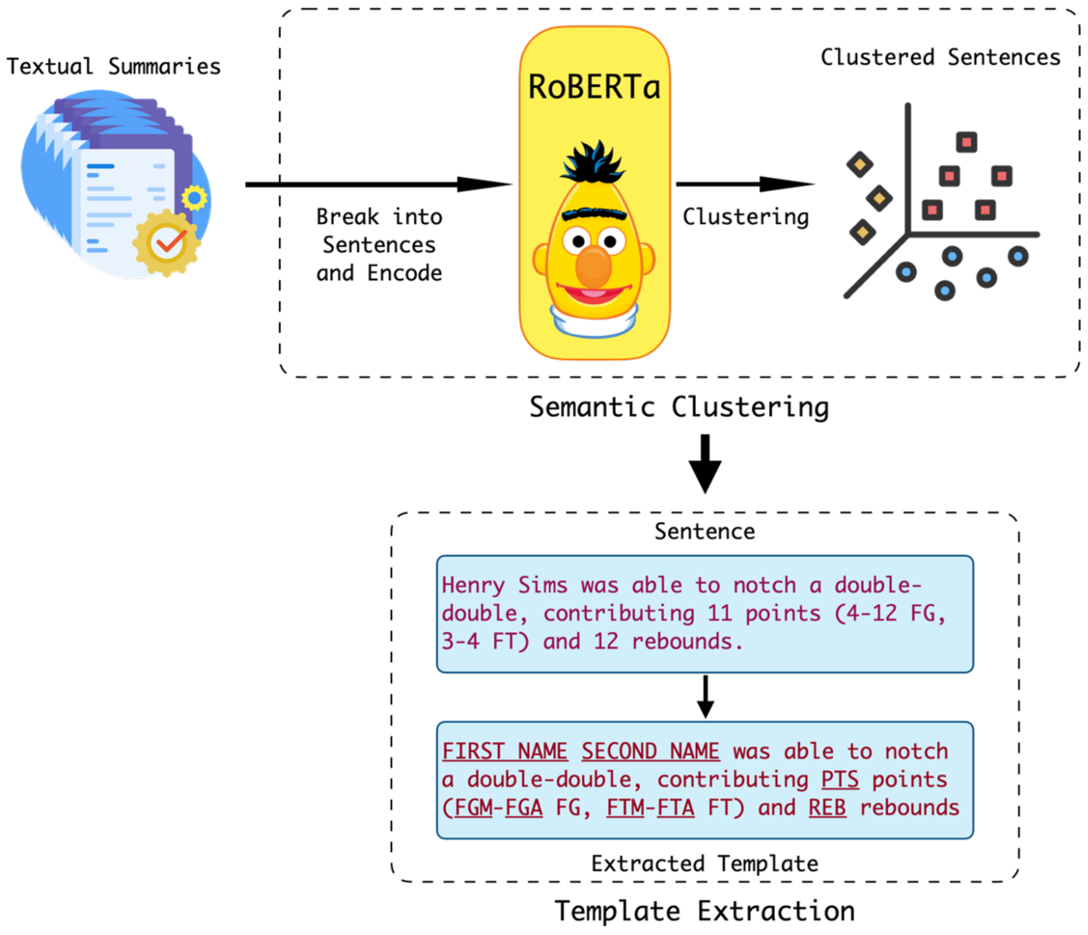

Textual Summaries
Break into Sentences and Encode
RoBERTa
Clustering
Clustered Sentences
Semantic Clustering
Sentence
Henry Sims was able to notch a double-double, contributing 11 points (4-12 FG, 3-4 FT) and 12 rebounds.
FIRST NAME SECOND NAME was able to notch a double-double, contributing PTS points (FGM-FGA FG, FTM-FTA FT) and REB rebounds
Extracted Template
Template Extraction

Figure 4.2: Creating Case-Base for a component

4.2.1 Case-Base Creation

The case-base for each component is created using a semi-automated process where first textual summaries are broken into sentences and similar sentences are clustered into the same group (described as *semantic clustering* below). Clusters related to the different components of the summary are identified manually and then used to extract the templates. Several templates will be selected for a new problem to assemble the textual solution. A pictorial representation of template extraction process is given in Figure 4.2. A representative case-base of a component is also shown in Figure 4.3.

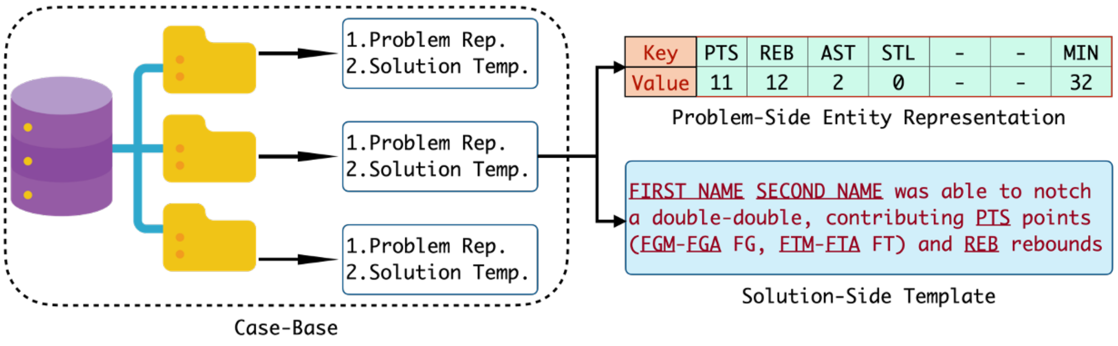

1.Problem Rep.
2.Solution Temp.
1.Problem Rep.
2.Solution Temp.
1.Problem Rep.
2.Solution Temp.
Case-Base
Key | PTS | REB | AST | STL | - | - | MIN
Value | 11 | 12 | 2 | 0 | - | - | 32
Problem-Side Entity Representation
FIRST NAME SECOND NAME was able to notch a double-double, contributing PTS points (FGM-FGA FG, FTM-FTA FT) and REB rebounds
Solution-Side Template

Figure 4.3: A component's Case-Base from the proposed CBR-D2T system

Semantic Clustering

We first extract all the sentences from the parallel corpus of input data and summaries. The named-entities and pos-tags are replaced using a place-holder to create an abstract version of summaries using the method described in Thomson et al. (2020b). The abstracting process uses open-source NLP libraries spaCy and neuralcoref combined with some domain-specific rules [1]. Through this process, a sentence from the dataset, e.g. '*The Atlanta Hawks (41-9) beat the Washington Wizards (31-19) 105-96 on Wednesday.*', is transformed to '*PROPN-ORG (X-Y) beat the PROPN-ORG (X-Y) X-Y on NOUN-DATE.*'. These abstract sentences are then embedded into a 786 dimensional vector using DistilRoBERTa Language Model [2]. Then a k-Means clustering algorithm is used to cluster these sentences into 50 clusters. These clusters are manually combined into two clusters, one representing sentences describing player performance, and another describing team performance.

[1]these rules can be found in the code here: `https://github.com/ashishu007/data2text-cbr/blob/main/clustering/create_abs.py#L42-L75`. Please note, this sentence abstraction is similar to one used in (Thomson et al., 2020b).

[2]`https://github.com/UKPLab/sentence-transformers`

Template Extraction

For each sentence in the cluster, the entity mentions are extracted[3]. If a sentence contains only one entity mention (e.g., specific team or player) then that entity's performance statistics are taken from the corresponding game. These entity statistics are in form of a stats-dictionary with multiple 'key-value' pairs, where the key is the feature-name and value is the numerical-value of that feature. Based on these statistics, a string matching is performed to find if any of the numerical-values from the stats-dictionary is mentioned in the sentence. If yes, then the numerical value is replaced with the key (feature-name) of matched value in the stats-dictionary. For example, from the sentence: "*Henry Sims was able to notch a double - double*[4] *, contributing 11 points (4 - 12 FG , 3 - 4 FT) and 12 rebounds .*" where Henry Sims' performance stats are: $\{STARTER : no, PTS : 11, FGM : 4, FGA : 12, FG\_PCT : 33, FG3M : 0, FG3A : 0, FG3\_PCT : 0, FTM : 3, FTA : 4, FT\_PCT : 75, OREB : 5, DREB : 7, REB : 12, AST : 2, TO : 0, STL : 0, BLK : 0, PF : 2, MIN : 32, IS\_HOME : no, FIRST\_NAME : Nerlens, SECOND\_NAME : Noel\}$, the template extracted is: "*FIRST_ NAME SECOND_ NAME was able to notch a double - double , contributing PTS points (FGM – FGA FG , FTM - FTA FT) and REB rebounds*"[5].

4.2.2 Learning Parameters

After creating the case-base, where each case's problem-side is an entities representation and solution-side is a template describing that entity (one template per entity). The next step is to learn some parameters in order to perform better retrieval, hence resulting in better summary generation. Two main learning modules used in CBR-D2T method are:

- **Entity Importance Classifier**: this module learns how to select important entities of an event.
- **Feature Weighting**: this module learns the weights for the features in the case-base.

[3]This entity extraction is performed automatically by searching for the exact string match of any player/team name in the sentence. Since the co-reference resolution is already performed on sentences on summary level in previous step, all the pronoun mention or short name mentions are replaced by full name of the entities, i.e., team or player. For each sentence, we also know which game's summary it is coming from, so its easier to search for string match of player/team names. The code for template extraction process is shared in the github repo at: https://github.com/ashishu007/data2text-cbr/blob/main/src/extract_template.py

[4]https://en.wikipedia.org/wiki/Double-double

[5]In case, a player doesn't score a double-double in the game, then the template retrieval process is expected to retrieve a similar template that doesn't mention double-double

These two are discussed in detail below.

Entity Importance Classifier

Many D2T domains contain multiple entities in each event, where only a handful of those entities contribute to the event summary. For example, in a sports domain, an event summary mainly discusses a few important players who have performed well in the game. Thus, selecting the important entities of an event beforehand can help improve summary generation. Specifically, for CBR-based systems, the selection of important entities can help filter out important information from the event's data, resulting into better retrieval.

To achieve this, we train an *entity-importance* classifier that classifies whether an entity is important enough to be discussed in the output summary of an event. Usually an entity's importance can not be decided in isolation, thus to decide an entities importance, the classifier also uses the information of other entities. This is achieved by concatenating the feature vector of the target entity in question with the feature vector of all other entities from the event. So, for an event with o entities with f features, an entity O_1 is represented as: $\{(O_{11}, O_{12}, \cdots, O_{1f}), (O_{21}, O_{22}, \cdots, O_{2f}), \cdots, (O_{o1}, O_{o2}, \cdots, O_{of})\}$, where O_{11} is O_1's first feature value and O_{of} is O_o entity's f^{th} feature value.

An entity is given class 1 (important) if it was mentioned in the summary of that game, or class 0 (not important) if it wasn't mentioned. A classifier is then trained to learn if the entity should be selected for discussing in the summary or not. We train a logistic regression classifier which achieves 87% accuracy and 85% f1 score on the validation set of the SporSett dataset.

It is to be noted that this module may not be needed in many domains, specially in domains with smaller number of entities involved in the event's data. For example, in obituary generation domain, an event only consists of one entity for which the summary is written and thus the entity selection step is not needed.

Feature Weighting

Feature weighting is important as not all features in the case-base have equal importance, and thus it is necessary to weigh these features based on their importance in case-retrieval. For this, we propose to learn the weights for the features of the case-base. We use a

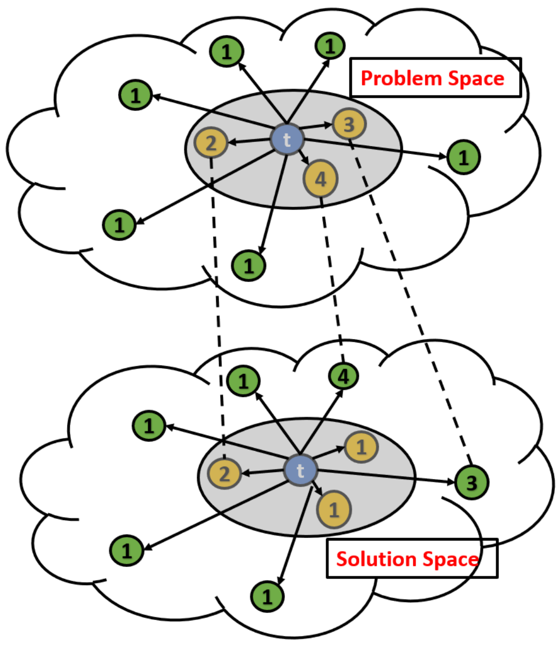


Figure 4.4: Case alignment of a CBR system

novel case-alignment metric as the loss function for learning the weights iteratively. In this section, we describe the case-alignment metric first and then move on to describing the novel method of using case-alignment for feature weighting.

Case-Alignment Metric A key principle of CBR is that "*similar problems have similar solutions*". The extent to which this principle holds true can be assessed by measuring the alignment between the problem-side and solution-side space. It is surmised that a good system design will have better alignment (Massie et al., 2007). Thus, we employ a novel approach to measuring case alignment by using normalised discounted cumulative gain to assess the correlation between problem-side and solution-side nearest neighbours. If the alignment is good then for a given problem-solution pair, the k nearest cases on problem-side should be similar to the k nearest-cases on the solution side.

For a given case-base C containing all the cases $\{c_1, c_2, \cdots, c_n\}$. Cases in C consist of problem$-$solution pairs (entity represented with its feature vector as problem, and a textual template describing that entity as solution), such that $c_i = \{p_i, s_i\}$, where $p_i \in P$ (problem set) and $s_i \in S$ (solution set). Given a target problem t represented using the case knowledge (an entity represented with its feature vector), we will retrieve two lists *pl* & *sl* which are sorted in order to the most similar cases both from the problem

(pl) and the solution (sl) set respectively. On the solution side, RoBERTa (Liu et al., 2019b) is used to encode the sentences and then cosine similarity between the test sample and other samples is used to generate the ordered list of similar cases. For the problem side the ordered list is created by ranking cases based on their problem-side similarity, calculated using euclidean distance. Both the lists will have $n-1$ items, where n is the size of the case-base.

From the list pl, we create a new list of weighted scores for the problem-side. We call it problem list weighted or plw. The weighting is done as follows:

$$plw(i) = \begin{cases} (k+1)-i, & \text{if } i \leq k \\ 1, & \text{otherwise} \end{cases} \tag{4.1}$$

where k is the number of neighbours (the cases with problem-solution pair, i.e., an entity's feature vector as problem and a text template describing that entity as solution) considered for retrieval and i is the index of each element from the pl. Similarly the cases in sl are weighted according to their pl counter-part and creating a solution list weighted or slw.

For example, if we have 10 cases in the case-base and for a given case c_i, with $k=3$ the sl and pl (both are ordered lists based on their similarity with respective problem and solution sides) are as follows:

$$pl = [8, 5, 6, 1, 4, 2, 3, 7, 0]$$
$$sl = [5, 6, 2, 4, 3, 7, 8, 1, 0]$$

These are the indices of the cases from both the sets. According to the eq. (4.1), weighted lists plw and slw are given as follows:

$$plw = [4, 3, 2, 1, 1, 1, 1, 1, 1]$$
$$slw = [3, 2, 1, 1, 1, 1, 4, 1, 1]$$

For an ideal case, both of the list should have same ranking order as they are retrieved for the same case. To measure the alignment of a target case t we can use the *"normalised Discounted Cumulative Gain" (nDCG)* using the following formula:

$$nDCG(t) = \frac{DCG_(slw)}{DCG_(plw)} \tag{4.2}$$

where, slw and plw are the weighted lists for the target case while DCG is the *"Discounted Cumulative Gain"*, defined for some list lw as:

$$DCG(lw) = \sum_{i=1}^{|lw|} \frac{lw(i)}{\log_2(i+1)} \tag{4.3}$$

where, lw is some weighted list (e.g., plw or slw) and $|lw|$ is the size of that list. The value of nDCG $\in [0, 1]$.

The alignment of whole case-base can be the average of nDCG score of all the cases in the case-base (CB).

$$AlignScore(CB) = \sum_{i=1}^{n} \frac{nDCG(i)}{n}, \forall i \in CB \tag{4.4}$$

where, n is the size of case-base. For component retrieval method, the total alignment score would be the average of $AlignScore$ of all the components.

Feature Weighting Algorithm Here, we now propose a novel case-alignment based feature weighting algorithm that can be used to learn the feature weights of non-classification problems; for example, where the solution is text summaries.

The alignment score is used as a loss function for a Particle-Swarm Optimiser whose parameters are the features' weights for a case-base. The loss function is formally defined in Algorithm 1. The ranked list on the problem side is generated using the euclidean distance between the problem-side of the target problem and the cases in the case-base, while the solution-side ranked list is generated using the cosine distance between the solution-side of target problem and cases in the case-base.

Algorithm 1 Calculate the loss value for each candidate generated in PSO algorithm

```
Input: PSO candidate $\mathbf{W}$, Problem-side $\mathbf{P}_{CB}$ and Solution-side $\mathbf{S}_{CB}$ of case-base
Output: The loss value for $\mathbf{W}$
 1: $\mathbf{WP}_{CB} = \mathbf{P}_{CB} * \mathbf{W}$
 2: $AlignScore_{CB} = 0$
 3: for each idx $\in range(\mid \mathbf{WP}_{CB} \mid)$ do
 4:     $\mathbf{P}_T, \mathbf{S}_T = \mathbf{WP}_{CB}[idx], \mathbf{S}_{CB}[idx]$
 5:     $\mathbf{P}\hat{_{CB}}, \mathbf{S}\hat{_{CB}} = \mathbf{WP}_{CB}[\sim idx], \mathbf{S}_{CB}[\sim idx]$
 6:     Generate problem-side ranked list PL using $\mathbf{P}_T$ and $\mathbf{P}\hat{_{CB}}$
 7:     Generate solution-side ranked list SL using $\mathbf{S}_T$ and $\mathbf{S}\hat{_{CB}}$
 8:     $AlignScore_{idx} = nDCG(\mathbf{PL}, \mathbf{SL})$
 9:     $AlignScore_{CB} += AlignScore_{idx}$
10: end for
11: $\mathcal{L}_W = 1 - (AlignScore_{CB} / \mid \mathbf{WP}_{CB} \mid)$
12: return $\mathcal{L}_W$
```

4.2.3 Summary Generation

Summary generation takes place in two parts: first, relevant templates are retrieved for each component and filled with respective information from the new problem; second, the filled templates for each components are combined into a single and final summary. The process is shown in Figure 4.5 and described in detail in following sections.

Template Retrieval

Template retrieval is done using a standard CBR process where the top-k most similar cases are retrieved for each target problem and their solutions are utilised to generate the solution for target problem.

Generation takes place separately for each component. For a target problem, k-nearest neighbours are retrieved from the case-base using problem-side representation. k solutions are generated by filling the template tags with their corresponding values in the nearest neighbour solutions. A GPT-2 (Radford et al., 2019) language model is used to rank the k sentences based on perplexity score. The template with the lowest perplexity score among k is chosen as the final solution for the given target problem. Since sentence ranking is a domain specific task, the GPT-2 language model is fine-tuned on the same dataset used in our experiments (a pictorial representation in Figure 4.6).

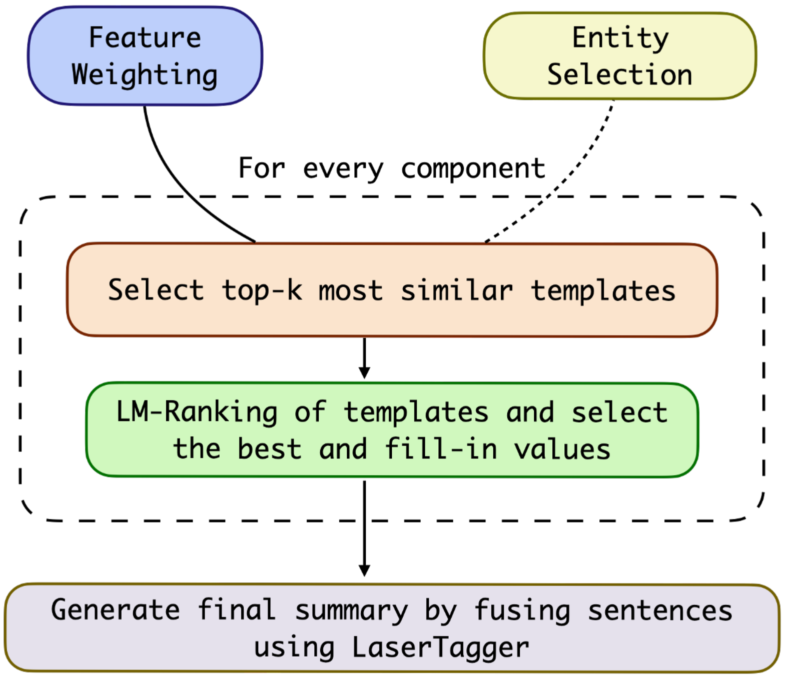

Feature Weighting
Entity Selection
For every component
Select top-k most similar templates
LM-Ranking of templates and select the best and fill-in values
Generate final summary by fusing sentences using LaserTagger

Figure 4.5: Generating final summary

Combining Templates

Now several sentences are generated for different components and are fused into a paragraph using the sentence fusing algorithm LaserTagger (Malmi et al., 2019) fine-tuned on the same dataset used in our experiments.

Where the datasets are complex: first, important entities are selected using the classifier; then, for each important entity selected, a sentence is generated using the process described above. Similarly, a sentence is generated for the entities from other component. Finally, all these sentences are fused into a paragraph using LaserTagger (Malmi et al., 2019).

4.3 Experiment Setup

Our proposed methodology is evaluated on a sports domain dataset, SportSett (Thomson et al., 2020a) that contains baseball game statistics paired with human written summaries from the NBA league in the USA. The task is to generate the game summary for a given game statistics. The experiments compare the CBR-D2T method against a neural benchmark and a rule-based template baseline. These systems are evaluated against

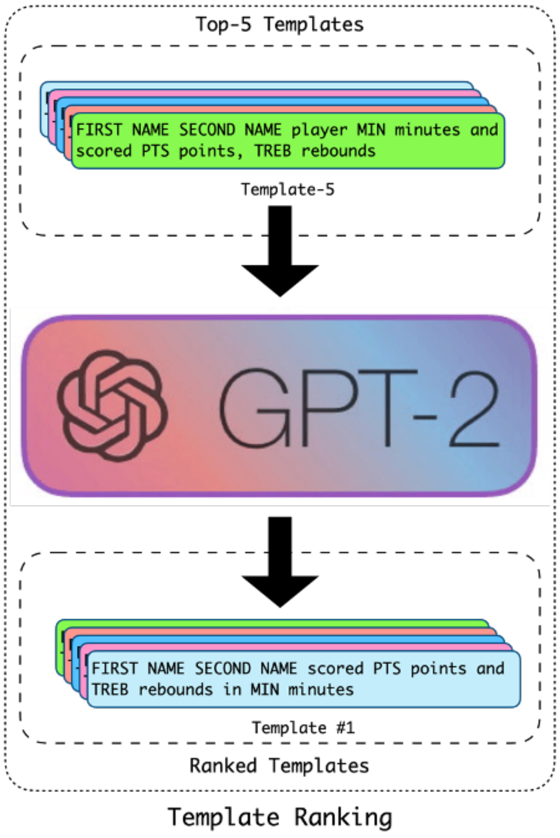

Top-5 Templates
FIRST NAME SECOND NAME player MIN minutes and scored PTS points, TREB rebounds
Template-5
GPT-2
FIRST NAME SECOND NAME scored PTS points and TREB rebounds in MIN minutes
Template #1
Ranked Templates
Template Ranking

Figure 4.6: Ranking retrieved templates using GPT-2 LM

accuracy, diversity and fluency of their generated summaries.

4.3.1 Size of Case-Bases

The summaries from SportSett dataset are divided into four components: first, winning team - describing the points made by each team and also the venue of the game; second, teams - both teams performance is described briefly; third, players performance - some important players from the game are described; and fourth, next game - which mentions the venue for next game of both teams. The sentences for first and last components are generated using a simple rule method while for the remaining two are generated using our CBR method. The team performance component has a case-base of 1200 cases, while players performance component has a case-base of 14985 cases.

4.3.2 Baseline and Benchmark

We compare our proposed CBR system, CBR-D2T, with two different types of D2T systems from literature: first, a rule-based template system - which provides high accuracy with low fluency and diversity; and second, a neural-based system - which provides high fluency and diversity but with lower accuracy.

Rule-Based Template System (Temp)

Wiseman et al. (2017) propose a rule-based template system for generating baseball summaries. This system has two standard templates, one for describing both team's data and another for the player's data [6]. We use the same system with some modification in our experiments. We extend the templates to include day, and arena of the game, as well as next-opponent team's name since this new information is now available in the SportSett data.

Learning-Based Neural System (Neural)

We use the Entity model of Puduppully et al. (2019b). It is a sequence-to-sequence model consisting of an MLP encoder and LSTM decoder with copy mechanism. There's an added module to update the input record's representation during the generation process. At each decoding step, a GRU is used to decide the record that needs to be updated and then its value is updated [7].

4.3.3 Evaluation

Extractive Evaluation

We use the family of Extractive Evaluation (EE) metrics for (Wiseman et al., 2017) for evaluating the models. These metrics are trained to extract entity names and numerical values from the text and predict the relation (feature name) between them. For example, from a sentence *'Ian Smith scored 15 points in the game'*, the IE models (ensemble of three LSTMs and three CNNs) can extract entity name *Ish Smith* and numerical value *15*. Then the model can predict its relation name as *PTS* (points) and will return a tuple as $t = (EntityName|FeatureValue|FeatureName)$ as $(IshSmith|15|PTS)$. The models extract several tuples from both human-written gold (y) and system generated ($\hat{y}$) summaries. These tuples are then compared to calculate the following metric scores:

[6] https://github.com/harvardnlp/data2text/blob/master/rulebased.py
[7] https://github.com/ratishsp/data2text-entity-py

- **Relation Generation (RG)** is the precision of unique tuples t extracted from generated summary $\hat{y}$ that also appeared in the input data. This metric can be used to measure the system's capability of generating factually correct texts supported by input data, i.e., accuracy of the system.
- **Content Selection (CS)** is the precision and recall between unique tuples t extracted from gold summary y and generated summary $\hat{y}$. Here, the systems ability of selecting content is measured in comparison with the human written summaries.
- **Content Ordering (CO)** is measured as the normalized Damerau Levenshtein Distance (Brill and Moore, 2000a) between the sequences of tuples extracted from generated summary $\hat{y}$ and gold summary y. This demonstrates the systems ability of ordering the content in generated summary.

CS primarily targets the challenge of *what to say?*, while CO targets the *how to say it?* aspect.

Initial versions of the Extractive Evaluation metrics (Puduppully et al., 2019a; Wiseman et al., 2017) only evaluated the numerical claims (such as points, rebounds, steals made by a player or team) mentioned in the text summaries. Authors in (Thomson et al., 2020b), proposed an extended version capable of evaluating day names, dates and game arenas as well. We further extend these metrics to evaluate the few more claims made in texts, for example, if a player was the leading scorer or if a player scored a double-double.

Fluency and Diversity

Apart from these metrics, we also use BLEU (Papineni et al., 2002) score to compare the generations. BLEU score compares the n-gram overlap between the gold summary and generated summary and primarily rewards fluent texts rather than generations capturing more information from the input (Wiseman et al., 2017). From all the metrics discussed here, RG can be used to measure the factual correctness of generations. While we acknowledge the fact that human judgement is the best evaluation practice for text generation systems, these autonomous metrics are a widely used proxy methods as crude surrogate for human judgement.

4.4 Results and Discussion

We discuss the results of our experiments in multiple ways.

Table 4.1: CBR-D2T against other systems on two datasets (Gold is human-authored summaries)

System	RG	CS-Precision	CS-Recall	CO	BLEU
SportSett					
Gold	89.46	–	–	–	–
Temp	95.35	55.20	23.65	10.61	6.50
Neural	71.07	45.66	40.81	19.56	17.68
CBR-D2T	77.22	46.46	33.53	11.92	11.93

- First, we compare our proposed CBR-D2T system for data-to-text generation with other established systems from the literature. Here we compare our system against rule-based baseline (Temp) and neural benchmark system (Neural). This experiment aims to establish the benefit of the CBR-D2T against other systems.
- Then we perform some ablation studies on SportSett dataset to analyse the effects of the following variables: size of training set (or case-base); case-alignment based feature weighting; LM scoring for selecting target-case's solution out of top-k most similar cases; LaserTagger for combining the sentences into final summary.
- Finally, we present a qualitative analysis of a summary generated from CBR system and analyse its incorrectness. We discuss the possible reasons for these mistakes and also present some possible mitigation methods for them.

The headline results from this chapter are:

- CBR-D2T is able to produce summaries more accurate than state-of-the-art neural system while also maintaining better fluency and diversity than a baseline template-based system
- CBR-D2T achieves better content accuracy than neural system even when one-third of training data is used

4.4.1 Comparison with Benchmark and Baseline

The results of our first experiments are shown in Table 4.1. We observe that our CBR-D2T system achieves better performance on RG (77% for CBR-D2T vs 71% for Neural) and CS-precision (46% for CBR-D2T vs 45% for Neural) metrics on SportSett dataset as compared to the Neural system, indicating CBR-D2T system's benefit over Neural on

Table 4.2: Effect of training set size on CBR and Neural systems

System	RG	CS-Precision	CS-Recall	CO	BLEU
Gold	89.46	–	–	–	–
$Neural_{oneS}$	61.34	34.84	25.68	9.84	9.93
$Neural_{allS}$	71.07	45.66	40.81	19.56	17.68
$CBR - D2T_{oneS}$	73.22	50.18	24.78	10.91	10.40
$CBR - D2T_{allS}$	77.22	46.46	33.53	11.92	11.93

Table 4.3: Ablation study results

	RG	CS-Precision	CS-Recall	CO	BLEU
wo CA FW	75.22	49.53	28.73	11.78	10.84
wo LM scoring	76.09	44.64	34.88	12.33	9.26
wo LaserTagger	76.36	44.50	33.04	11.30	11.04
$CBR - D2T_{allS}$	77.22	46.46	33.53	11.92	11.93

accuracy. We also note that Temp system achieves best score on RG (95%) and on CS-precision (55%). This is not surprising as Temp is hard-coded with domain knowledge thus is very accurate [8]. But Temp fares poorly on mimicking the gold summaries, as it only recalls 23% of the contents from gold summaries.

On the terms of fluency, we observe that BLEU score achieved by CBR-D2T is better than the Temp. This reflects that our system is more fluent compared to Temp, but Neural does very well against others. To measure the diversity, we first calculate the vocabulary of texts generated from different systems. We identify that the Gold summaries from the test set have a vocabulary of more than 5000 words, while both the CBR-D2T and neural systems have the vocabulary of 2000 words while Temp has vocabulary of only 900 unique words. In terms of content selected for output summary, Temp is only able to discuss one-third of the unique record types discussed in Gold summary, while our CBR-D2T is able to discuss all of them.

This evidence suggests that CBR-D2T is able to balance the accuracy vs diversity trade-off. CBR-D2T does so by dynamically retrieving similar templates from the case-base to achieve better diversity while not comprising on accuracy.

4.4.2 Ablation Studies

We further perform four ablation studies on our **CBR-D2T** system. The first study aims to analyse the effect of training data on CBR-D2T and Neural systems on SportSett dataset. Here, for both Neural and CBR-D2T, two separate systems are trained on two sizes of train set. First only one season data (NBA 2014 season) with 1230 instances is used to train both systems, regarded as **CBR-D2T**$_{oneS}$ and **Neural**$_{oneS}$. Then all the data from SportSett's train set (NBA 2014, 2015, 2016 seasons) with a total of 4575 instances is used to train these systems and are regarded as **CBR-D2T**$_{allS}$ and **Neural**$_{allS}$. From the results shown in Table 4.2, we note that our **CBR-D2T**$_{oneS}$ outperforms **Neural**$_{oneS}$ on all metrics, except CS-recall. This demonstrates that CBR-D2T is much better at producing quality texts compared to Neural with fewer training samples. However, with the increase in training data, there is a substantial gain in performance of **Neural**$_{allS}$ across all metrics while **CBR-D2T**$_{allS}$ system improves very little. Although even then, **CBR-D2T**$_{allS}$ is better than **Neural**$_{allS}$ in terms of RG and CS-Precision.

In the second study, we analyse the effect of our proposed case-alignment based feature weighting against the information gain based feature weighting (see Section 4.2.2). The results of this ablation are shown in the first row of Table 4.3. Here information gain from the important player classifier data is used to weight the features in the players' component, however, no weighting is applied for teams' component. From the results, we can see that apart from CS-precision, there's at-least some drop in all metrics while a sizeable drop in CS-recall. The drop in CS-recall shows that the system with case-alignment feature weighing is able to select templates that have contents closer to the human written summaries.

In the third study, we compare the effect of selecting the 'nearest neighbour' against 'best out of top-k nearest neighbour' for generating the new solution, of which results are shown in second row of the Table 4.3. Here, the **CBR-D2T**$_{allS}$ system is used without GPT2 solution ranking module. Again, we can see there's not much difference in the EE metrics but there's a sizeable difference in BLEU score. This is expected as GPT2 scores the sentences based on perplexity that rewards fluency. So with the addition of an extra scoring component for solution reuse, we can improve the fluency of our generated text summaries.

[8]Note that the scores are not 100% because the metrics are based on trained models, which themselves achieve around 90% accuracy and f1 score while training

The visiting Atlanta Hawks (13-29) defeated the host Philadelphia 76ers (27-16) 123-121 at Wells Fargo Center on Friday.
The 76ers shot 52 percent from the field , including 33 percent from long range but were not able to hang on for the full 48 , as The Hawks surged back to get their revenge . *Ben Simmons had a triple - double with 23 points* **(7 - 13 FG , 1 - 2 3Pt)** , *10 rebounds , 15 assists , three steals and one block in 43 minutes . Meanwhile , Jimmy Butler was the high-point man for Philadelphia, with 30 points on 9-of-19 shooting, in 40 minutes* . JJ Redick was next in line with *20 points* , **three rebounds , an assist** and *a steal* , ***as the only other 76ers player who managed double - digit points*** . **Rookie DeAndre' Bembry** shot *6 - for - 11 from the field to score 14 points* , while also chipping in **five rebounds** . *Kevin Huerter was the high - point man for the Hawks* as he tied a season - high with *29 points on 11 - of - 17 shooting , including 5 - of - 8 from long range* . John Collins was the 3 prong of the Hawks attack , as he finished with *25 points (10 - 17 FG , 1 - 1 3PT , 1 - 3 FT)* , along with **five rebounds** , *two assists* , **two steals and one block , in 27 minutes** .
The 76ers now head to New_York for a Sunday night showdown versus the Knicks while the Hawks will return home to face the Bucks on Sunday.

Table 4.4: A summary generated from our $\textbf{CBR}_{allS}$ system

The final study analyses the effects of applying LaserTagger for sentence fusion. Results are shown in the third row of the Table 4.3. We can see that there's slight drop in most metrics when LaserTagger is not used for sentence fusion. This is because that the texts generated from CBR-D2T have some incoherence such as: '*Bradley Beal led the way for* ***the Wizards (32-48)*** *with 25 points, complementing* ***the Wizards (32-48)*** *with five assists and two rebounds.*'. This incoherence is the result of using co-reference resolution while template extraction process. With LaserTagger applied to the above sentence, it is modified into '*Bradley Beal led the way for* ***the Wizards*** *with 25 points, complementing* ***them*** *with five assists and two rebounds.*'.

4.4.3 Qualitative Analysis

A summary of the NBA match between 76ers and Hawks on 11th Jan, 2019 generated from the proposed system is shown in Table 4.4. The first and last sentences are generated from a set carefully crafted rules, while other sentences are generated by CBR-D2T. The accurate facts are shown in *italics* while the inaccurate ones in **bold**. The **bold**

inaccuracies are due to the imperfection in the template extraction process, where sometimes an entity is replaced with incorrect feature name because of more than one features having the same value. By addressing such issues during template extraction accuracy can be improved in terms of numerical facts being conveyed. Another type of inaccuracy is shown in ***bold and italics*** sentence, which wrongly mentions JJ Redrick being the only player with double-digit points after Jimmy Butler. These facts are much more complex to identify and are grounded in the text, as this information is calculated across multiple entities (you need to know other players' scores as well to decide only two scored in double-digits). To address such errors, new features are needed to explicitly identify the information from across entities and/or events. Those new features will help the system in deciding the similarity of a template with across-entity information in such cases.

4.5 Conclusion

In this chapter, we proposed a CBR method for D2T, called CBR-D2T, to mitigate the accuracy and diversity trade-off when compared to neural- and rule- based data-to-text systems. CBR-D2T dynamically selects relevant templates for an event based on the system's previous experience of generating summaries of previous events. It follows a component based approach to summary generation where first the summary of an event is broken down into multiple components and for each component a separate case-base is build.

Experimentation results on a sports domain dataset, SportSett, demonstrates the benefit of our proposed CBR-D2T system over neural- and rule-based systems. CBR-D2T is able to achieve better accuracy than a neural benchmark while better fluency and diversity than a rule-based baseline.

Chapter 5

Content-Type Profiling of Datasets

In the background chapter, we discussed different types of D2T datasets related to several real-world use-cases. We identified that some datasets have longer summaries than others and also need to process much more data in order to generate summaries. The summaries of these datasets can contain different type of information with several level of complexity involved in deriving them. It is important to understand the different types of content present in the summary to help us better define the system requirements so that we can build better systems.

In this chapter, we propose a novel typology of content types, that we use to classify the contents of event summaries. Using the typology, a profile of a dataset is generated as the distribution of the aggregated content types which captures the specific characteristics of the dataset and gives a measure of the complexity present in the problem.

5.1 Motivation

An ecologically valid task requires the automated systems to resemble the real-world scenario as closely as possible in its output (de Vries et al., 2020). Accordingly, a D2T system needs to convey important insights extracted from the data in the textual summaries (Gatt and Krahmer, 2018; Reiter, 2007b). Many D2T problems can be viewed as a stream of time-stamped events with a textual summary of each event presenting the insights. An event is the time-period of interest for which the textual summary is written. For example, in sports reporting - a game played between two teams can be an event; whereas in weather forecasting - the time period and location for which the

The Bucks (10-10) *handled* **the Heat (9-9) 109-85** on **Friday night in Milwaukee**. ***It was the second victory over Miami for the Bucks this season after emerging victorious in Miami 91-84 on Nov. 16***. *Milwaukee fell behind early but clawed back into the game in the second quarter and held a four-point advantage at half. The Bucks were led by an unlikely face in Kendall Marshall*, who scored a ***season-high*** **20 points (7-8 FG, 4-5 3Pt) in 24 minutes**.

Figure 5.1: Part of a basketball summary showing different types of content. Information in "**bold**" such as Bucks' points in the game (109) can be directly copied from input data, while the ones in "*italics*" needs to be derived from multiple records such as the fact - Kendall Marshall leading the Bucks team. Finally, the information in "***bold & italics***" such as this was Miami's second victory against Bucks are derived using records from multiple events in the stream.

forecast is written, can be considered one full event. The summaries can contain different types of facts with information sometimes coming from multiple events in the stream. In many cases, the facts in an event summary are verbatim of input records. Other times, these facts are derived from multiple records of either the same or multiple events in the stream. As an example, we show a part of baseball summary with multiple types of content in Figure 5.1.

A distribution of different content types in the summaries of a dataset can be used to generate a profile that can capture the specific characteristics of the dataset and provide a measure of complexity present in the problem. The dataset profile can help us better define the D2T system requirements, such as: the type of system to build - *is a complex domain-specific system required?* or *can a general system be effective*; or any information gap (Thomson et al., 2020b) that needs to be bridged at the data level. Generally, a problem's complexity is identified with the evaluation of systems built for the task. There are several methods to evaluate the D2T systems, mostly by measuring the factual accuracy of generated texts (Garneau and Lamontagne, 2021; Kasner et al., 2021; Thomson and Reiter, 2020; Wiseman et al., 2017) or lexical similarity of generated texts with reference texts (Lin, 2004; Papineni et al., 2002; Zhang et al., 2020b). These, whilst being promising are reactive measures where a full cycle of system development is needed to evaluate both the dataset and D2T system together. Whereas dataset profiling can be a proactive measure to gain important insights about the task before even starting system development. There are other utilities of dataset profiling method as well, most notably, it can be used as a measure of dataset's complexity in *datasheets for dataset* (Gebru et al., 2021).

5.2 Methodology

The idea proposed in this chapter is that we can create a profile capturing the specific characteristics of a dataset by looking at the distribution of content types present in its summaries. First, the content typology is defined by identifying the different types of information that can be included in an event summary and then, a general approach is defined to build a content type classifier to quickly profile the datasets.

5.2.1 Content Type Typology

The textual summary in a D2T dataset may contain information derived from different sources. To begin with, a fact can be derived from either the same event or from the records of different events. For example, a basketball game summary generally mentions different stats scored by players in the game. Such information is explicitly present in the input data of the game and can be directly copied to the output summary. Most times, summaries also mention the average stats recorded by a player in the past few games. To derive such facts, the generation system needs to consider the records from previous games as well. So based on the source of the information, content can be categorised as either **intra-event** (contains facts derived from the current event's records) or **inter-event** (contains facts derived from across-event records).

We can further separate the intra-event contents into more granular categories to identify the difficulty of generating a fact within the same event. Again, taking an example from a basketball summary, the summary could either mention some specific statistic (points or rebounds) scored by a player in the game, or mention if the player has scored a double-double. The information of a specific statistic of players is explicitly present in the input data, which can be directly copied to the output summary. While the fact that the player scored a double-double is not explicitly present in the input data. To derive such facts, the system needs to consider several records of that player. So, within intra-event, there can be two different types of content: **basic**, that can be just copied directly from the input data; and **complex**, that needs to be derived from multiple input data records of the current event.

Considering the following notations: each summary $\mathcal{S}_i$ is a combination of multiple **sentences** $\mathcal{T}_{i,j}$, which will contain at-least one or more **information elements** (facts) $\mathcal{L}_{i,j,k}$.

$$\mathcal{S}_i = \{\mathcal{T}_{i,1}, \mathcal{T}_{i,2}, \cdots, \mathcal{T}_{i,j}\} \tag{5.1}$$

$$\mathcal{T}_{i,j} = \{\mathcal{L}_{i,j,1}, \mathcal{L}_{i,j,2}, \cdots, \mathcal{L}_{i,j,k}\} \tag{5.2}$$

The D2T problems are formally described in Chapter 3 as follows:

A data instance in an event-based **D2T dataset** $\mathcal{DB}$ is an **event** $\mathcal{E}_i$ with a **data structure** $\mathcal{D}_i$ for which a **textual summary** $\mathcal{S}_i$ is written summarising the insights and information of the event. A data structure consists of multiple **entities** $\mathcal{O}_{i,o}$ that are the objects involved in the event, and each entity is described by multiple **features** $\mathcal{F}_{i,o,f}$ which are the attributes of those entities.

$$\mathcal{DB} = [\mathcal{E}_{i-e}, \cdots, \mathcal{E}_i, \cdots, \mathcal{E}_{i+e}] \tag{5.3}$$

$$\mathcal{E}_i = \{\mathcal{D}_i, \mathcal{S}_i\} \tag{5.4}$$

$$\mathcal{D}_i = \{\mathcal{O}_{i,1}, \mathcal{O}_{i,2}, \cdots, \mathcal{O}_{i,o}\} \tag{5.5}$$

$$\mathcal{O}_{i,o} = \{\mathcal{F}_{i,o,1}, \mathcal{F}_{i,o,2}, \cdots, \mathcal{F}_{i,o,f}\} \tag{5.6}$$

When building a text generation system g for a D2T task, the summary of an event is the function of current event as well as other events in stream:

For problems that do not require contextual awareness:

$$\mathcal{S}_i = g(\mathcal{D}_i) \tag{5.7}$$

For problems that require contextual awareness:

$$\mathcal{S}_i = g(\mathcal{D}_i, \mathcal{DB}) \tag{5.8}$$

A value $\mathcal{R}_{i,o,f}$ will be recorded for each feature $\mathcal{F}_{i,o,f}$ of an entity $\mathcal{O}_{i,o}$ that can be considered a record in the data structure. So if an input data structure is flattened into a sequence of records (mostly for training a neural generation system), it looks as:

$$\begin{aligned}\mathcal{D}_i = \{\mathcal{R}_{i,1,1}, \mathcal{R}_{i,1,2}, \cdots, \mathcal{R}_{i,1,f}, \cdots \\ \mathcal{R}_{i,2,1}, \mathcal{R}_{i,2,2}, \cdots, \mathcal{R}_{i,2,f}, \cdots \\ \mathcal{R}_{i,o,1}, \mathcal{R}_{i,o,2}, \cdots, \mathcal{R}_{i,o,f}\}\end{aligned} \tag{5.9}$$

We propose a typology of three classes to distinguish the types of contents based on the information elements a sentence contains in the summary:

Intra-Event Basic ($\mathcal{B}$): a sentence which contains information element(s) directly copied from the set of input records of the same event.

$$\mathcal{B} \iff \mathcal{L}_{i,j,k} = \mathcal{R}_{i,o,f} \tag{5.10}$$

Intra-Event Complex ($\mathcal{C}$): a sentence which contains information element(s) derived from multiple records of the same event.

$$\mathcal{C} \iff \mathcal{L}_{i,j,k} = \mathcal{R}_{i,o,f} \otimes \mathcal{R}_{i,o\pm l,f\pm m} \otimes \cdots \tag{5.11}$$

Inter-Event ($\mathcal{I}$): a sentence containing information element(s) derived using at-lease one record of a past-event. This includes: a single record copied from a past event; an information element derived using multiple records of one or many past events; or an information element derived from records of current event and at-least one past event.

$$\mathcal{I} \iff \mathcal{L}_{i,j,k} = \mathcal{R}_{i,r} \otimes \mathcal{R}_{i-n,o\pm l,f\pm m} \otimes \cdots \tag{5.12}$$

where, l, m, n are positive integers and $\otimes$ is an operation that requires inference between more than one records.

Taking the example from Figure 5.1, the last sentence in the summary: "The *Bucks* were *led* by an unlikely face in *Kendall Marshall*, who scored a *season-high 20 points* (*7-8 FG, 4-5 3Pt*) in *24 minutes*" has multiple information elements. To calculate the

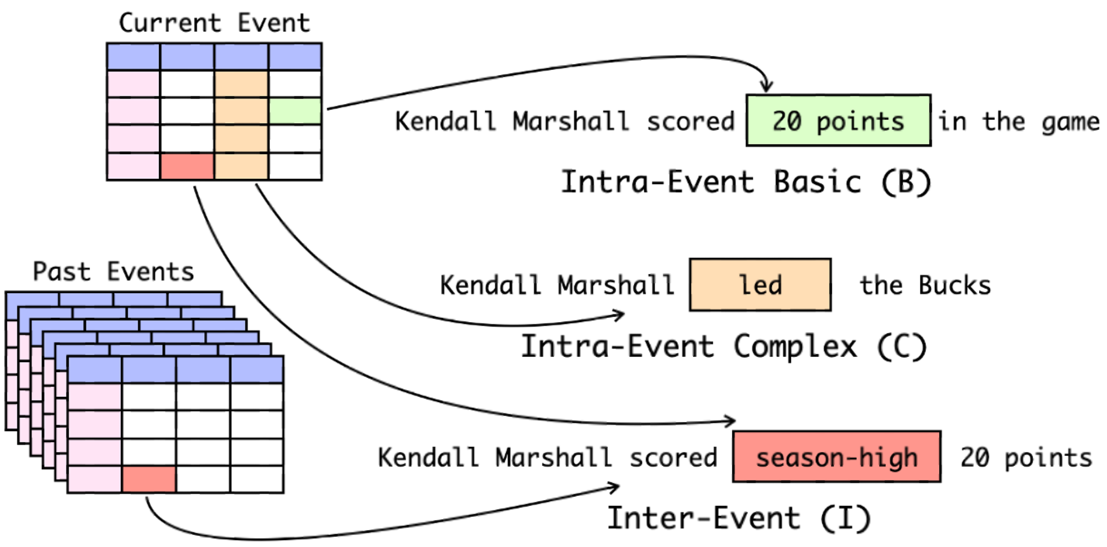

Current Event
Past Events
Kendall Marshall scored 20 points in the game
Intra-Event Basic (B)
Kendall Marshall led the Bucks
Intra-Event Complex (C)
Kendall Marshall scored season-high 20 points
Inter-Event (I)

Figure 5.2: Different content-types in a D2T summary. The Intra-Event Basic fact is taken from only one record, while the Intra-Event Complex fact is derived from multiple records of the same event. Finally, the Inter-Event fact is derived from the records of multiple events.

information that *Kendall Marshall led the Bucks*, the generation system should be able to analyse the records of all players in Bucks team, which is **Intra-Event Complex** ($\mathcal{C}$) type of information element. Then the information that he scored a *season-high 20 points*, will only be calculated by analysing the records of all games in which Kendall Marshall played, which is **Inter-Event** ($\mathcal{I}$) type of information element. And finally, the shot-breakdown (*7-8 FG, 4-5 3Pt*) and *number of minutes* he played are the information elements that can be directly copied from the input, which are **Intra-Event Basic** ($\mathcal{B}$) type of information elements.

It is also possible to extend the Inter-Event class into Basic and Complex categories. But we don't do that here because: first, the occurrence of Inter-Event Basic is rare. For example, in a baseball summary, mentioning a player's stats from a previous game is an inter-event basic information, but the chances of discussing just such facts is rare, usually those stats are used to derive more complex facts such as averages across games. And second, both pose similar challenges to current state-of-the-art D2T systems which has access to across-event data in the input during run-time.

5.2.2 Building Content Type Classifier

Now that we have defined the typology of different content-types in a dataset, we want to build a classifier that can automatically classify all the contents in a dataset to build the profile. Since, sentence $\mathcal{T}_{i,j}$ in a summary $\mathcal{S}_i$ can have multiple information elements $\mathcal{L}_{i,j,k}$ of multiple types, the sentence can also be of more than one type, which makes the task of content-type classification a multi-label classification problem. We can build a multi-label classifier f to map the sentence $\mathcal{T}$ to its type y as:

$$y_{i,j} = f(\mathcal{T}_{i,j}) \tag{5.13}$$

where $y_{i,j} \subseteq \mathcal{Y}$, and $\mathcal{Y} = \{\mathcal{B}, \mathcal{C}, \mathcal{I}\}$

One can take a manual approach for creating the dataset profile by asking human annotators to manually annotate the contents. But that is expensive as datasets can be large with some having more than 20,000 summaries, each at-least 15-20 sentence long. Accordingly, we look at alternative methods by building a classifier that alleviates the expense of labelling all summaries manually.

Even to build the classifier, we need some labelled samples beforehand for training and testing. To do that we used an Active Learning (AL) approach to create the train/test sets for content-type classification (Settles, 2009). Employing an AL approach, we start with a seed of 200 randomly labelled samples, and then use a 'Query-by-Committee (QBC)' strategy to select new samples for labelling until the performance plateaus. QBC method works as follows:

1. train a committee of classifiers with 'k' members on the 200 randomly labelled samples and also measure their individual performance on a held-out test set;
2. use the classifiers to predict the class probabilities for unlabelled samples;
3. rank the unlabelled samples based on the disagreement between classifiers, the higher the disagreement - the more informative a sample is. Disagreement between classifiers for a sample is measured using a vote entropy method (Settles, 2009, p. 17);

Table 5.1: Content-Type classifiers performance on different datasets

Dataset Classifier	MLB RobFT	SportSett RobFT	SumTime SVM w/ RobEmb	Obituary SVM w/ RobEmb
# Train Samples	600	600	200	200
Macro F1	85.64	91.0	98.66	98.46

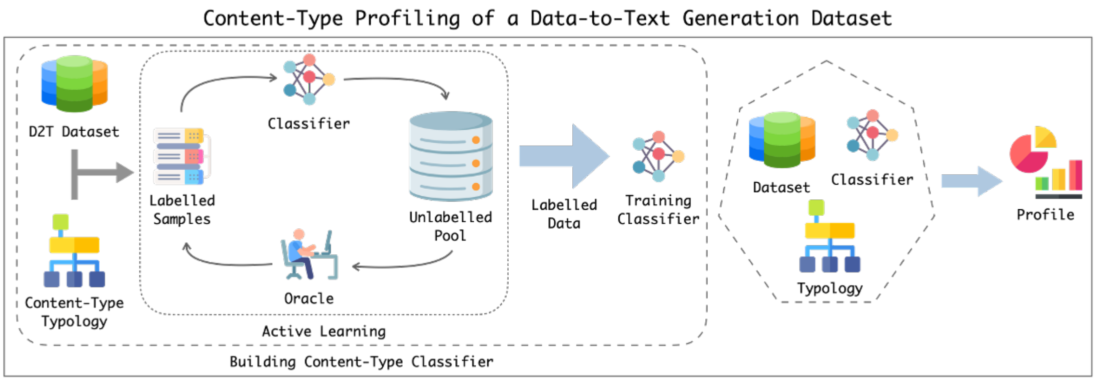


Figure 5.3: Generating the Content-Type Profile of a dataset

4. label the top 50 most informative samples from the ranked list of unlabelled samples;

5. repeat the above steps until desired performance from the classifiers is achieved or all the unlabelled samples have been labelled;

The committee of classifiers for QBC are build using three learning methods: Linear SVM; Random Forest; and Logistic Regression, with three different feature representations: TF; TF-IDF; and embeddings from DistilRoBERTa language model (RobEmb) (Reimers and Gurevych, 2019). With a combination of three learners and three feature representations, a total of 9 classifier are build. We also add a fine-tuned DistilRoBERTa model with multi-label classifier head (RobFT) (Liu et al., 2019b) to the committee as 10^{th} classifier. The best performing classifier from the committee on the human-annotated held-out test set is then used as the content-type classifier. Performance of best performing classifier for each dataset is shown in Table 5.1. A pictorial representation of creating a dataset's profile using our proposed typology is show in Figure 5.3.

Table 5.2: Statistics of different datasets

	Obituary	SumTime	SportSett	MLB
# Samples in Train set	80	793	3690	22821
# Samples in Valid set	20	99	1230	1739
# Samples in Test set	20	100	1230	1744
Avg. Attributes in Input Data	20	500	500	600
Avg. Sentences in Output Summaries	3	12	15	20

5.3 Experiment Setup

The experiments are performed in three phases. The first phase, the aim is to understand the characteristics of human authored summaries, for which, dataset profiles are generated based on human authored summaries using the proposed content-type typology (5.4.1). The second phase aims to demonstrate the challenge state-of-the-art generation systems face in attempting to generate complex content (5.4.2). This is evaluated by comparing the errors made by the generation systems for each content-types. Finally, the proposed methodology is used to understand the concept-drift issue in a problem domain which can help in building better systems capable of handling such domain-specific issues (5.4.3).

5.3.1 Datasets

Four datasets from different domains are used for profiling: **MLB** (Puduppully et al., 2019b) and **SportSett** (Thomson et al., 2020a) datasets from sports domain; **SumTime** (Sripada et al., 2003) dataset from weather forecasting; and **Obituary** (Upadhyay et al., 2020) from Obituary generation domain. Table 5.2 presents some basic statistics of datasets used in the experiments.

5.3.2 Generation Systems

Phase two of the experiments analyses the ability of state-of-the-art generation systems in producing different types of content. For this, several state-of-the-art generation systems are used to produce summaries on the held-out test-set of datasets mentioned above. For MLB and SportSett, two benchmark neural systems from literature are used: first, the macro-planning model (**Plan**) from Puduppully and Lapata (2021); and second, the entity-based model (**Ent**) from Puduppully et al. (2019b) are used on both datasets. In addition, the hierarchical transformer model (**Hir**) from Rebuffel et al. (2020) is used as a third model for SportSett. For Obituary and SumTime, we are not aware of any existing neural benchmarks, therefore we develop our own generation systems by fine-tuning T5-base (**T5**) from Raffel et al. (2020), BART-base (**BART**) from Lewis et al. (2020), and Pegasus (**Peg**) from Zhang et al. (2020a) on each dataset respectively [1].

5.3.3 Generation Systems' Accuracy Evaluation

We use the evaluation methodology from Thomson and Reiter (2020) to evaluate the factual accuracy of system generated summaries. The distribution of sentences across different content-types from all the evaluated system generated summaries is shown in Table 5.3.

5.4 Results and Discussions

The headline results from this chapter are as follows:

- human authored D2T summaries are mix of different types of contents, often content derived from multiple records of same or other related events in the stream
- state-of-the-art neural systems struggle to generate content of complex types that require processing of multiple records

5.4.1 Analysing Human Written Summaries

The content type distribution found in human authored summaries from different datasets (generated using the content-type classifier discussed in Section 5.2.2) is shown in Figure 5.4. On the x-axis, the different content-type categories are shown, while the y-axis displays the percentage of sentences containing that category. We can see that MLB has the highest amount of sentences (48%) containing inter-event information with 55% containing intra-event complex, and 45% containing intra-event basic information. SportSett has 29% sentences containing inter-event information with 65% sentences containing intra-event complex and 63% with intra-event basic. In SumTime, although there are no sentences with inter-event information, 87% sentences contain intra-event complex with 99% of them also containing intra-event basic information. Obituary has 91%

[1] The train set of obituary dataset is augmented using a simple entity replacement method to increase the size from 80 to 800. This is done because only 80 samples were not enough for training the neural models

Table 5.3: Number of sentences from different categories manually annotated for error-rate evaluation

Systems	Intra Basic ($\mathcal{B}$)	Intra Complex ($\mathcal{C}$)	Inter ($\mathcal{I}$)
MLB			
Ent (Puduppully et al., 2019b)	83	119	55
Plan (Puduppully and Lapata, 2021)	94	193	47
SportSett			
Ent (Puduppully et al., 2019b)	71	71	44
Plan (Puduppully and Lapata, 2021)	84	53	33
Hir (Rebuffel et al., 2020)	82	84	46
SumTime			
T5 (Raffel et al., 2020)	20	10	0
BART (Lewis et al., 2020)	20	10	0
Peg (Zhang et al., 2020a)	20	10	0
Obituary			
T5 (Raffel et al., 2020)	35	11	0
BART (Lewis et al., 2020)	30	10	0
Peg (Zhang et al., 2020a)	33	10	0

sentences with intra-event basic information as well as the least percentage of sentences with intra-event complex information (36%). Obituary also doesn't have any inter-event type sentences in the summaries.

These numbers suggest that human written summaries do not contain just information copied from input data. Rather, they are full of complex insights derived from multiple records, and possibly multiple events in the stream. This demonstrates that while designing a D2T system, two requirements are necessary: first, in most D2T tasks, an event cannot be considered as independent, as it's derived summary might depend on the data from multiple events; and second, a system developed for a D2T task should be capable of performing complex analytical operations in order to derive implicit information from

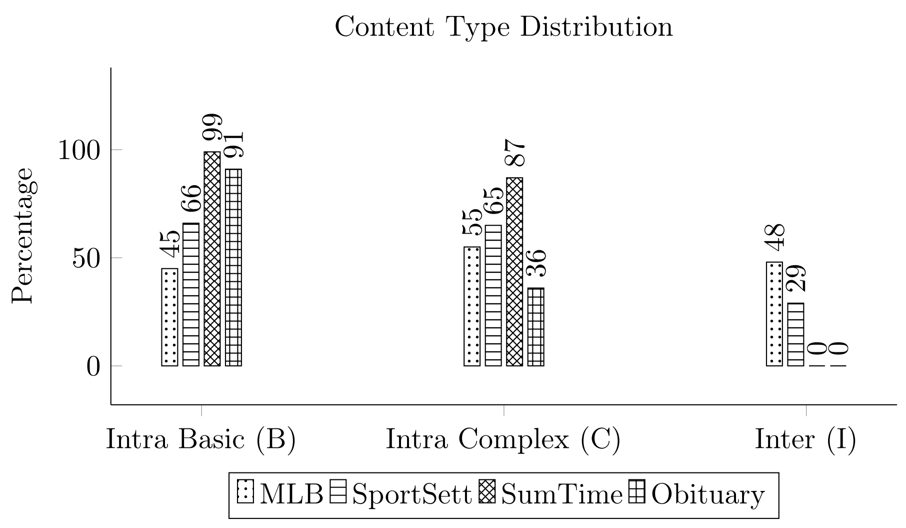


Figure 5.4: Distribution of content types in human authored summaries

the given input data. Ignoring these requirements will lead to building systems capable of generating only the easier and potentially dull content while completely missing out on complex insights, similar to problems in other language generation tasks such as concept-to-text generation or dialogue generation (Du and Black, 2019; Feng et al., 2021).

5.4.2 Analysing System Generated Summaries

After analysing the human written summaries, we move on to investigate the abilities of current state-of-the-art generation systems in producing different types of contents [2]. In Table 5.4, we show the percentage of accuracy errors made by each system within each content-type category. The accuracy scores demonstrate that generation systems struggle to produce the complex type content. Every generation system across all datasets have low error-rates in intra-event basic content while higher error-rates in intra-event complex and highest error-rates in the inter-event contents.

Error rate in **intra-event basic contents** ranges from below 3% in Obituary, around 10% in MLB and SportSett, to over 35% in SumTime. SumTime has the higest error, which may be due to lack of training data and a highly domain specific problem that requires identification of multiple relationships to generate summaries. The systems used for SumTime are not custom designed for the task as with MLB and SportSett,

[2]the automated evaluation results of different text generation systems is shown in Table 5.5

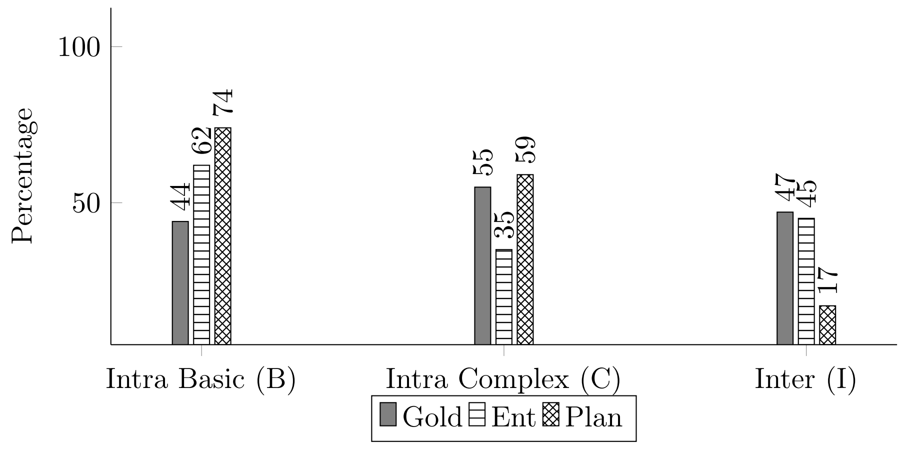


Figure 5.5: Profile of Automated Systems on MLB Dataset

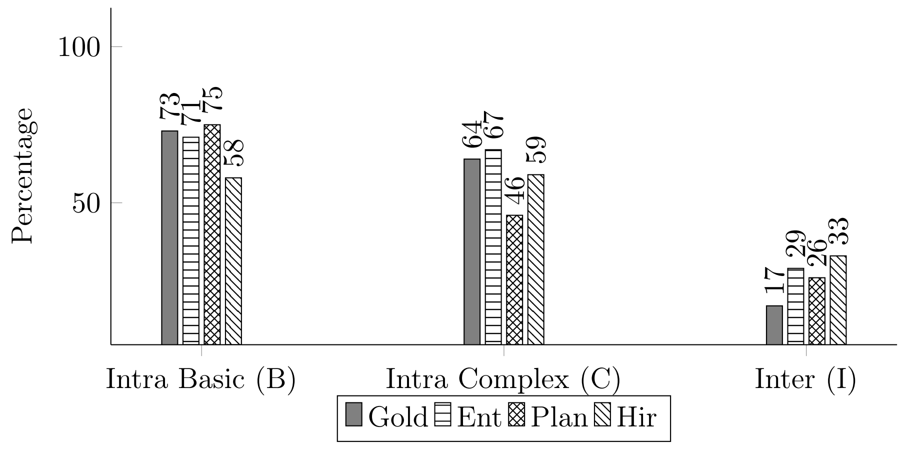


Figure 5.6: Profile of Automated Systems on SportSett Dataset

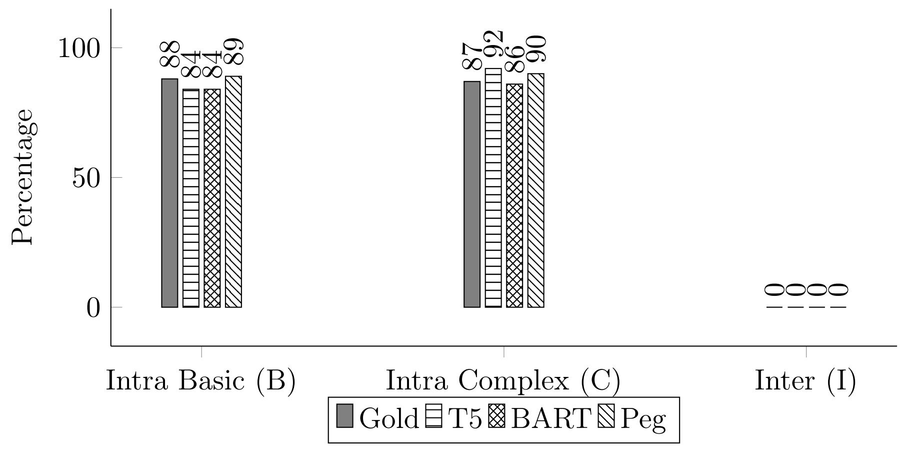


Figure 5.7: Profile of Automated Systems on SumTime Dataset

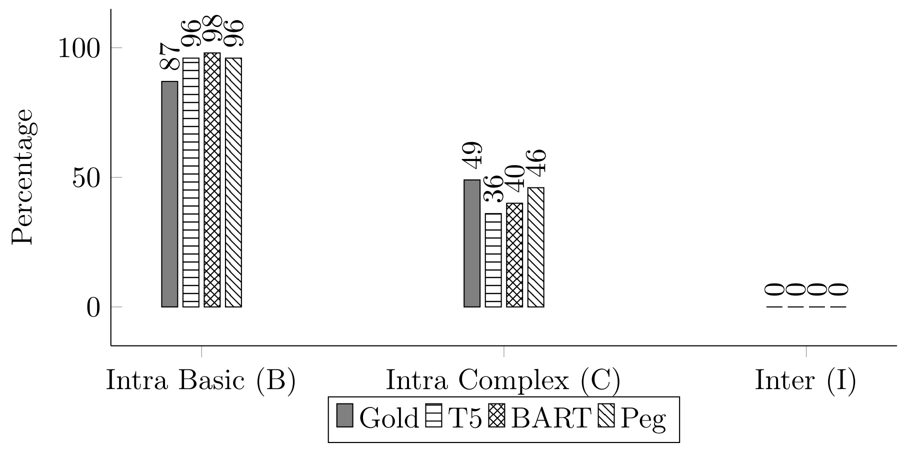


Figure 5.8: Profile of Automated Systems on Obituary Dataset

and employs general NLG models. Overall, error-rate across all datasets and generation systems in intra-event basic content is lower than intra-event complex or inter-event (where present) contents.

In the case of **intra-event complex contents**, almost all of the systems across all datasets have around 40 - 60% error rate. This shows systems struggle to learn the domain specific relationships required to derive complex information from the supplied data. Only **Ent** system in MLB and **Plan** system in SportSett have notably lower than 50% errors. This is because these systems are generating comparatively lesser intra-event complex contents compared to other systems (as can be seen in Figures 5.5 and 5.6). These two systems seems to be improving accuracy by producing less complex contents which are easier to generate.

All the **inter-event contents** in MLB are incorrect while the error-rate in SportSett for inter-event content is also very high. This is not surprising as the input to these systems doesn't take data from across the event stream into account during run-time. SportSett has actually produced some accurate inter-event content by producing standard phrases learnt from the training data (e.g. "team X has won four out of last five games") that turns out to be correct sometimes. All these results clearly demonstrate the difficulty generation systems have in producing complex content types.

We also show the content-type distribution in the system generated along with human reference test-set summaries in Figures 5.5 to 5.8. It can be observed that generations on SumTime and Obituary are able to generate similar amount of intra-event basic & complex contents as in human written summaries, however, with inaccuracies (as noted earlier). MLB and SportSett generated texts have different content-type distribution compared to human reference summaries. If we look at the the generations of Plan system in both datasets, it has the lowest inter-event content in MLB and lowest both inter-event & intra-event complex contents in SportSett. This might be because of the macro-planning design of the system which restricts the generations by only producing information explicitly available in input data. Another pattern can be observed in SportSett, where the system generated summaries have relatively more inter-event content than human written summaries, which we explore in the next section.

5.4.3 Concept Drift in SportSett

We further apply our proposed content-type profiling methodology to extract insights on the concept drift issue in SportSett. The dataset contains NBA games from season 2014 to 2018 and follows a seasonal partition to generate the train/test/valid splits. Seasons 2014, 2015 and 2016 are used for training set while 2017 and 2018 are used for validation and test sets respectively. In Figure 5.9, the content-type distribution of summaries by year is shown. The summaries from earlier years contain greater amount of inter-event content while comparatively little in later years. Even with intra-event content, there is less intra-event basic content in summaries from earlier years than later ones, indicating that summaries in the training set are more complex than in the validation and test sets.

This discrepancy explains the observed distribution of summaries generated from the different systems as shown in Figure 5.6. We can see the system generations have more inter-event content because they reflect the training data used which has more inter-event contents. Concept drift in this D2T problem is clearly identified by our dataset profiling method. This concept-drift can be explained with the change in authors in different years writing the summaries. In Figure 5.10, we also show two authors who wrote summaries in different years: ‘Auth1’ in 2014-15; and ‘Auth2’ in 2017-18. It is clear that different authors from different years have different distribution which may explain the concept-drift problem in the dataset.

5.5 Conclusion

In this chapter, we presented a typology of different content types in D2T summaries. The proposed typology is used to profile multiple datasets, which captures their characteristics and provide a measure of complexity present in them. Extensive experimentation is performed to demonstrate the challenge facing generation systems in producing complex

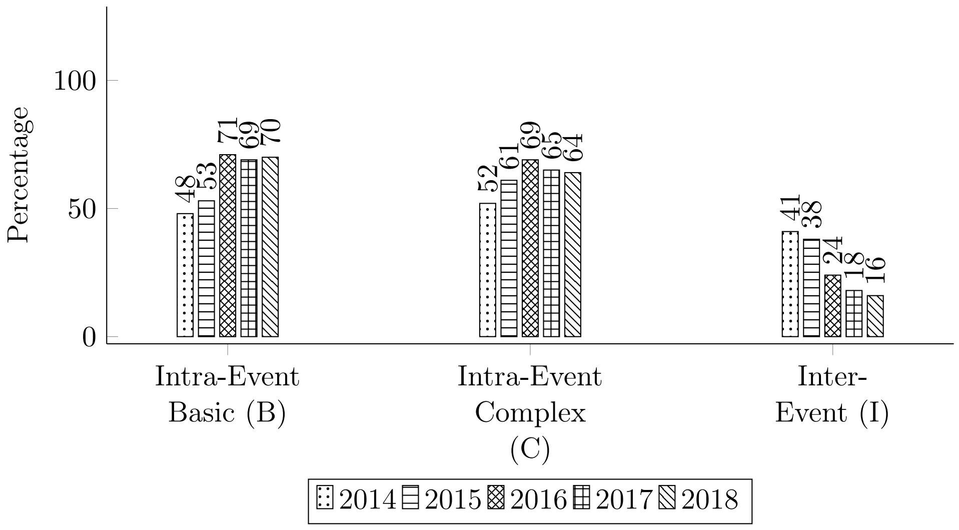


Figure 5.9: SportSett summaries' distribution by years

types of content. The dataset profiling method presented can help in building better systems capable of producing summaries closer to human authored summaries in terms of the summary's content type distribution.

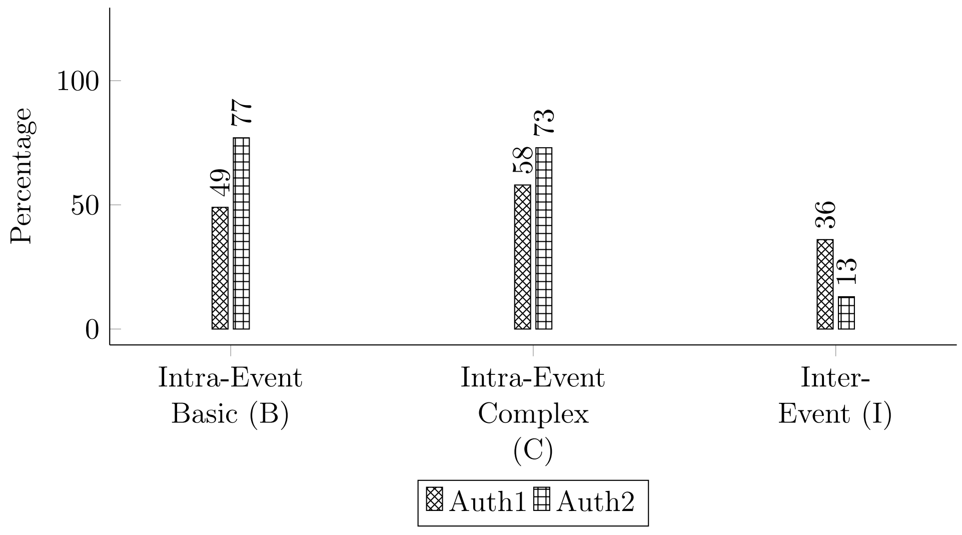


Figure 5.10: SportSett summaries' distribution by authors

Table 5.4: System-wise error-rates of different datasets categorised by content-types (lower is better; ↓)

Systems	Intra Basic ($\mathcal{B}$)	Intra Complex ($\mathcal{C}$)	Inter ($\mathcal{I}$)
	MLB		
Ent (Puduppully et al., 2019b)	13.98	38.6	100
Plan (Puduppully and Lapata, 2021)	10.84	45.27	100
Total	12.4 ± 1.5	41.9 ± 3.3	100 ± 0
	SportSett		
Ent (Puduppully et al., 2019b)	13.64	61.29	86.96
Plan (Puduppully and Lapata, 2021)	6.72	27.54	77.66
Hir (Rebuffel et al., 2020)	16.72	51.79	91.59
Total	12.3 ± 4.1	46.8 ± 14.2	85.4 ± 5.7
	SumTime		
T5 (Raffel et al., 2020)	38.1	50.57	-
BART (Lewis et al., 2020)	39.19	51.85	-
Peg (Zhang et al., 2020a)	35.95	48.45	-
Total	37.7 ± 1.3	50.2 ± 1.4	-
	Obituary		
T5 (Raffel et al., 2020)	2.74	41.67	-
BART (Lewis et al., 2020)	4.86	52.27	-
Peg (Zhang et al., 2020a)	9.09	48.39	-
Total	5.5 ± 2.6	47.4 ± 4.3	-

Table 5.5: Automated metric scores of different systems (↑, higher is better)

Systems	**BLEU**	**ROUGE-L**	**METEOR**	**chrF++**	**BERTScore**
			MLB		
Ent (Puduppully et al., 2019b)	11.51	22.08	27	32	85.03
Plan (Puduppully and Lapata, 2021)	13.99	21.71	32	39	84.6
			SportSett		
Ent (Puduppully et al., 2019b)	18.19	26.19	33	42	86.95
Plan (Puduppully and Lapata, 2021)	17.6	26.29	32	40	86.45
Hir (Rebuffel et al., 2020)	12.18	22.65	33	41	85.74
			SumTime		
T5 (Raffel et al., 2020)	24.67	52.92	38	47	89.66
BART (Lewis et al., 2020)	18.77	47.61	33	46	88.82
Peg (Zhang et al., 2020a)	23.54	51.06	39	48	89.68
			Obituary		
T5 (Raffel et al., 2020)	47.3	65.38	64	67	94.04
BART (Lewis et al., 2020)	50.88	65.99	66	69	94.29
Peg (Zhang et al., 2020a)	45.03	66.4	61	65	93.73

Table 5.6: Extractive Evaluation results of systems developed for MLB and SportSett dataset (↑, higher is better)

Systems	**RG**	**CS-Precision**	**CS-Recall**	**CO**
		MLB		
Ent (Puduppully et al., 2019b)	81.1	40.9	49.5	20.7
Plan (Puduppully and Lapata, 2021)	94.4	40.8	54.9	21.8
		SportSett		
Ent (Puduppully et al., 2019b)	72.77	45.35	38.12	19.13
Plan (Puduppully and Lapata, 2021)	86.48	53.95	33.09	14.84
Hir (Rebuffel et al., 2020)	73.77	45.42	30.15	10.59

Chapter 6

Context-Aware Content Planning

In the previous chapters, we identified that D2T summaries are a mix of different content types derived from different sources in the input data stream. We also presented a modular CBR methodology for data-to-text generation that mitigates the accuracy vs diversity trade-off present in neural and template based systems. However, the proposed CBR-D2T methodology doesn't have a specific module for planning the summary content and thus may fail to achieve the desired content type distribution in the system generated summaries.

In this chapter, we present a CBR based content planning methodology to generate the content plan for an event summary to ensure a representative distribution of content types and concepts that are closer to that of human written ones. We also identify that current D2T approaches only take current event's data into account for content planning or summary generation and ignore the wider environmental context in which the events take place. Thus, our presented CBR based content planning method utilises information from the wider environmental context and produces a contextually-aware content plan for a given event.

6.1 Motivation

Standard approaches to the D2T problem consider only data from the current event when creating the summary. This approach can work well where each event is independent of each other and not affected by the wider environment in which it takes place. However in many real life scenarios the context in which an event takes place is important and

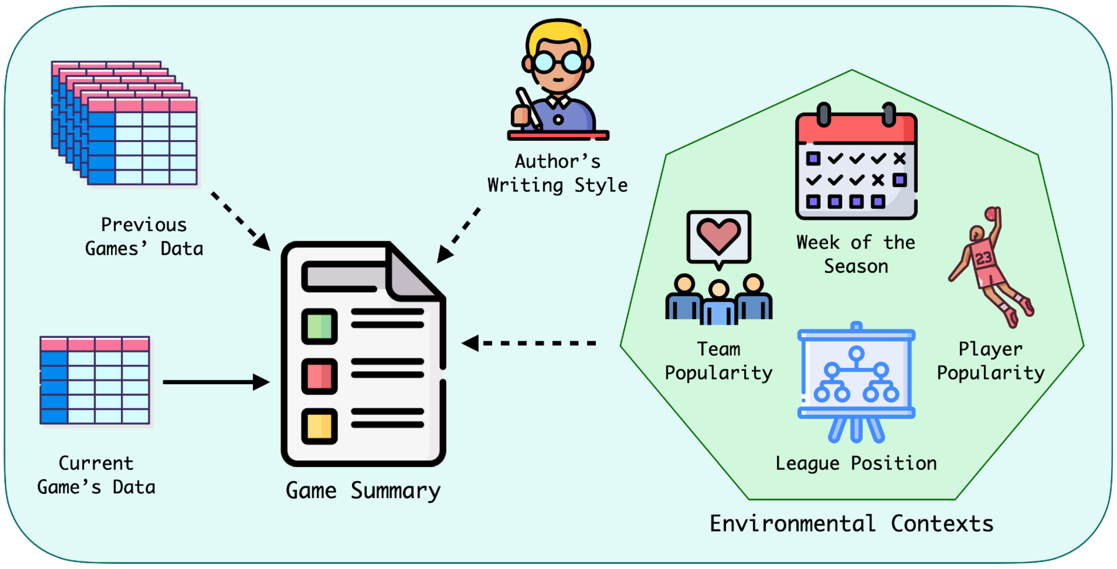


Figure 6.1: Knowledge sources contributing to a basketball summary.

has an influence on the summary that should be generated (Thomson et al., 2020b). For example, the content of a basketball event's summary is dictated by not just the event's data but also statistics from previous games as well as environmental contexts, such as a player's or team's popularity (see Figure 6.1).

In NLG problems context-awareness can be considered either in relation to the environment in which the event takes place, or from the perspective of the reader and their personal opinions and preferences. While, Dudy et al. (2021) consider context in relation to personalisation to match user preferences, in this work context is considered in relation to the wider environment in which an event takes place and we aim to establish the importance of environmental context to the content of generated summaries. For instance, a reporter writing a sports summary will not only take the game statistics into account but also their general knowledge of the sport. Sometimes this knowledge is based on the popularity of teams and players; other times, it might be based on the surrounding context of the game, such as its importance to final league positions. Current D2T systems, by ignoring wider environmental context fail to include important content in the summaries they generate.

D2T is a multi-stage process with several sub-tasks. First, each domain requires its specific domain model to identify the entities (or objects in broader terms) and their properties involved. This domain model is used to represent the input data of an event in the problem domain. For example, in a sports domain, there will be at least two types of entities: players and teams; each with their specific properties. Next, the D2T systems typically need to address two main challenges: **content planning** - selecting and ordering appropriate content from the event's input data, as in *what to say* and in *which order*; and **surface realisation** - generating text from the selected content, as in *how to say* (Puduppully and Lapata, 2021; Wiseman et al., 2017). These systems are evaluated on their ability to address these two challenges. The content planning ability is assessed on the closeness of their selected content to the human written summaries; while surface realisation is assessed based on the accuracy, fluency, and diversity of their generated text.

We focus now on the content planning task with complex D2T datasets that require longer summaries of an event to be generated. The idea is to use environmental context during the content planning phase and then to empirically demonstrate its benefit on SportSett (Thomson et al., 2020a), a sports domain D2T dataset. This is done by identifying different specific contextual themes that contribute to summary writing and utilise them in our content planning system for D2T. The experimental results suggest that 'player popularity' and 'team popularity' along with 'the week of the season' and 'team's position in the league', all have some influence on the content of game summaries. In addition to these themes, the author's writing style also has a big impact, suggesting different authors choose to present different views of the same event in a summary.

6.2 Background

Human written summaries contain three types of content based on their complexity Figure 6.3:

1. **Intra-Event Basic (B)**: information that is directly copied from the input data of an event;
2. **Intra-Event Complex (C)**: information that is derived from multiple records of an event's data; and
3. **Inter-Event (I)**: information that is derived from records of multiple events.

The domain model of sports domains contains two types of entities: players (**P**) and teams (**T**). Thus, based on the entities described in a summary sentence, it can be classified into following five categories: just one Player (P); just one Team (T); more

TEAM	WIN	LOSS	PTS	FG_PCT	REB	AST	...
Pacers	4	6	99	42	40	17	···
Celtics	5	4	105	44	47	22	···

PLAYER	H/V	AST	REB	PTS	FG	CITY	···
Myles Turner	H	1	8	17	6	Indiana	···
Thaddeus Young	H	3	8	10	5	Indiana	···
Isaiah Thomas	V	5	0	23	4	Boston	···
Kelly Olynyk	V	4	6	16	6	Boston	···
Amir Johnson	V	3	9	14	4	Boston	···
Avery Bradley	V	5	8	13	5	Boston	···
James Young	V	1	3	12	5	Boston	···
...							

The **Boston Celtics** *defeated* the host **Indiana** Pacers **105-99** at Bankers Life Fieldhouse on Saturday. ... **Boston (5-4)** has had to deal with a gluttony of injuries, but they had the fortunate task of playing a team just as injured here. **Isaiah Thomas** *led the team* in scoring, totaling **23** points and **five** assists on **4–of–13** shooting. He got most of those points by going **14–of–15** from the free-throw line. **Kelly Olynyk** got a rare start and *finished second* on the team with his **16** points, **six** rebounds and **four** assists. **Avery Bradley** also tallied **13** points, **eight** rebounds and **five** assists, while **Amir Johnson** finished with **14** points and **nine** boards. Boston will return to action on Monday against the New Orleans Pelicans. **Indiana (4-6)** had a tough task here ***without Paul George*** and they simply couldn't overcome that loss. ...

Figure 6.2: Excerpt of a basketball game's summary from SportSett dataset corresponding to its statistics table. Most information in the summary is explicitly available in the statistics table (**bold-faced**) or can be implicitly derived from the available data (*italics*). However, the content of the summary is also affected by information that is not available in data. For example, Paul George (***bold-faced and italics***) is mentioned in the summary even though he didn't play the game while James Young isn't mentioned even with a decent performance. The reason behind this could be the fact that Paul George was a much popular player than James Young at the time of writing the report.

than one Players (*P&P*); more than one Teams (*T&T*); and finally, both Players and Teams (*P&T*). We propose the idea of using sentence structure of event summaries to represent its content plan. The sentence structure, termed as a 'concept', highlights the type of content and entities described in the sentence. A content plan of an event is given as the list of concepts each denoting the sentence structure for its summary. Considering an example from Figure 6.4, the first sentence of the summary describes two teams from the game with their Intra-Event Basic and Intra-Event Complex type information. Similarly, the last sentence should describe a player and a team with their Intra-Event Basic, Intra-Event Complex and Inter-Event type information.

Sentences in a summary can be classified based on the type of information it contains into one of the seven categories: just Intra-Event Basic, **B**; just Intra-Event Complex, **C**; just Inter-Event, **I**; both Basic and Complex, **B&C**; both Basic and Inter, **B&I**; both Complex and Inter **C&I**; and finally, all Basic, Complex and Inter, **B&C&I**. In addition

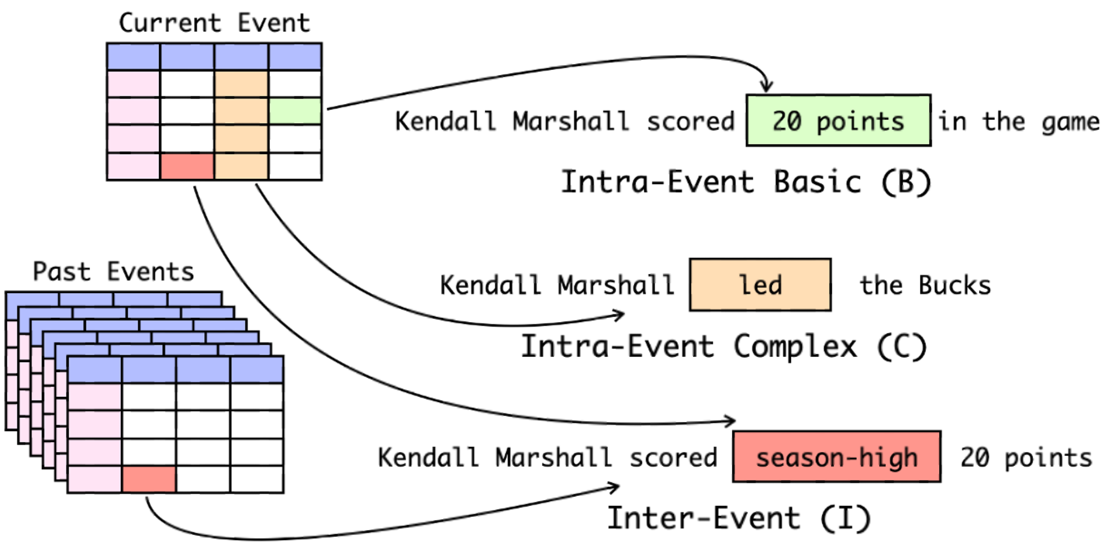

Current Event
Kendall Marshall scored 20 points in the game
Intra-Event Basic (B)
Past Events
Kendall Marshall led the Bucks
Intra-Event Complex (C)
Kendall Marshall scored season-high 20 points
Inter-Event (I)

Figure 6.3: Different types of content in a D2T summary

to different types of information in the summary, each sentence can describe different types of entities: a **Player (P)**; and a **Team (T)**. Thus, a sentence, based on the entity it describes, can be classified into the following five categories: just one Player, **P**; just one Team, **T**; more than one Players, **P&P**; more than one Team, **T&T**; and finally, both Players and Teams, **P&T**.

Extracting Content Plans from Human-Authored Summaries An event summary from the SportSett dataset, based on the information and entities a sentence describes, can be classified into a total of 35 concepts (7 types of information times 5 types of entities). We show the proportion of these concepts in our case-base in Figure 6.5. On $x - axis$, we see all possible concepts, and on $y - axis$, we show the number of sentences categorised as that concept. These statistics are calculated using an automated system that extracts the entities mentioned in a sentence and classifies the sentence into its information-type category. This system consists of two modules: first, an entity extraction module, the same as the method used in building train data for IE models in Wiseman et al. (2017); and second, an information-type classifier, which is a Roberta model Liu et al. (2019b) fine-tuned with a multi-label classifier head trained on 600 samples and tested on 250 manually annotated samples. The classifier achieves 91% of the Macro-F1 score.

Sentence	**Entities**	**Content Types**	**Concept**
Sixers came out in domination mode in the third and outscored Bulls, 37-18, to take a 102-76 lead heading into the fourth.	Team, Team	Complex	$T\&T - C$
Bulls put up a fight in the fourth but the Sixers were able to cruise to their first win of the season without a problem.	Team, Team	Complex, Inter	$T\&T - C\&I$
Joel Embiid led the Sixers with 30 points on 9-of-14 shooting, in 33 minutes of action.	Player, Team	Basic, Complex	$P\&T - B\&C$
⋮	⋮	⋮	⋮
Bobby Portis followed up with 20 points, 10 rebounds, two assists and two steals, while Antonio Blakeney added 15 points, five rebounds and two assists.	Player, Player	Basic	$P\&P - B$

Content-Plan: $\{\ T\&T - C,\ T\&T - C\&I,\ P\&T - B\&C,\ \cdots,\ P\&P - B\ \}$

Figure 6.4: Content-plan of a summary taken from SportSett dataset

Puduppully and Lapata (2021) propose the content plan as a list of paragraph-plans which identify the entities being discussed in the paragraph. However, their paragraph plans have one to many mapping where an identical plan (set of entities) can lead to multiple sentences in the summary. However, with our proposed content plan, each concept (equivalent to Puduppully and Lapata (2021)'s paragraph-plan) identifies the set of entities as well as the content type of the sentence, thus reducing the one-to-many a one-to-one mapping.

6.3 Methodology

In this section, we first demonstrate the process of building the case-base for content planning. Then our content planning methodology is defined to produce a solution (content plan) for a target problem (D2T event). We finally demonstrate the process of identifying environmental contextual features and adding them to the case's input data (problem representation).

6.3.1 Case Representation

The standard D2T datasets contain event information in structured format on the input side while the textual summary describing the event on the output side. We utilise the same organisation in building the case-base $\mathcal{CB}$ for content planning in D2T with n cases, where each case is an event. The structured input data of the event is used to build problem-side representation R while the solution-side content plan CP of the case

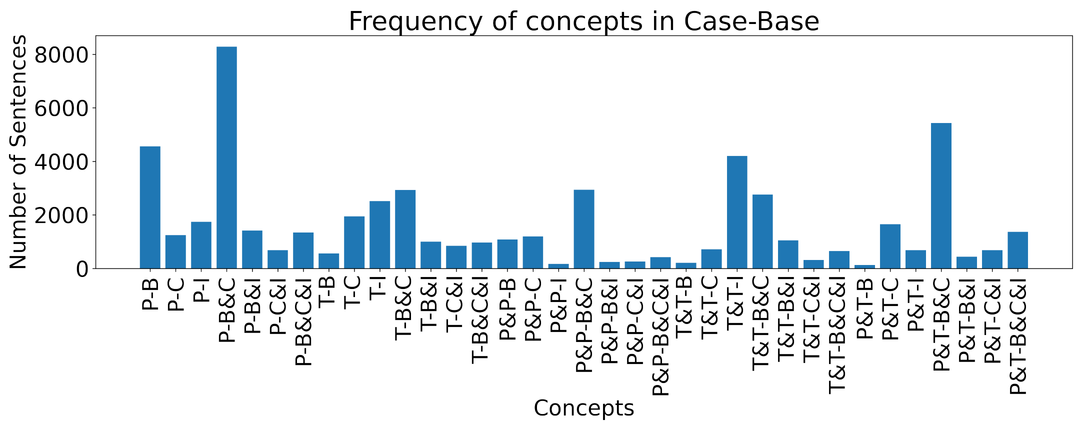


Figure 6.5: Frequency of concepts in the content planning case base of SportSett dataset

is extracted from the event summary.

$$\mathcal{CB} = \{(R_1, CP_1), \cdots, (R_i, CP_i), \cdots, (R_n, CP_n)\} \tag{6.1}$$

Problem-Side Representation:

The input data structure consists of multiple **entities**, ($\mathcal{O}$) which are the objects involved. Each entity is described by some core **features** ($\mathcal{F}$) which are the attributes of those entities.

Taking an example from the Figure 6.2, a case will be the basketball match, while the players and teams from the match are different entities $\mathcal{O}$. The different statistics such as PTS, REB, AST are the features, $\mathcal{F}$ describing these entities. Each case, $\mathcal{C}_i$, is represented using a vector $\mathcal{R}_i$ on the problem-side as follows:

$$R_i = (R_{ip}, R_{it}) \tag{6.2}$$

where R_{ip} and R_{it} are the vectors for player entities and team entities. These two vectors of different entities are concatenated to form case-representation vector and are calculated as follows:

$$R_{ie} = \{(\sum_{o=1}^{|\mathcal{O}|} F_{1,o,i})/|\mathcal{O}|, (\sum_{o=1}^{|\mathcal{O}|} F_{2,o,i})/|\mathcal{O}|, \cdots, (\sum_{o=1}^{|\mathcal{O}|} F_{f,o,i})/|\mathcal{O}|\} \tag{6.3}$$

where, R_{ie} is either R_{ip} or R_{it}, $F_{f,o,i}$ is the recorded value for f^{th} feature of o^{th} entity in the i^{th} case; and $|\mathcal{O}|$ is the total number of entities of that type (player, p or team, t) in the case.

Solution-Side Representation

The task is to produce a content plan of the event summary for a given event. The content plan is a list of concepts, each referring to the entity and content types to be described in a sentence. Considering an example from Figure 6.4, the first sentence of the summary describes two teams from the game with their Intra-Event Basic and Intra-Event Complex type information. Similarly, the last sentence describes two players with their Intra-Event Basic type information.

A case's content plan $\mathcal{CP}$ on the solution-side is given as follows:

$$CP_i = [ET_1 - CT_1, ET_2 - CT_2, \cdots, ET_n - CT_n,] \tag{6.4}$$

where ET_i is the entity-type and CT_i is the content-type described in the sentence i of the event summary (as shown in Figure 6.4).

6.3.2 Environmental Features

We now present different environmental contextual features that convey general knowledge of the domain which is not captured in the core features. The main idea is that some key environmental information about the events will be helpful in organising the summaries so that they are closer to the human written ones. There is a variety of contextual-aware information that could be included. By clustering summary sentences from the corpus, a few regular features that provide contextual environmental information were identified. Each of these themes are added as extra features to the event representation R_i in Equation (6.3). A pictorial example of final event representation, R_i, is shown in Figure 6.6.

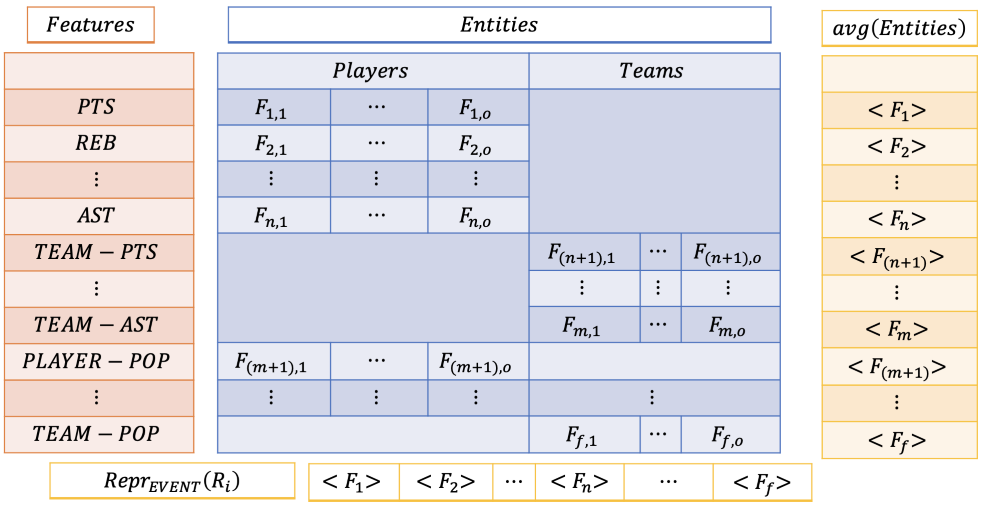


Figure 6.6: Problem-side representation (R_i) of Content Planning case-base

In sports domain, we identify the following environmental contextual themes: Player Popularity (PP); Team Popularity (TP); Team's League Position (Pos); Week and Day of the Season (W). Taking an example of player popularity, the event's problem-side representation will now have an extra feature named PP, with a floating point value between 0 to 100.

Player Popularity (PP):

Popular players get more coverage than a newcomer. Thus using entities popularity can help building better content plans for automated summary generation systems. Each entity's popularity is measured numerically by calculating the proportion of human written summaries mentioning them.

$$PP_p = M_p/P_p \tag{6.5}$$

where M_p is the number of summaries a player is mentioned in, and P_p is the number of games a player p has played.

Team Popularity (TP):

A more popular team might get better coverage in the summary than a less popular one. Thus, the popularity is calculated as the average of ratio of number of sentences they are mentioned out of the total in a summary.

$$TP_t = \frac{\sum_{i=1}^{tg}(M_{t,i}/T_{t,i})}{tg} \tag{6.6}$$

where $M_{t,i}$ is the number of sentences with the mention of a team t in the summary of i^{th} game, while $T_{t,i}$ is the total number of sentences in that summary. tg is the total number of games from the train set in which the team has played in.

Day and Week of the Season (W):

The day or week of the season may have impact on the content of the game summaries. When the season reaches towards the end or start of play-offs, the summaries start discussing more about teams' possibility of making the play-offs. External information on week of the season can be easily estimated from a game's date and the season's start/end date [1].

Team's Position in the League (Pos):

A team's position in the league may affect the content of the summary. For example, if a team is highly ranked or is fighting for a play-off berth in the season, then it might be discussed a lot more than usual in the summary. Conversely, if the team is not already out of the play-off contentions or is low ranked then the players might be discussed more in the summary than the team itself [2].

6.3.3 Content Planning Methodology (CBR-Plan)

The task here is to produce a content plan of the event summary for a given event for which a case-base is built using core and environmental features. We now present our CBR methodology to produce the content plan for a new event which reuses the solution

[1]Day and week information was downloaded from https://www.basketball-reference.com

[2]Team's match-by-match league position information was downloaded from https://www.basketball-reference.com

of k-most similar cases retrieved using a weighted similarity measure. Our methodology is demonstrated in Figure 6.7.

Retrieval and Reuse:

Euclidean distance metric is used to measure the similarity for retrieving the top-k most similar cases from case-base. After retrieving top-k nearest neighbours, their corresponding content-plans is ranked according to their lengths. The content-plan on the median position is then used as the content-plan of target event.

Important Entity Selection:

Not all entities in the event are equally important. In fact, in sports domain, only a handful of them are even mentioned in the event summary. Thus, an entity selection step is also performed to only use the important entities from the event to build its representation. This is achieved by building an *entity importance* classifier using the entities data from the event that learns to predict if an entity should be mentioned in the summary or not (this is similar to the entity importance classifier in Section 4.2.2).

Learning Feature Weights:

We also learn weights for each feature using the case-alignment feature weighting method from Chapter 4. This method learns the optimal feature weights using a Particle-Swarm Optimisation algorithm (Kennedy, 2010) with case-alignment metric from Chapter 4 used as the loss function.

6.3.4 Entity Selection

The next step in content selection and planning is to select the entities (or combination of entities) that should be described in each of the selected concepts from the previous stage. This is achieved by ranking all the different types of entities in a stack where the highest-ranked entity will be described using the first concept of its type, the second-highest ranked entity will be described with the second concept of its type, and so on. Thus, an algorithm is needed to rank the entities of an event. This can be achieved by learning the feature weights of the entity's representation, which can be used to score the entities and rank them based on the scores. To formalise:

$$Repr_{ENT} = [f_1, f_2, f_3, \cdots, f_n]$$
$$W = [w_1, w_2, w_3, \cdots, w_n], \forall w \in (-1, 1)$$
$$Score_{ENT} = \sum_{i=1}^{n} f_i \cdot w_i$$

The feature weights W are calculated using a PSO algorithm Kennedy (2010) optimised on a classification dataset for both the entity types (players and teams). For team entities, the classification data is prepared by subtracting the losing team's representation from winning team's representation and assigning it the label 1 (or win), and vice-versa for label 0 (or lost).

$$(Rep_{clf})^i = [(f_{1W} - f_{1L}), (f_{2W} - f_{2L}), \cdots, (f_{nW} - f_{nL})]$$
$$(Rep_{clf})^j = [(f_{1L} - f_{1W}), (f_{2L} - f_{2W}), \cdots, (f_{nL} - f_{nW})]$$
$$Lab^i = 1 \& Lab^j = 0$$

Similarly, win-loss data can be created for player entities as well where a player mentioned in the respective event summary will be considered a winner compared to a player from the event not mentioned in the summary.

6.3.5 Author's Writing Style

Different authors may either consciously or unconsciously present a different view of the same event, choosing to discuss different information from the same game. Thus by including information of author's writing style in system development, it may be possible to build better systems that reflect specific styles. To include this information, we separate the data from different authors into different training sets and only use the training set of the selected author in order to build the system summary. The system can then be evaluated on specific author's data. So while the training set is more focused on author style, the trade off here is that less data is available as the training sets size is reduced.

6.4 Experiment Setup

The experiments compare the CBR-Plan method against different neural D2T benchmarks on a standard D2T dataset from sports domain. An ablation study is also performed to understand the effect of environmental features in CBR-Plan method.

6.4.1 Dataset

The SportSett dataset (Thomson et al., 2020a) of NBA matches is used for experimental evaluations [3] in which a match is an event or an instance in the dataset. Each match contains the event summary and the associated match statistics, with the box- and line-scores. We use the method described in Section 6.2 to extract the content plan from these summaries and prepare a case-base. The problem-side input representation of a case in the case base is calculated using the box- and line-scores of the event's input data as described in Section 6.3.1. While the solution-side representation is the content plan extracted from event summary. The case base consists of total 3690 cases, which is the training set of SportSett dataset.

6.4.2 Baseline and Benchmark Systems

CBR-Plan is compared with a template based baseline as well as three state-of-the-art neural D2T systems, as follows:

- **Template (Temp)**: the baseline model proposed in Wiseman et al. (2017) which contains a few handcrafted templates to verbalise the data of a few entities from an event. In this work, an updated version of this model is used which adds a few more templates for generating extra information (next-game information of a team).
- **Entity Model (Ent)**: an entity-focused approach (Puduppully et al., 2019b) uses a sequence-to-sequence model consisting of an MLP encoder and LSTM decoder with copy mechanism. An added module updates the input record's representation during the generation process. At each decoding step, a GRU is used to decide the record that needs to be updated and then update its value.
- **Hierarchical Transformer (Hir)**: is a sequence-to-sequence model with a hierarchical transformer as encoder and a LSTM decoder (Rebuffel et al., 2020). The

[3] GEM version (Gehrmann et al., 2022) of the dataset is used from `https://huggingface.co/datasets/GEM/sportsett_basketball`

hierarchical encoder first encodes the each entity in an event using its features and then encodes all the encoded entities to generate the final event representation.

- **Macro-Plan Model (MP)**: a neural pipeline model for data-to-text generation proposed in (Puduppully and Lapata, 2021). It consists of two separate modules: first, a micro-planning module, which takes all the entities as input and selects and orders the important entities using a Pointer network to build a macro-plan. The second module is an LSTM based sequence-to-sequence text generator with a copy mechanism to generate a summary from the macro-plan.

All the models are trained using the same hyper-parameters as described by the authors in their respective works.

6.4.3 Evaluation Metrics

Performance is measured on two important dimensions of data-to-text generation: **content selection** (**CS**) and **content ordering** (**CO**) (Wiseman et al., 2017). Both dimensions are measured for each system by comparing their content-plan against the content-plan extracted from human-authored (gold) summaries. The content-plan for benchmark systems is extracted from their generated summaries in the same way as from human-authored gold summaries.

The **content selection** is evaluated by measuring the **F1 score** between the content-plan of each system and the gold summaries. Along with F1 scores, we also show the precision, and recall scores. For measuring content ordering ability of different systems, **normalised Damerau-Levenshtein Distance (DLD)** between the system generated content-plan and gold summaries' content-plans is used from Wiseman et al. (2017).

6.5 Results and Discussion

The experimental results are discussed in two stages:

- first, the baseline content-planning method (CBR-Plan) is analysed and compared against different neural benchmark systems; and
- second, the effect of authors is investigated by building separate systems for different authors;

The headline result from this chapter is: CBR-Plan is able to produce content plan for

Table 6.1: CBR-Plan versus baseline and benchmark systems

System	CS F1	CS Prec	CS Rec	CO DLD	Len Avg
Gold	-	-	-	-	12.76
Temp	29.06	37.81	23.6	6.09	7.97
Ent	24.86	24.05	25.73	10.94	13.66
MP	28.63	33.13	25.2	8.8	9.71
Hir	33.38	34.45	32.37	11.27	11.99
CBR-Plan	**35.51**	**37.96**	**33.36**	**13.13**	11.22

Table 6.2: Effect of different environmental contextual themes

CBR-Plan	CS F1	CS Prec	CS Rec	CO DLD	Len Avg
Gold	-	-	-	-	12.76
No Context	32.02	33.85	30.39	11.06	11.46
w/ Player Popularity (PP)	33.1	33.61	32.6	9.68	12.39
w/ Team Popularity (TP)	36.87	41.34	33.27	15.81	10.27
w/ Week (W)	32.58	33.5	31.72	11.32	12.09
w/ League Position (Pos)	31.04	32.15	30.0	14.12	11.91
with All	35.51	37.96	33.36	13.13	11.22

summaries that are more closer to content plans from human authored summaries than benchmark neural systems.

6.5.1 Content Selection and Ordering

The results of evaluating content selection (CS) and content ordering (CO) capabilities for the different systems is shown in Table 6.1. The first thing to observe is that our CBR-Plan model outperforms every other baseline and benchmark system on both CS and CO scores. We observe +2 CS-F1 points improvement than the best bechmark (Hir) and +6 CS-F1 points than baseline (Temp). CBR-Plan also gains 1.9 points on CO-DLD than Hir and +7 CO-DLD points than Temp. The reason for Temp performing worse than all other models in every case could be because the Temp system only contains a few handcrafted templates to verbalise the hand-picked data and is not enough to change the plan with a change in data.

Table 6.3: Correlation between concept frequency of different systems versus gold

System	CBR-Plan$_{euc}$	Temp	Ent	Hir	MP
Correlation	0.7380	0.6281	0.3105	0.6153	0.5954

We also analyse the effect of different contextual themes on content planning separately Table 6.2. The First thing we observe in this experiment is that adding contextual themes improves performance of the content planning across CS & CO metrics, except with Pos for CS. The models with contextual themes also achieve comparable performance for average concept length. We observe 1 point CS-F1 improvement with PP, while +4 point improvement with TP, and 1.5 point improvement with W. In the case of CO-DLD, we see improvements with all themes except with PP.

When all these environmental contextual themes are combined to enrich the problem representation, then we observe much better gains across all metrics. We observe 3.5 point gain on CS-F1, +2 point gain on CO-DLD. These results clearly demonstrate the positive effect of adding contextual information for the content planning task. The contextual information helps the system in understanding the types of entities and content to discuss in the summaries.

6.5.2 Content Type Distribution

We also compare the content type distribution of the plans generated by our system with the plan generated by benchmarks. We also show the distribution of human written summaries from test set (Test) and training set (Train). The results are shown in Figure 6.8.

We observe a drop in the inter-event content percentage in test set compared to the train set. That's why all systems have more inter-event content in their plans than the test set summaries, as they are trying to emulate the training data. However, we also observe that both variants of CBR-Plan achieve similar distribution of content in all three types as the training set summaries. This shows that our content planning methodology is able to plan summaries with content much closer to the human written summaries.

6.5.3 Content Diversity

We also investigate the ability of different systems to select different concepts ($P - B$, $P - C$, $T - B\&I$ etc.) in their summary generations. In Figure 6.9, the frequency

Table 6.4: Different authors' writing style

	CS			CO	Len Avg	
Author	F1	Precision	Recall	DLD	Sys	Gold
All	35.51	37.96	33.36	13.13	11.22	12.76
Ben Miller	53.37	53.61	53.12	17.23	18.2	18.37
Juan Pablo	44.95	44.68	45.23	15.83	10.28	10.15

of different concepts being selected is shown for the human reference summaries and each evaluated model. The human reference summary has a relatively even distribution over all concept types. Intuitively, we would expect a well-automated system to generate summaries with a similar distribution over the concepts as the human-generated solution.

If we look at the distribution of the different systems, we can observe that Template system is only selecting a few popular concepts and completely ignoring the others. With the neural systems, while MP does select across many concepts there is still a popularity bias by heavily selecting the most popular concepts. For example, it is selecting mostly P-B, P-B&C, T-B, and T-B&C, which seem to be the most popular ones, but is missing many other important concepts. Similarly, Ent has a popularity bias in selecting high numbers of a few popular concepts.

CBR-Plan can select a broad range of concepts that is most similar to the distribution seen with human reference summaries. Along with selecting the most popular ones, CBR-Plan is also able to select non-popular but important concepts. We show Pearson's correlation coefficient of concept distribution in system generation versus human reference summaries in Table 6.3. CBR-Plan has the highest correlation with human reference summaries followed by Hir, MP and Ent. It is also surprising to see the Ent model performing so poorly on this measure, which appears to be due to the model selecting higher numbers of some rare concepts, e.g. T-B&I, and T-I.

6.5.4 Author's Writing Style

We also investigate the effect of different authors' writing style on generating content-plans. In the sports domain, different authors may have different writing styles: some may prefer to discuss the players' or teams' historical information along with the current game while others just discuss the current game. To analyse the writing style of two different authors (Ben Miller and Juan Pablo) who have written summaries in different NBA seasons from SportSett dataset; we plot their summaries' content-type distribution in Figure 6.10. On the x-axis, we show the number of sentences from the authors' discussing each type of information. It can be observed that Ben Miller likes to discuss twice as much inter-event content as Juan Pablo.

To investigate the effect of writing style on content planning in SportSett dataset, we build separate case-bases for both authors by using the summaries written by them. Results from the CBR-Plan system evaluated for each author separately is shown in Table 6.4. The results show a large improvement for custom authors' systems as compared to the common system. The CS-F1 score for Ben Miller's system performance improved to 53 from 35.51 while Juan Pablo's system performance improved to 44.95. Ben Miller's system also achieves 17.23 CO-DLD points while Juan Pablo's achieves 15.83 CO-DLD as compared to 13.13 points of all authors' system. These findings suggest that including different authors' writing style can have large positive impact on generating content-plans.

6.6 Conclusion

Current D2T systems do not take general contextual knowledge into account while generating summaries. However, contextual knowledge of the environment is very important when human authors write summaries in real-world domains. In this chapter, we identified five such general contextual themes: player's popularity, team's popularity, team's position in the league, the week of the season, and author's writing style; and proposed methods to utilise them in building the content plan for a D2T event summary. Through experimentation on the SportSett dataset, we demonstrated that these contextual themes help systems build better content-plans for summary generation. Our proposed methodology is empirically evaluated and compared against neural benchmarks on two dimensions of D2T evaluations: Content Selection and Content Ordering. The experimental results demonstrate the benefits of using environmental contextual themes in content planning to achieve content plans closer to that of human written summaries.

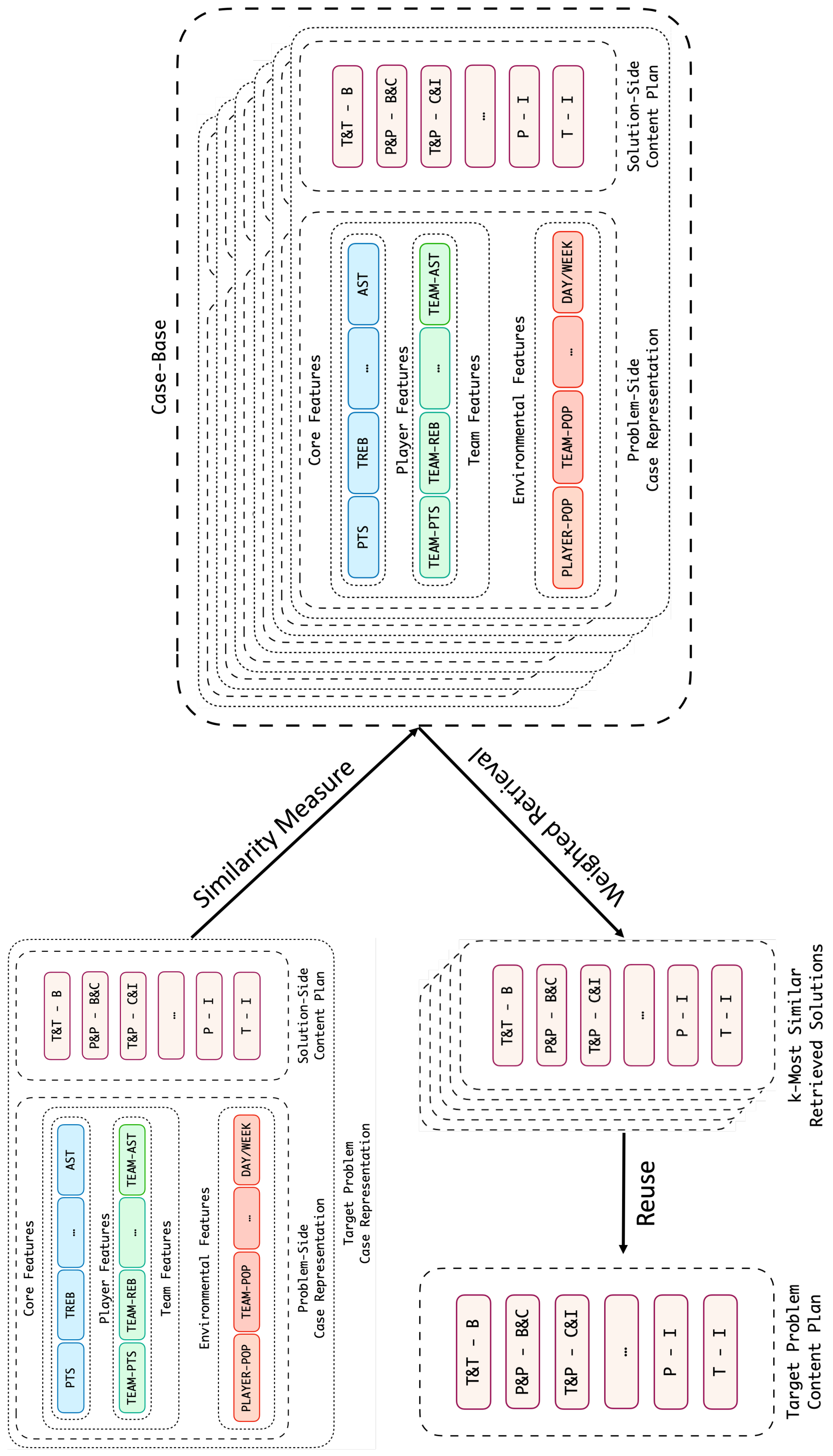


Figure 6.7: CBR-Plan methodology

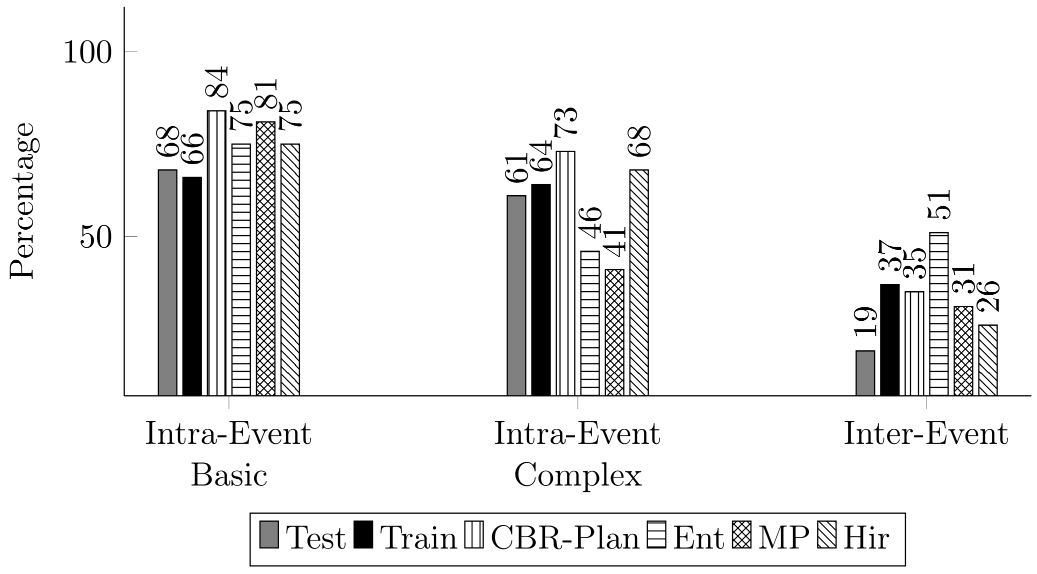


Figure 6.8: Content type distribution of content plans on SportSett dataset

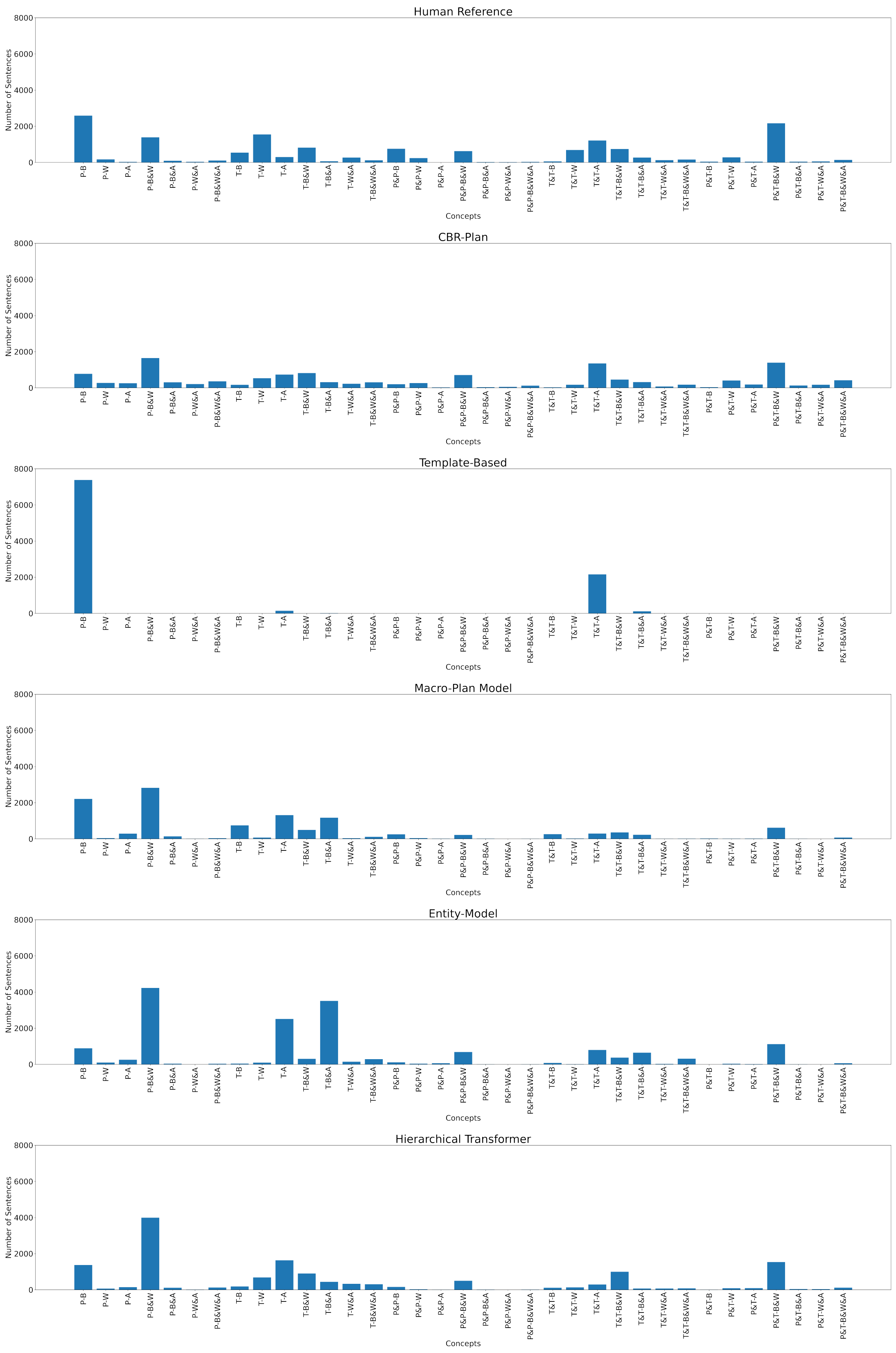


Figure 6.9: Proportion of different concepts in different systems

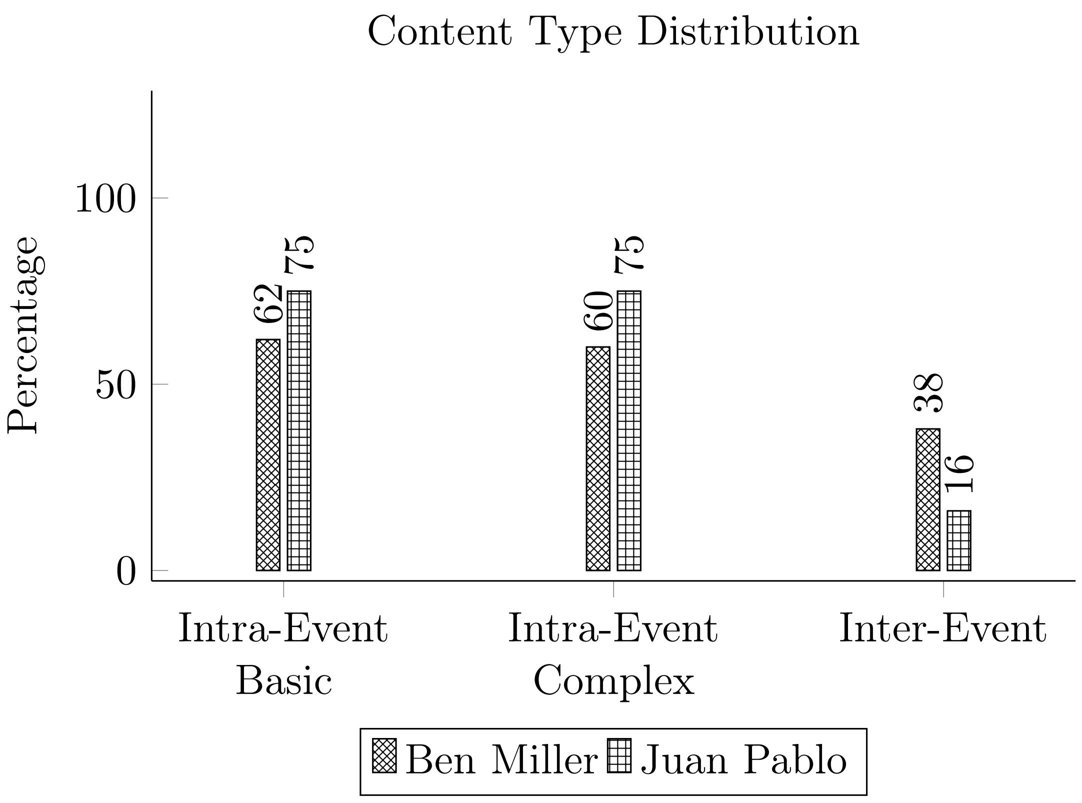


Figure 6.10: Content-Type distribution of summaries from different authors

Chapter 7

Context-Aware Surface Realization

In the previous chapter, we presented a CBR-based context-aware content planning method to produce a content plan for an event summary. This content plan is used to organise the summary content in such a way that it is as much closer to human authored summaries as possible. In this chapter, we are going to use that content plan to generate the final textual summary using a contextually-aware surface realization method. This method will be able to generate summaries with accurate inter-event content. Current state-of-the-art neural D2T systems struggle to generate content from past events with interesting insights. This is because these systems have limited access to historic data.

Another problem with neural solutions is that they tend to hallucinate, providing inaccurate information. In the proposed surface realisation method hallucination is reduced by a CBR-assisted context-aware surface realisation methodology that carefully selects important contextual data from past events and utilises a hybrid CBR-based dynamic template method together with a neural text generator to create the final event summary.

7.1 Introduction

We consider D2T problems that consist of a series time-stamped events where a textual summary is written for each event. The summaries are very rich and contain contextual information derived from past events in the time-series as well. The excerpt of a basketball summary shown in Figure 7.1, shows the contextual content derived from past event's data (bold-faced). For example, the information that it was Celtics' second victory Pacers can only be derived by processing past events data.

TEAM	WIN	LOSS	PTS	FG_PCT	R
Pacers	4	6	99	42	4
Celtics	5	4	105	44	4

PLAYER	H/V	AST	REB	PTS
Myles Turner	H	1	8	17
Thaddeus Young	H	3	8	10
Isaiah Thomas	V	5	0	23
Kelly Olynyk	V	4	6	16
...				

The Boston Celtics defeated the host Indiana Pacers 105-99 at Bankers Life Fieldhouse on Saturday. It was the **second victory** over Pacers for the Celtics this season **after emerging victorious in Boston 91-84 on Nov. 16.** ... Isaiah Thomas led the team in scoring, totaling 23 points and five assists on 4–of–13 shooting. Kelly Olynyk got a rare start and finished second on the team with his 16 points, six rebounds and four assists. ... **Boston will return to action on Monday against the New Orleans Pelicans**.

Figure 7.1: Excerpt of a basketball game's summary from SportSett dataset corresponding to its statistics table

An effective D2T system de Vries et al. (2020) must be able to generate summaries with contextual content from past events along-with content from the current event. This should be done in a way such that the distribution of different types of content in the summary is similar to that of human written ones. For this, the system must be able to process contextual data from past events and include them in generated summaries. Processing all past events data becomes a challenge for current neural D2T systems, as the amount of data that needs to be processed grows exponentially. Current neural D2T systems, have limited input size window and can not process input data of very high length. Thus a need for selecting important content from past contextual events arises that can be passed to these systems along-with the current event's data for summary generation.

We present a methodology for context-aware surface realisation that is able to: first, select important contextual content from past events; and second, include them in the final textual summary. For this, we first identify two themes that convey contextual information derived from past events and are commonly present in the human written summaries. Then a theme classifier is built to select the important content that can be included in the summary. Finally, this content is used by a hybrid CBR and neural D2T system to generate the final textual summary for an event.

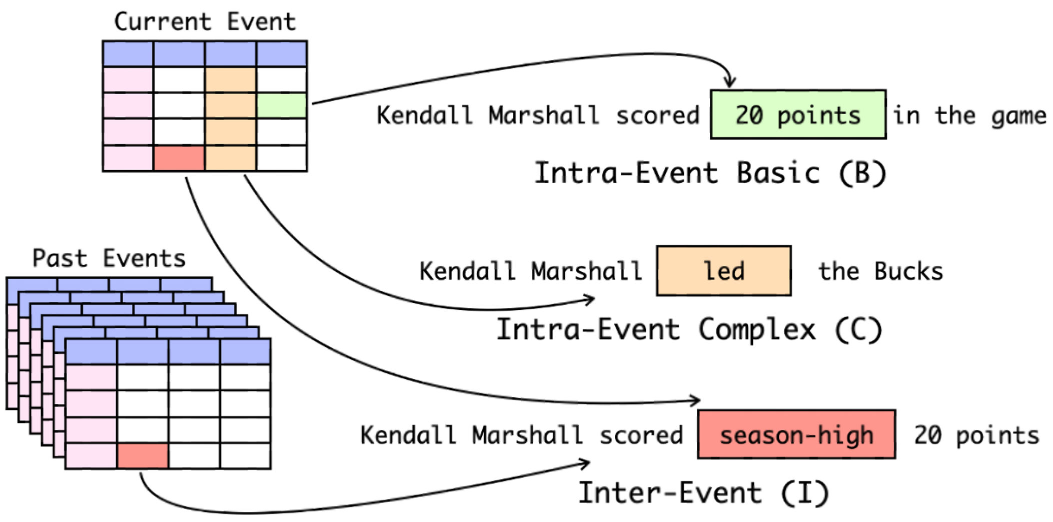

Current Event
Kendall Marshall scored 20 points in the game
Intra-Event Basic (B)
Past Events
Kendall Marshall led the Bucks
Intra-Event Complex (C)
Kendall Marshall scored season-high 20 points
Inter-Event (I)

Figure 7.2: Content types in D2T summaries (as defined in Chapter 5)

7.2 Background

We identified that content of the D2T event summaries can be classified into three categories: **Intra-Event Basic (B)**, fact directly copied from current event's input data; **Intra-Event Complex (C)**, fact derived from current event's input data; and **Inter-Event (I)**, fact copied or derived from other events' data (see Figure 7.2). We now propose a method to generate accurate content of Inter-Event type for D2T event summaries.

A CBR apporach to content planning in D2T was proposed in Chapter 6 where the plan is a sequence of concepts represented by the sentence structure of the summary. Figure 7.3 shows the content plan extracted from a basketball summary using this approach. The content plan is a list of concepts, each denoting a sentence structure conveying the entity and content type to describe in that sentence. As denoted, the first sentence in the summary should describe two team entities with intra-event complex type content.

Here, we take the content plan generated from this approach and then generate a final event summary according to the plan. The method works in two stages: first, inter-event data selection, where we use a machine learning technique to select important historical data; and second, surface realisation, where we use a hybrid neural and CBR system for text generation in order to generate accurate inter-event type content.

Sentence	Entity Types	Content Types	Concepts	Concepts Filled
Sixers came out in domination mode in the third and outscored Bulls, 37-18, to take a 102-76 lead heading into the fourth.	Team, Team	Complex	$T\&T-C$	Sixers&Bulls-C
Bulls put up a fight in the fourth but the Sixers were able to cruise to their first win of the season without a problem.	Team, Team	Complex, Inter	$T\&T-C\&I$	Sixers&Bulls-C&I
Joel Embiid led the Sixers with 30 points on 9-of-14 shooting, in 33 minutes of action.	Player, Team	Basic, Complex	$P\&T-B\&C$	Joel Embiid&Sixers-B&C
⋮	⋮	⋮	⋮	⋮
Bobby Portis is averaging 20 points and 10 rebounds on the season.	Player	Inter	$P-I$	$BobbyPortis-I$

Content-Plan:
{ $T\&T-C$, $T\&T-C\&I$, $P\&T-B\&C$, $\cdots$, $P-I$ }
Content-Plan with Entities Filled:
{ Sixers&Bulls-C, Sixers&Bulls-C&I, Joel Embiid&Sixers-B&C, $\cdots$, Bobby Portis-I }

Figure 7.3: Content Plan of an event summary

7.3 Methodology

Context aware surface realization takes place in two-stages. First, a contextual inter-event data selection module is used to select important inter-event data from previous related events; and second, a CBR-based dynamic template method is used with a neural surface realisation method to generate the final event summary with accurate inter-event content. The method is shown in Figure 7.4.

7.3.1 Contextual Inter-Event Data Selection

The contextual inter-event data selection is performed in three steps:

- **Finding Possible Inter-Event Themes**: The first step is identify the possible themes that convey inter-event information about entities from the event.
- **Building Resources for these Themes**: The next step is to develop a parallel resource for these inter-event themes that can be queried to get information during the run-time of an event summary generation.
- **Select the Inter-Event Themes to include in the summary**: Finally, for each

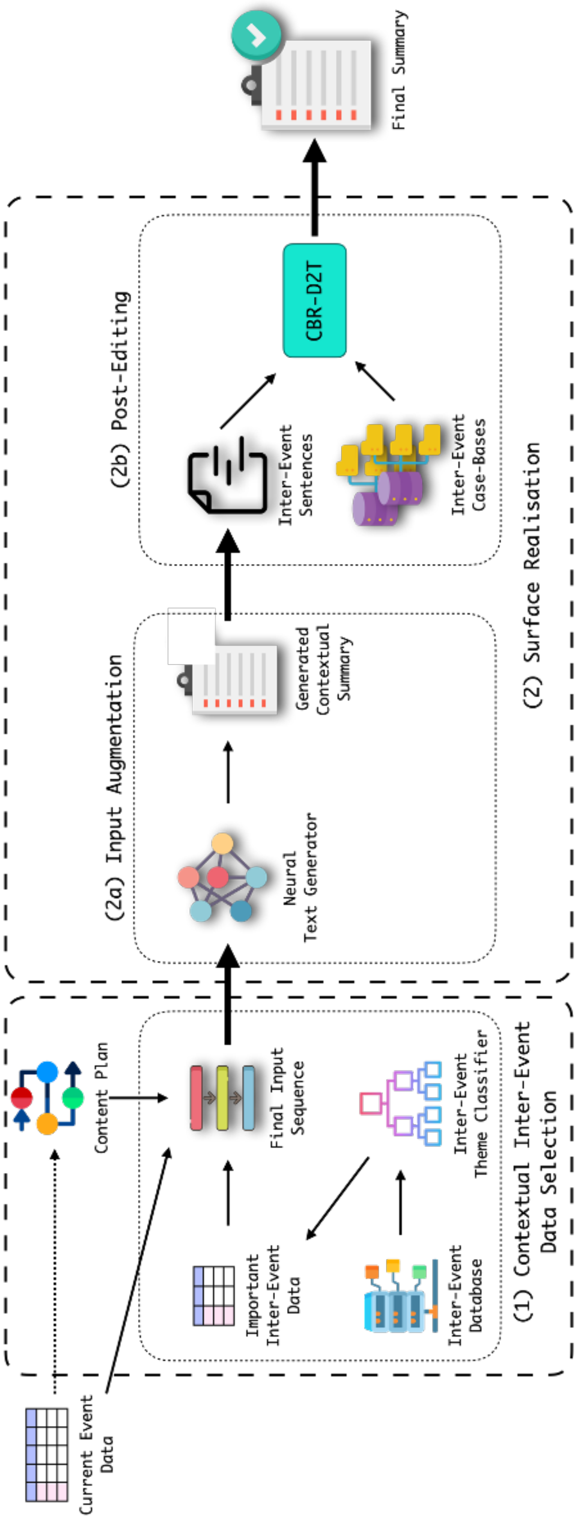


Figure 7.4: Context-Aware Surface Realisation

summary, during run-time, select the important inter-event features that could be included in the final textual summary.

In the next few subsections, we discuss the details of each of these steps respectively.

Finding Possible Inter-Event Themes

The first step in selecting inter-event data is to identify some popular themes that are commonly discussed in the event summaries. There can be broadly two approaches to do this: first, knowledge transfer, as in taking insights from domain experts; or second, data analysis performed on training data; can help us in identifying the common themes discussed in the summaries. Depending on the domain requirements and availability of resources, either approach is possible. Here, since we have access to a sizeable amount of training data, we choose the second approach and perform some analysis on the data to identify relevant themes.

To perform the analysis, we follow these steps:

- break the summaries into sentences and then classify the sentences into their content-types (as in Figure 7.2);
- take the sentences classified as containing 'inter-event' content and divide them in different entity types (in sports domains: players and teams)
- apply topic modelling on the sentences from each entity and select the top topics;

By this process, we select a dominant topic from each of the entity type, which are:

1. **Players' Average Stats**: player A is averaging X points in last Y games;
2. **Teams' Win/Loss Streak**: this was team B's j^{th} straight loss/win;

In our topic modelling, we also found some other common themes such as: player's total double-double scores of the season; or, the team's standing in the league/conference. However, we decide to experiment with only two themes selected above to keep the problem complexity simple and evaluate the idea properly.

Building Resources for these Themes

Once the inter-event themes have been identified, the next step is to build some parallel resources that can be used during run-time to query and get the information about a theme for an entity in an event. This parallel resource will store the inter-event information relating to the theme for each entity in the event.

We first identify a few inter-event features that will be used to represent the entities along with their intra-event features. For player average theme, the features are: *average/total X in last Y games*, where X ∈ (points, rebounds, assists, blocks, steals) and Y ∈ (2, 10). For the team streak theme, the features are: streak count, and streak type, where streak count ∈ (0, 82) and streak type can be win or loss.

After identifying these features, for each entity from every event in the dataset - we generate the values for these identified inter-event features and store them into a separate parallel resource for each theme (currently json, but a better choice could be a relational database). The process of generating the values for these features is as follows:

- **Filter**: filter all the events from time-series containing a given entity and happening before the current event;
- **Sort**: sort these events based on the timestamp in ascending order of time delta, where the most recent event is the closest; and
- **Aggregate**: aggregate all the relevant values of the entity feature into the identified inter-event feature;

For example, consider an event which is the 25th match of player Kevin Durant. To calculate his average points in the last 5 games: we first filter all the matches from this season in which Durant played and the match occurred before this one; we then sort these matches based on their date; and finally, average the number of points made by Durant in the most recent 5 games. So, for this event, the inter-event feature - *Player Average Stats in Last 5 Games*, where player is Kevin Durant, the value is averaged number of points.

Selecting Important Attributes

After building the parallel resource, the next step is to select the inter-event features from each theme that could be included in the final summary. This is done by training a binary classifier for each theme whose task is: given an inter-event feature for an inter-event theme, classify if it should be added to the final summary or not.

To build a theme classifier, an important step is to identify attributes needed to train these classifiers. Through our domain knowledge, we identify the following attributes for the two themes:

- Player average theme
  - **player name** - converted into a number using label encoding;
  - **player popularity** - calculated as the ratio of number of game summaries mentioning the player to the number of games the player has played;
  - **record type** - label encoded value for point, rebound, assist, steal or block;
  - **last Y games** - number of games totalling or averaging for (2 to 10);
  - **value** - actual value of the inter-event feature;
  - **average or total** - a binary attribute denoting if this average score or total score over last Y games.
- Team streak theme
  - **team name** - converted into a number using label encoding;
  - **team popularity** - ratio of the number of sentences mentioning the team to the number of sentences in the summary averaged over toal number of games in the season;
  - **streak count** - count of the streak;
  - **streak type** - a binary value denoting if this is a win or loss;
  - **broken streak count** - denoting if the team has been on a different streak than current result (if there has been a winning streak before if the current one was the lost game);
  - **broken streak type** - type of the broken streak.

We build the train and test set for these theme classifiers using the train and validation set of D2T dataset respectively.

We also show the feature correlation of these attributes from each theme classifier using information gain in Figures 7.5 and 7.6.

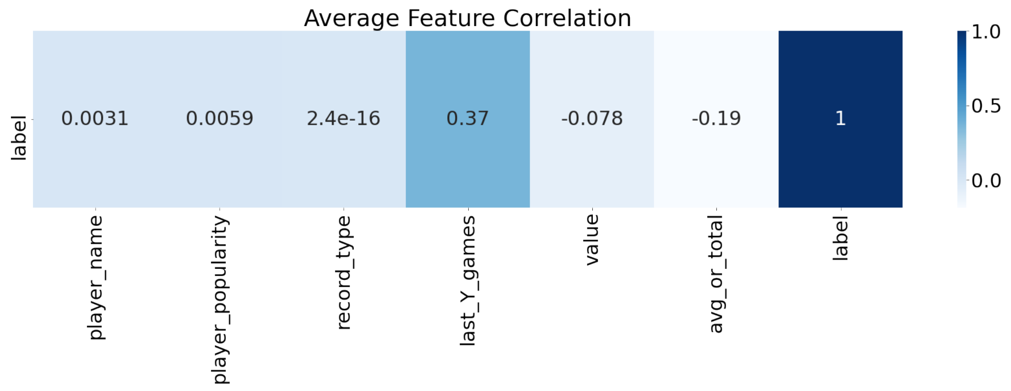


Figure 7.5: Feature correlation of Player Average Theme Classifier

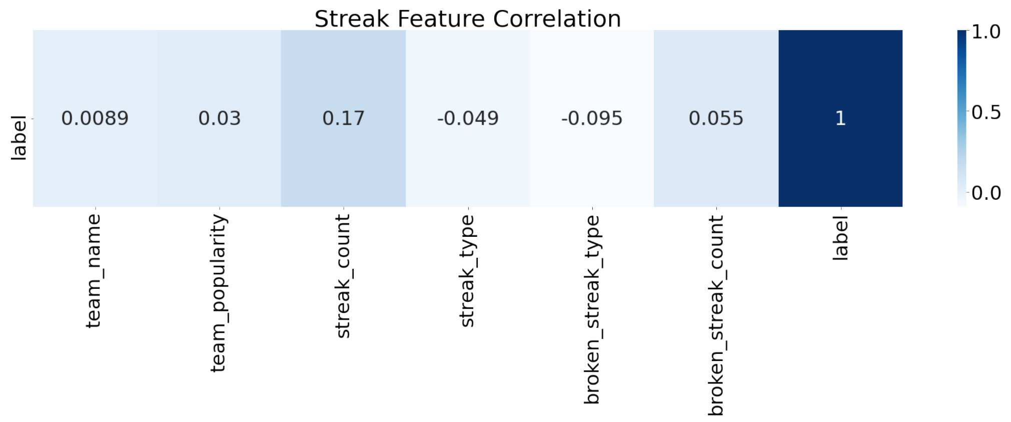


Figure 7.6: Feature correlation of Team Streak Theme Classifier

7.3.2 Surface Realisation

Now with the important inter-event data selected, we move on to using this data in accordance to the plan (derived using the method described in Chapter 6) to generate the final summary. This stage of text generation in D2T is known as surface realisation (part (2) in Figure 7.4). Earlier studies have shown that neural networks are capable of producing good enough content of intra-event types (both basic and complex), however struggle in producing content of inter-event type Chapter 5. Thus in this work, we propose two different methods to improve the inter-event content of summaries generated by neural systems.

- **Input Augmentation**: the first approach is to augment the input to the neural system by adding the content plan and selected inter-event content to its input and train the model to generate summaries with better coverage; and

- **Post-Editing**: the second approach is to further post-edit the output of the neural system by identifying the sentences with inter-event content and replacing those with sentences generated using the CBR-D2T dynamic template method from Chapter 4.

There could be other methods to build a surface realisation module that incorporates the content plan and selected information to generate a summary. One method is to build a separate case-base for each of the possible concepts identified in the content plan from Chapter 6 and then apply dynamic template method from CBR-D2T Chapter 4 to generate the summary. This method dynamically selects the most suitable template from the concept's case-base for each concept in the content plan and then populates the templates with the values of selected entities. However, in this work, we take a hybrid approach to utilises both neural and CBR systems for realisation.

Input Augmentation

The first proposed method is called **Input Augmentation**. It augments the input of a neural D2T system in accordance with the content plan from Chapter 6 and the selected inter-event content from the methodology described above. The current state-of-the-art in neural D2T uses a pipeline approach with separate planning and realisation phases (Puduppully and Lapata, 2021). The planning module outputs a sequence of paragraph plans, known as macro plan (which is similar to the concepts described in Section 7.2 except the paragraph plans only contain the entity information and not the content type information, these paragraph plans also only contain the intra-event data for the entities). The macro plan is then fed to the surface realiser, which is a neural sequence-to-sequence model, to produce the final summary.

In Input Augmentation method, we generate a similar macro plan which follows the content plan created by Chapter 6. For each concept with intra-event type content, we keep the paragraph plan the same as before (i.e., only containing the current event data of the entity). However, for each inter-event concept, we append the inter-event features with their values in the paragraph sequence which were classified as 'yes' in the content selection process described in Section 7.3.1. Finally, this input sequence of paragraph plans is fed into the neural surface realiser to produce the textual summary (part (2a) of Figure 7.4).

In this work, we use the surface realisation module of Puduppully and Lapata (2021) for our neural surface realisation as well. This is a LSTM based sequence-to-sequence model

TEAM	WIN	LOSS	PTS	FG	REB	AST	...
76ers	2	1	116	45	46	33	···
Magic	1	2	115	41	49	31	···

PLAYER	H/V	AST	REB	PTS	FG	TEAM	···
DJ Augustin	V	9	3	2	12	Magic	···
Evan Fourier	V	3	8	31	5	Magic	···
Isaiah Thomas	H	5	0	23	4	76ers	···
Kelly Olynyk	H	4	6	16	6	76ers	···
...							

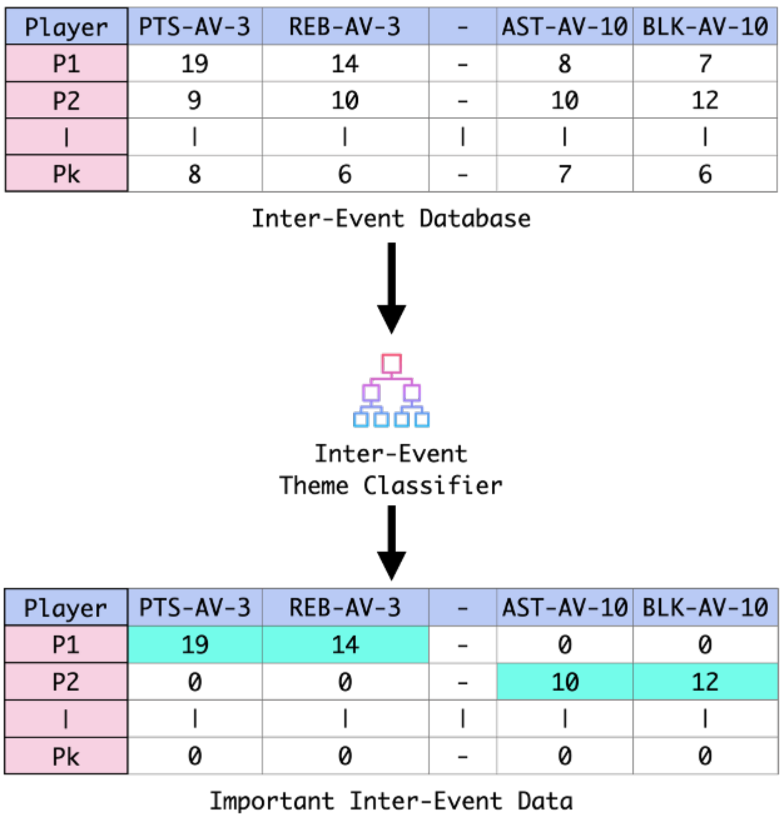
Figure 7.7: Event input data: a basketball game statistics table

where a bi-directional LSTM is used as encoder and another uni-directional LSTM as decoder. The model also uses attention mechanism to generate the most probable token at every decoding step[1].

Post-Editing

Even with the Input Augmentation method, it is difficult to control the learning of neural system and is possible to still have inaccuracies in the generated summary. Thus, to this end, we propose a method, called **Post-Editing**, to post-edit the neural system's summary using the CBR-D2T method of dynamic templating (proposed in Chapter 4).

In Post-Editing method, the neural network summary is broken into sentences and the sentences with inter-event content is identified. These sentences are then replaced with a new sentence generated using the dynamic templating CBR-D2T method (see part 2(b) of Figure 7.4). We build separate case-bases for player inter-event and team inter-event concepts. These case-bases contain the inter-event features on the problem-side and templates extracted from the inter-event sentences from training set (the process of building case-base and generating new sentence for a target problem is same as described in Chapter 4).



Player	PTS-AV-3	REB-AV-3	-	AST-AV-10	BLK-AV-10
P1	19	14	-	8	7
P2	9	10	-	10	12
|	|	|	|	|	|
Pk	8	6	-	7	6

Player	PTS-AV-3	REB-AV-3	-	AST-AV-10	BLK-AV-10
P1	19	14	-	0	0
P2	0	0	-	10	12
|	|	|	|	|	|
Pk	0	0	-	0	0

Figure 7.8: Inter event database and inter event theme classifier

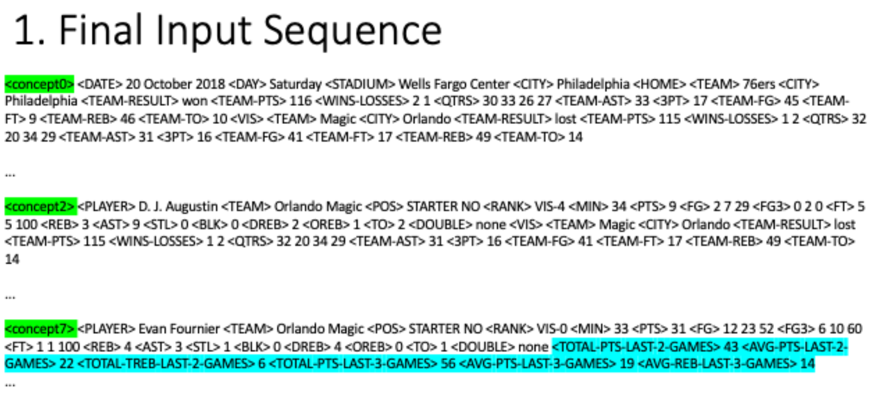
7.3.3 A Working Example

In this section we provide a brief working example of how our context-aware D2T system will work using an example. Let us take an example of an event's input data shown in Figure 7.7, for which a summary needs to be generated. Using the CBR-Plan method (Chapter 6), we produced a content plan, CP, for the event summary as follows:

$$CP = \{76ers\&Magic - B\&C, \cdots, DJAugustin - B, \cdots, EvanFourier - I, \cdots\} \tag{7.1}$$

For each event, the inter-event database (as shown at the top of Figure 7.8) will have a

[1]It is to note that any neural model capable of taking long text sequences (more than 1000 tokens) as inputs and generating long text sequences (on average 15-20 sentences) as output can be used for surface realisation phase. We choose Puduppully and Lapata (2021)'s module because its easy to configure and use in our case and also among the current state-of-the-art in complex D2T problems.



1. Final Input Sequence

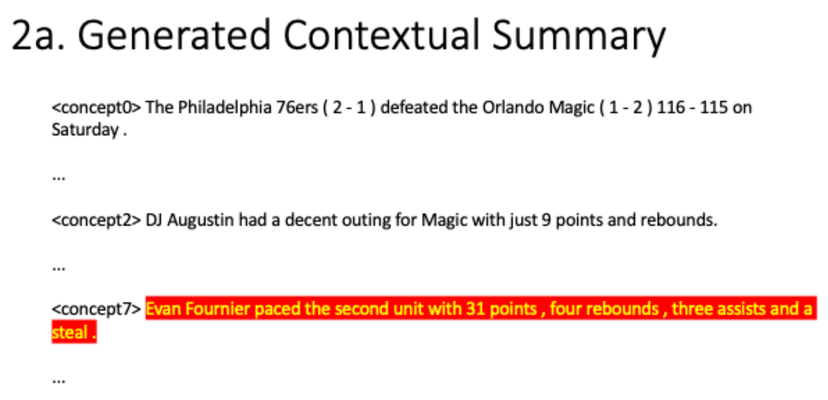
```
<concept0> <DATE> 20 October 2018 <DAY> Saturday <STADIUM> Wells Fargo Center <CITY> Philadelphia <HOME> <TEAM> 76ers <CITY>
Philadelphia <TEAM-RESULT> won <TEAM-PTS> 116 <WINS-LOSSES> 2 1 <QTRS> 30 33 26 27 <TEAM-AST> 33 <3PT> 17 <TEAM-FG> 45 <TEAM-
FT> 9 <TEAM-REB> 46 <TEAM-TO> 10 <VIS> <TEAM> Magic <CITY> Orlando <TEAM-RESULT> lost <TEAM-PTS> 115 <WINS-LOSSES> 1 2 <QTRS> 32
20 34 29 <TEAM-AST> 31 <3PT> 16 <TEAM-FG> 41 <TEAM-FT> 17 <TEAM-REB> 49 <TEAM-TO> 14

...

<concept2> <PLAYER> D. J. Augustin <TEAM> Orlando Magic <POS> STARTER NO <RANK> VIS-4 <MIN> 34 <PTS> 9 <FG> 2 7 29 <FG3> 0 2 0 <FT> 5
5 100 <REB> 3 <AST> 9 <STL> 0 <BLK> 0 <DREB> 2 <OREB> 1 <TO> 2 <DOUBLE> none <VIS> <TEAM> Magic <CITY> Orlando <TEAM-RESULT> lost
<TEAM-PTS> 115 <WINS-LOSSES> 1 2 <QTRS> 32 20 34 29 <TEAM-AST> 31 <3PT> 16 <TEAM-FG> 41 <TEAM-FT> 17 <TEAM-REB> 49 <TEAM-TO>
14

...

<concept7> <PLAYER> Evan Fournier <TEAM> Orlando Magic <POS> STARTER NO <RANK> VIS-0 <MIN> 33 <PTS> 31 <FG> 12 23 52 <FG3> 6 10 60
<FT> 1 1 100 <REB> 4 <AST> 3 <STL> 1 <BLK> 0 <DREB> 4 <OREB> 0 <TO> 1 <DOUBLE> none <TOTAL-PTS-LAST-2-GAMES> 43 <AVG-PTS-LAST-2-
GAMES> 22 <TOTAL-TREB-LAST-2-GAMES> 6 <TOTAL-PTS-LAST-3-GAMES> 56 <AVG-PTS-LAST-3-GAMES> 19 <AVG-REB-LAST-3-GAMES> 14
...
```

Figure 7.9: Final input sequence for neural surface realiser

tabular entry where each row represent an entity (player or team) from the event and each column is the inter-event feature. Thus each cell represents an inter-event feature value for an entity from the event. The inter-event theme classifier now classifies each cell for identifying if it can be discussed in the summary or not. The output after this step is important inter-event data, as shown at the bottom of Figure 7.8.

Now, based on the summary's content plan (Equation (7.1)), we produce a final input sequence by verbalising the input records of entities into textual sequence. Concepts with just the Intra-Event content types (B and C), the event input data is enough. For concepts with inter-event content (I) required (i.e., for Evan Fourier), the selected important inter-event data is used. An example of final input sequence is shown in Figure 7.9.

This final input sequence is fed into the neural surface realiser (the Input Augmentation method) to generate an initial contextual summary (see Figure 7.10). This summary contains one sentence per concept from the content plan. The summary is expected have good content accuracy for intra-event contents but may struggle with inter-event contents (as seen in the sentence highlighted red). Thus, the initial contextual summary is then post-edited with a CBR-D2T method to replace all the sentences corresponding to an inter-event concept (i.e., the concept in content plan marked with I content type) with a sentence produced by filling up a dynamically selected template for the entity in the


2a. Generated Contextual Summary

<concept0> The Philadelphia 76ers (2 - 1) defeated the Orlando Magic (1 - 2) 116 - 115 on Saturday .

...

<concept2> DJ Augustin had a decent outing for Magic with just 9 points and rebounds.

...

<concept7> Evan Fournier paced the second unit with 31 points , four rebounds , three assists and a steal .

...

Figure 7.10: Output from neural surface realiser after 'Input Augmentation' phase

concept. An example of a final post-edited contextual summary is shown in Figure 7.11.

7.4 Experiment Setup

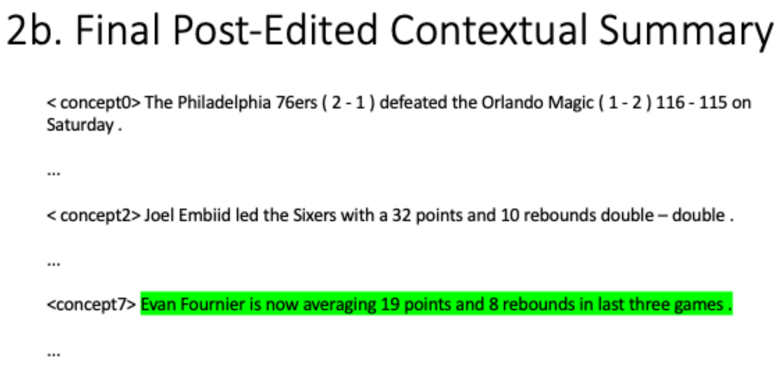
We now define the experiment setup used to evaluate our proposed method.

7.4.1 Dataset

The SportSett dataset (Thomson et al., 2020a) of NBA matches is used to evaluate the proposed content selection and surface realisation methodology [2]. Each match from the dataset contains a textual summary as the output and the associated match statistics, with the box- and line-scores, as the problem input. There is a temporal aspect involved here, as future matches should not be available to the learner. Hence the training set contains the earlier matches from the 2014, 2015 and 2016 seasons (total of 4775, some matches from the 2016 season have more than one summary) while the validation and test sets contain matches from the 2017 and 2018 seasons (1230 matches each) respectively.

The data for training the theme classifier for inter-event data selection, is build using the train and validation sets of SportSett. For each theme, if its inter-event features are included in a summary for an entity of the event, then its label is given as 1 otherwise

[2]we use the GEM version of the dataset from https://huggingface.co/datasets/GEM/sportsett_basketball



2b. Final Post-Edited Contextual Summary

< concept0> The Philadelphia 76ers (2 - 1) defeated the Orlando Magic (1 - 2) 116 - 115 on Saturday .

...

< concept2> Joel Embiid led the Sixers with a 32 points and 10 rebounds double – double .

...

<concept7> Evan Fournier is now averaging 19 points and 8 rebounds in last three games .

...

Figure 7.11: Final event summary after post-editing

Table 7.1: Dataset stats for building theme classifiers

	Player Average		Team Streak	
Label	Train Size	Test Size	Train Size	Test Size
Positive	3790	65	1707	488
Negative	73850	1470	5673	1972
Total	77640	1535	7380	2460

0. We use the samples from train set of SportSett for building the train set of theme classifier while the validation set is used for building the test set of the classifiers.

7.4.2 Inter-Event Data Selection Models

Multiple binary classifiers are used for building the theme classifiers for contextual inter-event data selection. These are: Logisitic Regression (**LR**), k-Nearest Neighbours (**kNN**), Support Vector Machines (**SVM**), Multi-Layer Perceptron (**MLP**), and Random Forest (**RF**) [3].

[3]these models are trained with https://scikit-learn.org/stable/

7.4.3 Surface Realisation Systems

For surface realisation, we select the current state-of-the-art macro-plan model (MP) Puduppully and Lapata (2021) as the benchmark to compare our methods. We use the same macro-plan model for input augmentation and the post-editing methodologies. This is a pipeline-based neural network model with two components: content planner, which combines planning and selection and is based on Vinyals et al. (2015) that takes the event input data and generates a content plan, also referred as macro-plan; surface realiser, is a sequence-to-sequence neural model with Bi-LSTM encoder and LSTM decoder that takes the macro-plan as input and generates the textual summary as output.

Thus, in our experiments we have three model's outputs to compare against each other:

- **Ent**: entity-based model from Puduppully et al. (2019b);
- **Hir**: hierarchical transformers model Rebuffel et al. (2020);
- **$\mathbf{MP_{base}}$**: base MP model of Puduppully and Lapata (2021) with exact same input and training configuration. We also use its original content planning method proposed by their authors;
- **$\mathbf{MP_{aug}}$**: this model is the surface realiser from $\mathrm{MP_{base}}$ that takes the augmented input as described in Section 7.3.2.
- **$\mathbf{MP_{pe}}$**: this is the post-editing model which utilises the CBR-D2T method (from Chapter 4) to post edit the output of $\mathrm{MP_{aug}}$ model.

7.4.4 Evaluation Metrics

For content selection, basic classification metrics such as: Precision, Recall, and F1 score are used. Since the dataset is imbalanced, we report the marco-average of these metrics and use them for model selection.

For surface realisation, the following automated metrics are used:

- **Extractive Evaluation**: Inspired by information extraction evaluations from Wiseman et al. (2017), we use a set of regular expressions to extract inter-event tuples from the system generations. These extracted tuples are then matched with the input data to evaluate the performance of text generation model;
- **Content Type Distribution and Concept Selection**: we also check the content

type distribution of summaries generated from these models using the method proposed in Chapter 5. The concept selection abilities of these models is also evaluated in accordance with the content selection process described in Chapter 6. Here we check the Precision, Recall, F1, and DLD Brill and Moore (2000b) scores of concepts selected in each summary against the human written gold summaries. A concept denotes the sentences structure identifying the type of entity and content described in the sentence.

7.5 Human Evaluation Setup for Accuracy

A human evaluation is also performed to measure the accuracy of inter-event content in system generated summaries. Our study follows a similar design as many from literature where human evaluators are shown some sentences from an event summary and asked to count the supporting and contradicting numerical facts mentioned in them Puduppully et al. (2019a); Puduppully and Lapata (2021); Rebuffel et al. (2020); Wiseman et al. (2017). The main difference in our study is that instead of evaluating whole summary, we only evaluate the inter-event content from the summaries. We also measure the accuracy of systems according to different entity-types (in sports domain - player and team) and investigate if the accuracy varies across them.

Goal: The goal of this study is to measure the inter-event content's accuracy of the text generation systems by counting the number of supporting and contradicting inter-event numerical claims generated by them.

Systems: outputs from three text generation systems are compared: MP_{base}, MP_{aug}, and MP_{pe}. The reason for choosing MP_{aug} and MP_{pe} is that these are the two systems built using our proposed method. While MP_{base} is selected as the baseline for human evaluation because it is most similar to MP_{aug} & MP_{pe} as compared to the other two neural methods from the literature, Hir and Ent. It also performs similar to others on automated metrics (Tables 7.3 and 7.4).

Outputs: The sentences evaluated by human subjects are chosen randomly in a two-staged process. First, the content-type classifier for SportSett dataset (Section 5.2.2) is used to identify all the sentences belonging to inter-event category from all the system generated summaries. Then a sample of sentences is chosen randomly from the pool of inter-event sentences for human evaluation. We select 50 sentences for MP_{pe} system, 40 for $\mathrm{MP}_{\mathrm{aug}}$, and 30 for $\mathrm{MP}_{\mathrm{base}}$.

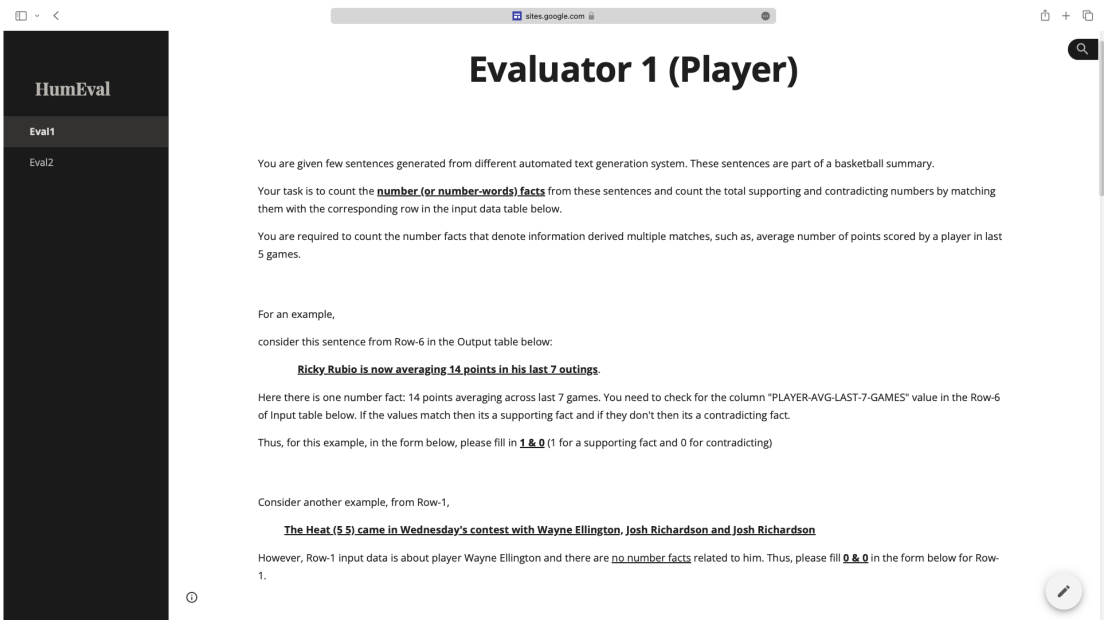
Number of human subjects and annotation process: We have three human subjects for our study. Each sentence in our evaluation pool is annotated by at least one annotator, while 25% of sentences (30), are annotated by two subjects to calculate the inter-annotator agreement.

Selecting Participants The participants for the study were selected based on the following three criteria:

- **Language Ability**: we wanted the participants to be fluent in English language because the event summaries are in English language.
- **Familiarity with AI Systems**: we wanted participants to have good familiarity of AI systems, especially generative AI, because this will allow them to free any pre-conceived bias of 'textual form' of system generated texts (Clark et al., 2021).
- **Domain Understanding**: even though the participants weren't domain experts, we wanted them to have some level of understanding of the basketball domain. They should be familiar with some common terminology used in basketball such as 3-pointers, or double-double.

Questions Asked to Participants A website was built to collect the human evaluation data. The website had three main features: first, the home page - with the necessary background information about the study with an example task; second, the data page - showing the input to the system and its corresponding output, which they were asked to evaluate; and third, the submission form - to submit their answers for the asked evaluation questions. The website is shown in Figures 7.12 to 7.14.

The participants were asked to count total supporting and contradicting numbers generated in the sentence. Each sentence, that needs to be evaluated, has corresponding input data that is used by the text generation system to generate that sentence. The input data is the inter-event attributes about the entity described in the sentence. Two different types of sentences are presented for evaluation: one, sentences describing player entities; and another, sentences describing team entities.



Evaluator 1 (Player)

You are given few sentences generated from different automated text generation system. These sentences are part of a basketball summary.

Your task is to count the **number (or number-words) facts** from these sentences and count the total supporting and contradicting numbers by matching them with the corresponding row in the input data table below.

You are required to count the number facts that denote information derived multiple matches, such as, average number of points scored by a player in last 5 games.

For an example,

consider this sentence from Row-6 in the Output table below:

Ricky Rubio is now averaging 14 points in his last 7 outings.

Here there is one number fact: 14 points averaging across last 7 games. You need to check for the column "PLAYER-AVG-LAST-7-GAMES" value in the Row-6 of Input table below. If the values match then its a supporting fact and if they don't then its a contradicting fact.

Thus, for this example, in the form below, please fill in **1 & 0** (1 for a supporting fact and 0 for contradicting)

Consider another example, from Row-1,

The Heat (5 5) came in Wednesday's contest with Wayne Ellington, Josh Richardson and Josh Richardson

However, Row-1 input data is about player Wayne Ellington and there are no number facts related to him. Thus, please fill **0 & 0** in the form below for Row-1.

Figure 7.12: Home page of human evaluation website

7.6 Results

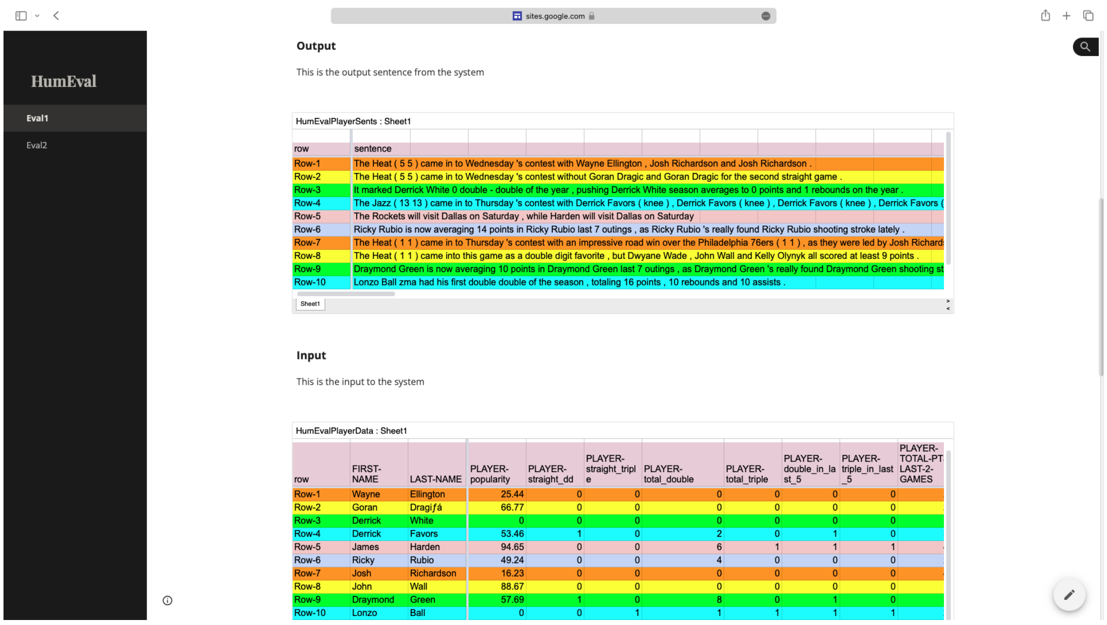
The results are discussed in two steps: first, we briefly discuss the results of content selection, where we identify the best learning algorithm for building a theme classifier; and second, we discuss the results of surface realisation experiments, where we compare the effectiveness of the different methods proposed for adding inter-event content to the summaries.

The headline results from this chapter are as follows:

- augmenting event's input data with important information from past related events improves the content accuracy of neural system generated summaries
- further post-editing of neural system generated summaries using CBR-D2T dynamic template method improves the distribution of different content types in summaries while also improving the content accuracy significantly

7.6.1 Inter-Event Data Selection

The performance of five theme classifiers built using different learning algorithms is shown in Table 7.2. We report the macro averaged scores of precision, recall and F1 metrics



Output

This is the output sentence from the system

HumEvalPlayerSents : Sheet1

row	sentence
Row-1	The Heat (5 5) came in to Wednesday 's contest with Wayne Ellington , Josh Richardson and Josh Richardson .
Row-2	The Heat (5 5) came in to Wednesday 's contest without Goran Dragic and Goran Dragic for the second straight game .
Row-3	It marked Derrick White 0 double - double of the year , pushing Derrick White season averages to 0 points and 1 rebounds on the year .
Row-4	The Jazz (13 13) came in to Thursday 's contest with Derrick Favors (knee) , Derrick Favors (knee) , Derrick Favors (knee) , Derrick Favors (
Row-5	The Rockets will visit Dallas on Saturday , while Harden will visit Dallas on Saturday
Row-6	Ricky Rubio is now averaging 14 points in Ricky Rubio last 7 outings , as Ricky Rubio 's really found Ricky Rubio shooting stroke lately .
Row-7	The Heat (1 1) came in to Thursday 's contest with an impressive road win over the Philadelphia 76ers (1 1) , as they were led by Josh Richard
Row-8	The Heat (1 1) came into this game as a double digit favorite , but Dwyane Wade , John Wall and Kelly Olynyk all scored at least 9 points .
Row-9	Draymond Green is now averaging 10 points in Draymond Green last 7 outings , as Draymond Green 's really found Draymond Green shooting st
Row-10	Lonzo Ball zma had his first double double of the season , totaling 16 points , 10 rebounds and 10 assists .

Sheet1

Input

This is the input to the system

HumEvalPlayerData : Sheet1

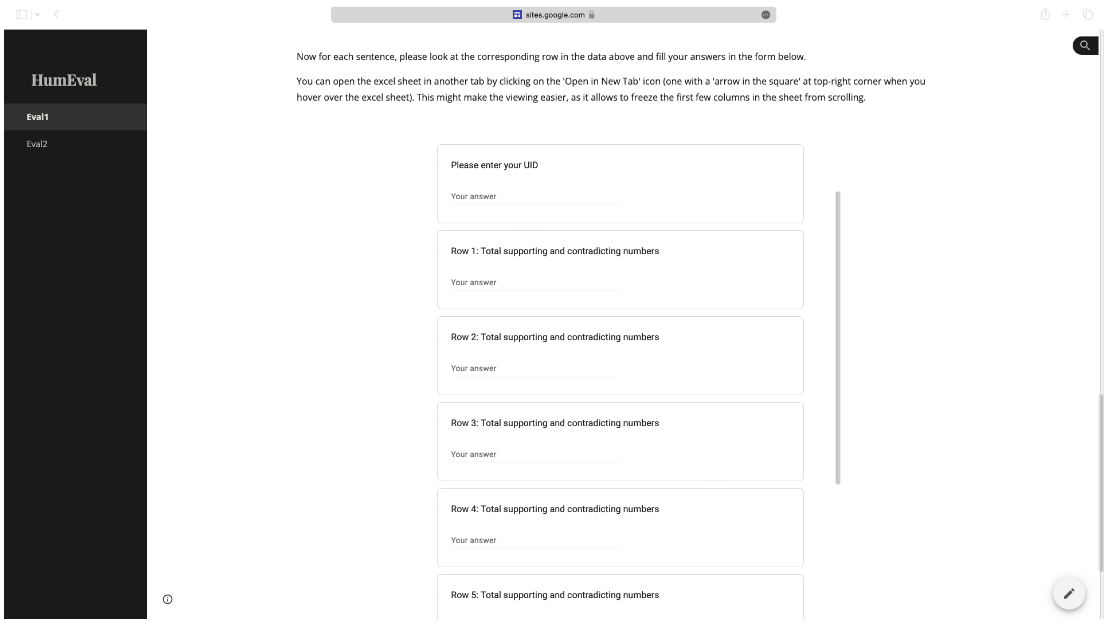
row	FIRST-NAME	LAST-NAME	PLAYER-popularity	PLAYER-straight_dd	PLAYER-straight_triple	PLAYER-total_double	PLAYER-total_triple	PLAYER-double_in_last_5	PLAYER-triple_in_last_5	PLAYER-TOTAL-PT LAST-2-GAMES
Row-1	Wayne	Ellington	25.44	0	0	0	0	0	0	
Row-2	Goran	Dragiʄá	66.77	0	0	0	0	0	0	
Row-3	Derrick	White	0	0	0	0	0	0	0	
Row-4	Derrick	Favors	53.46	1	0	2	0	1	0	
Row-5	James	Harden	94.65	0	0	6	1	1	1	
Row-6	Ricky	Rubio	49.24	0	0	4	0	0	0	
Row-7	Josh	Richardson	16.23	0	0	0	0	0	0	
Row-8	John	Wall	88.67	0	0	0	0	0	0	
Row-9	Draymond	Green	57.69	1	0	8	0	1	0	
Row-10	Lonzo	Ball	0	0	1	1	1	1	1	

Figure 7.13: Data page showing input data and output sentences from the D2T systems

Table 7.2: Performance of Theme Classifiers for inter-event data selection

	Player Average				Team Streak			
Model	Accuracy	F1	Precision	Recall	Accuracy	F1	Precision	Recall
LR	**95.77**	48.92	47.85	50	80.69	54.19	69.08	54.72
kNN	94.79	53.20	56.68	52.43	75.41	55.07	56.72	54.74
SVM	**95.77**	48.92	47.88	50	80.20	44.71	90.1	50.1
MLP	94.20	**60.49**	**61.74**	**59.48**	**80.85**	51.72	**72.61**	53.44
RF	94.85	57.09	61.44	55.41	79.07	**58.06**	63.42	**57.18**

along-with accuracy of the classifiers. It can be observed that despite a higher accuracy, the other metrics have lower scores. This is expected as the dataset for these theme classifiers is highly imbalanced towards negative class. Still we can see learners such as MLP and RF achieve around 60% of F1 scores. Thus MLP is selected as the Player Average Theme classifier while RF as the Team Streak Theme classifier.



Now for each sentence, please look at the corresponding row in the data above and fill your answers in the form below.

You can open the excel sheet in another tab by clicking on the 'Open in New Tab' icon (one with a 'arrow in the square' at top-right corner when you hover over the excel sheet). This might make the viewing easier, as it allows to freeze the first few columns in the sheet from scrolling.

Please enter your UID

Your answer

Row 1: Total supporting and contradicting numbers

Your answer

Row 2: Total supporting and contradicting numbers

Your answer

Row 3: Total supporting and contradicting numbers

Your answer

Row 4: Total supporting and contradicting numbers

Your answer

Row 5: Total supporting and contradicting numbers

Figure 7.14: The form to submit the data for counted supporting and contradicting facts

7.6.2 Surface Realisation

Extractive Evaluations

We start with discussing the results from regular expression evaluation of the surface realisation outputs. The evaluation consists of few regular expressions per theme that count the presence of inter-event content in the generated summaries. These expressions extract a tuple of information in the form of ($entity\_name$, $value$, $inter\_event\_feature\_name$) and match these with the input to count the number of supporting and contradicting claims. For example, for the given sentence - "*Kevin Durant is averaging 14 points over his last 5 outings*"; the extracted tuple would be - ($Kevin\_Durant$, 14, $AVG\_PTS\_LAST\_5\_GAMES$). This is then matched to the input data to identify if this is a supporting or contradicting claim. The results from this experiment for both the themes are shown in Table 7.3. The column name 'Correct.' shows the number of supporting claims, 'Total.' shows the number of total claims made by the system in test set, while '%Correct' is the percentage of correct/supporting claims out of total extracted claims.

The results clearly demonstrate the benefit of including inter-event content directly in the input data in order to include better inter-event content in the summaries. We see that MP$_{\text{base}}$ only generates 20 inter-event contents for player average theme with only 4

Table 7.3: Extractive Evaluation results with BLEU score (↑, higher is better)

	Player Average			Team Streak			
Systems	%	Total	Correct	%	Total	Correct	BLEU
Ent (Puduppully et al., 2019b)	22.28	193	43	1.28	78	1	15.93
Hir (Rebuffel et al., 2020)	17.33	150	26	6.25	48	3	19.68
MP_{base} (Puduppully and Lapata, 2021)	20	20	4	0	0	0	22.23
MP_{aug}	42.65	68	29	0	1	0	18.66
MP_{pe}	63.41	2859	1813	38.04	92	35	15.20

Table 7.4: Extractive evaluation results using Information Extraction metrics

Systems	RG	CS-Precision	CS-Recall	CO	BLEU
Ent (Puduppully et al., 2019b)	76.4	46.07	54.0	24.82	15.93
Hir (Rebuffel et al., 2020)	78.39	48.28	45.11	19.02	19.68
MP_{base} (Puduppully and Lapata, 2021)	86.48	53.95	33.09	14.84	22.23
MP_{aug}	83.07	52.09	27.01	16.98	18.66
MP_{pe}	74.66	47.41	25.78	11.79	15.20

of those being correct. It also doesn't generate any inter-event content for team streak theme at all. Next, we see a good performance gain with MP_{aug} when the input of model is augmented with the selected inter-event content. This model generates 68 player average theme claims out of which 42% are correct. However, it still doesn't generate any supporting team streak theme claims. This suggests that it is difficult to make neural model generate a specific type of content if there aren't sufficient examples in the training set. Finally, we observe the MP_{pe} model's performance and immediately notice substantial improvements across both themes. This model is able to generate around 2.9k player average theme claims, out of which 63% are also correct. For team streak theme as well, the model is generating 90+ claims with 38% of them being correct.

Now we show the results of extractive evaluation metrics calculated using information extraction (IE) method described in Wiseman et al. (2017). This method also extracts claims in the form of $(entity\_name, value, feature\_name)$ tuple, however it can only

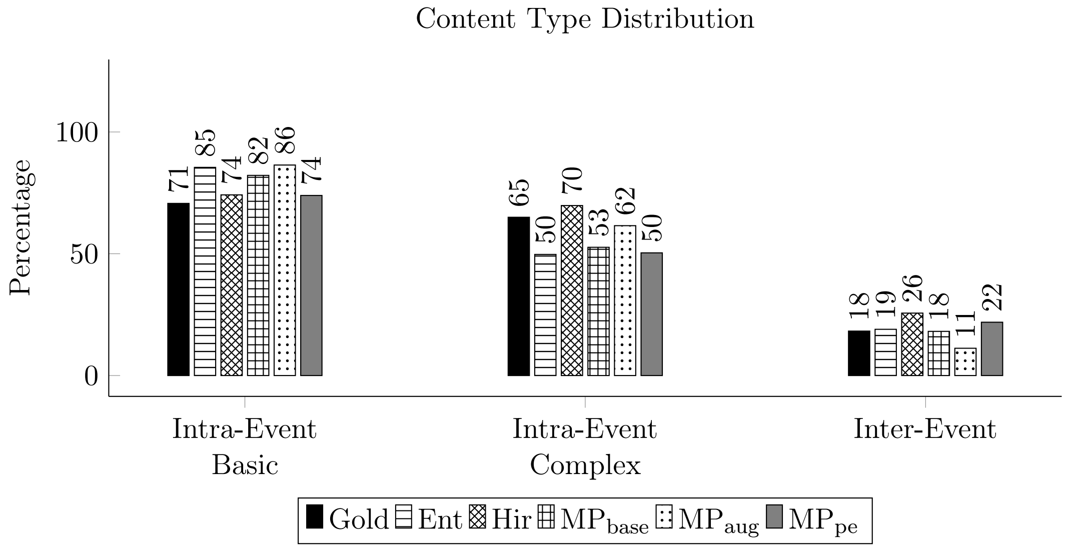


Figure 7.15: Content-Type distribution of summaries from different systems

extract intra-event basic type of claims. There are three types of metrics: Relation Generation (RG), which compares the extracted tuples from system generations with input data to measure the accuracy; Content Selection (CS), which is different to the content selection previously described as it compares the intra-event basic tuples extracted from system generations to tuples extracted from human written gold standard summaries; Content Ordering (CO), which checks the edit distance between tuples extracted from system generations and the gold standard summaries. The results are shown in Table 7.4.

We can see that MP_{base} performs best on RG, CS-Precision, & BLUE metrics; suggesting that the addition of inter-event content in the input might leads to small performance decrease. But this is not true because the IE method used for these metrics extracted numbers from a sentence and if they do not match the value of any intra-event basic type feature then it is considered incorrect. For example, in the sentence - "*Kevin Durant is averaging 14 points over his last 5 outings*", the extracted tuple - (*Kevin_Durant*, 14, *AVG_PTS_LAST_5_GAMES*) will be considered incorrect by this IE method because it did not match any intra-event basic type feature, i.e., information about current event. However, when we look at the BLEU score of the three systems, we do not see huge drops suggesting that the two proposed surface realisation models are able to generate summaries with similar fluency as the benchmark model.

Table 7.5: Concept selection ability of different systems

CBR-Plan$_{euc}$	CS F1	CS Precision	CS Recall	CO DLD	Length Avg
Gold	-	-	-	-	12.76
Ent (Puduppully et al., 2019b)	24.86	24.05	25.73	10.94	13.66
Hir (Rebuffel et al., 2020)	33.38	34.45	32.37	11.27	11.99
MP$_{base}$ (Puduppully and Lapata, 2021)	28.63	33.13	25.2	8.8	9.71
MP$_{aug}$	46.18	33.82	39.04	13.03	9.35
MP$_{pe}$	40.71	29.88	34.47	9.49	9.37

Content Type Distribution and Concept Selection

Next we investigate the content type distribution of summaries generated from these different systems. Figure 7.15 shows the percentage of sentences with different content types in summaries generated from the three systems and the human written gold summaries[4]. MP$_{pe}$ produces 22% inter-event sentences, slightly more than Gold. However, MP$_{base}$ is also able to generate equal amount of inter-event content as Gold but most of which is incorrect as identified in extractive evaluation results. We see that MP$_{aug}$ has the lowest amount of inter-event sentences despite having more content relating to inter-event themes as described in Table 7.3. This suggests that the MP$_{base}$ also generates some sentences with inter-event content that are not identified by regex evaluations. A quick look to the generated summaries shows that sentences such as: ‘this was player X’s first game after missing Y games due to injury’. Even though classified as inter-event, they do not contain any information that can be easily verified via automated metrics or even quick human evaluation.

We also investigate the content planning ability of the three systems using their generated summaries using the evaluation method described in Chapter 6. This method extracts a concept list (as shown in Figure 7.3) from a system generated summary and then compares it with the concept list from gold summaries. In Table 7.5, we show the F1, Precision, and Recall scores to compare the concepts selected in system generations, while DLD (edit-distance) scores are used to compare the ordering of these concept lists. We can see that with both the MP$_{aug}$ and MP$_{pe}$ systems are able to improve all four scores when compared to MP$_{base}$. This suggests that adding inter-event content to the input data helps in improving the organisation of a generated summary and bring it closer to

[4]It should be noted that a sentence can have multiple types of content, thus adding the percentage of different content types will not be equal to 100.

Systems	Team Streak #**Support** (↑)	 #**Contra** (↓)	Player Average #**Support** (↑)	 #**Contra** (↓)
$\textbf{MP}_{\textbf{base}}$ (Puduppully and Lapata, 2021)	0	0.667	0	3
$\textbf{MP}_{\textbf{aug}}$	0.067	0.93	0.428	1.5
$\textbf{MP}_{\textbf{pe}}$	0.7	0.3	1	0.1

Table 7.6: Human evaluation results divided into two inter-across themes

the human written ones.

7.6.3 Human Evaluation of Generated Summaries

The results from human evaluation are shown in Table 7.6. The annotated sentences by human subjects were divided into the two inter-event contextual themes: Team Streak, and Player Average[5]. First, we observe that MP_{base} is the worst performing system, with no supporting facts at all and 3 incorrect player average facts per sentence and 0.6 incorrect team streak fact per sentence. This is expected as MP_{base} doesn't have access to the contextual inter-event data during run-time. The incorrect facts are generated because of the system hallucinating in some common player average sentences such as 'Kevin Durant has been averaging 17 points, 12 rebounds and 6 steals in last 3 games'. In most cases, all the three claims about the player in this sentence are incorrect because they are not based on input data but rather system hallucinating those based on previously generated tokens.

Next, we observe that the MP_{aug} is able to achieve better accuracy by generating a Player Average supporting fact every two sentence. MP_{pe} is also able to generate 1 Player Average supporting fact each sentence and 2 Team Streak supporting sentence every 3 sentence. These results demonstrate that providing contextual information from past events to neural systems improve the quality of their generations. While further post-editing the part of summaries with templates provides even better rate of supporting facts generation.

We also observe that both, MP_{aug} and MP_{pe}, struggle more with Team Streak theme facts compared to Player Average theme. This behaviour can be explained in two parts:

[5]The inter-annotator agreement measured by Cohen's Kappa is 0.655 and Krippendorff's Alpha is 0.657.

first, for $\mathrm{MP}_{\mathrm{aug}}$, although the system now has access to inter-event data during run-time - the amount of Team Streak theme sentences are much less in the training set summaries compared to Player Average theme sentences. This makes learning the properties for Team Streak sentences difficult and still leaves some room for hallucinations.

In case of $\mathrm{MP}_{\mathrm{pe}}$, the automatically extracted templates for Team Streak convey more complex information than Player Average. For example, a Team Streak template can be: *"TEAM_NAME has won X out of last Y games and is on a Z games winning streak"*. These templates are able to convey correct information about team's streak (slot Z) but may have incorrect values filled up for slots X & Y. This can be improved either by providing correct data for X & Y or by commissioning even better templates with only relevant slots for available data.

7.7 Conclusion

Current D2T methods, despite achieving good performance, struggle to generate inter-event content. This type of content is challenging for most systems because they need to process large amount of data from previous events in order to generate inter-event content. Contextual Inter-Event Data Selection method selects a subset of important inter-event data from past events that can be added to the textual summary. Two methods for surface realisations (Input Augmentation, and Post-Editing) is proposed to utilise the selected content in conjunction with a content plan for generating the final textual summary. The input augmentation method enriches the input of neural D2T systems in order to improve their output quality. The post-editing methods takes the output of neural D2T systems and replaces their inter-event sentences with sentences generated using a CBR-D2T system. Through extensive experiments, performed using a range of automated and human evaluations, we demonstrate the benefit of Post-Editing method in producing accurate inter-event content.

Chapter 8

Conclusion and Future Work

This thesis investigated approaches to include contextual information in D2T systems when generating summaries of temporal related events. We identified that current methods lack contextual information when generating event summaries. As a result the generated summaries are not similar to human authored ones.

In this thesis, we investigated two research questions:

RQ1 How to build a D2T system that can effectively mitigate the accuracy and diversity trade-off in its generated summaries?

RQ2 How to build a D2T system that can effectively process the contextual information about the event and generate summaries with contextually aware insights?

The aim was to develop a method that can balance the accuracy and diversity trade-off of current D2T approaches and also capture different types of contextual information to include in the event summaries with interesting insights. To answer these research questions, we identified six objectives. In the next subsection, we revisit those objectives and discuss the approach taken to achieve them. After that, we discuss the some insights gathered during working on the project and followed by the limitations of our proposed method. Finally, we conclude with a discussion of possible direction for future work in this field to address some of the limitations.

8.1 Objectives Revisited

The overall contribution of this thesis is a pipeline-based context-aware modular system for event-based D2T problems. The pipeline consists of: a data selection module, to select important derived records from past related events; a content planning module, to plan and organise the content of a summary by utilising environmental context, capturing the real-world dependencies extracted from the data itself; and finally a surface realisation module, which captures the temporal context from of the time-series nature of the events and generates the final textual summary from the data.

This section revisits all the objectives identified in Chapter 1 and discusses how these objectives were met.

8.1.1 O1: Review the Natural Language Generation literature specially towards D2T

The success of deep learning based neural methods has given rise to a new AI sub-field, Generative AI (GenAI). GenAI promises disruptive improvements in many NLP tasks that were previously very hard to solve. NLP promises to bridge the gap between humans and machines by providing a natural language interface for humans to communicate with machines. D2T, being one of its subfields, aims to summarise structured (mostly non-linguistic data) into textual summaries. The primary goal of such systems is to extract valuable insights from vast amounts of data and present them to humans in a textual summary, a form which humans respond to better.

The emphasis of this objective was to lay down the foundation of the field of NLP and discuss its different types of tasks. This allows us to understand that where D2T fits within NLP landscape. We then move on to discussing D2T methods and challenges in-detail. This objective was achieved with delivery of the literature review in Chapter 2. The key findings were: first, neural system despite generating human-like fluent and diverse summaries struggle with hallucinations and generate inaccurate content. Second, in some D2T domains, events are not independent and the contents of their summaries are derived from past related events as well. Current systems are unable to identify related content from past events and thus struggle to generate summaries with interesting insights as present in human authored summaries.

8.1.2 O2: Build a method to identify the mix of content types in the human-authored D2T summaries

This objective was achieved with the creation of a new typology to identify different types of content present in event summaries. The typology, called Content Type Typology, is presented in Chapter 5 and can be used to generate a profile of a D2T dataset based on its event summaries. Human written D2T summaries often contain information of different complexity level. Some information is derived from multiple records of an event while some may be derived from multiple records of multiple events. Text generation systems face different level of complexities in order to generate information of different types. Thus, it becomes important to understand the content type distribution of human written summaries of the domain. This objective proposes a three-class typology of content types in event-based D2T summaries (Chapter 5). This typology identifies three types of content, with increasing level of complexity to produce them: first, intra-event basic, which is verbatim of a record from an event; second, intra-event complex, which is derived from multiple records of an event; and finally inter-event, which is the information derived from multiple records of multiple past-related events.

A survey of content type distribution is conducted in both human written and system generated D2T summaries from different domains. A content type classifier is build for different domains which is used to generate a profile of each dataset. The profile is the distribution of different content types in the human written summaries. Same process is used to generate the profile of text generation systems based on their generated summaries. An extensive human evaluation study is carried out to identify the accuracy errors made by text generation systems in different content type category. The study found out that the current state-of-the-art neural systems have very different distribution of content as compared to their human-authored counter parts. The system generated summaries contain a lot less inter-event and intra-event complex type content as compared to intra-event basic. The accuracy errors are also very high in inter-event content as compared to intra-event content. Within intra-event as well, the complex type content is more inaccurate than basic type content.

8.1.3 O3: Develop a data-driven methodology for D2T to mitigate the accuracy vs diversity trade-off in neural and rule based D2T systems

We have shown that rule-based template systems are able to produce high quality but generate monotonous summaries. On the other hand, neural based systems are able to generate very diverse and fluent summaries but often hallucinate with inaccurate generations. Our novel method, CBR-D2T, presented in Chapter 4, overcomes many of these issues. CBR-D2T uses a case-based approach to dynamically select templates and organise them to form a final summary. D2T summaries can be easily broken down into multiple components according to the entities and information they describe. CBR-D2T selects relevant templates for these components and populates them with relevant data. A collective content selection method is used to identify important entities that should be described in the summary and then use a CBR method to dynamically choose templates according to their representation to describe them.

8.1.4 O4: Develop a Context-Aware Content Planning method to use environmental context in building the content plan for an event summary

This objective is achieved by developing a novel method called CBR-Plan, which is a context-aware content planning method presented in Chapter 6. CBR-Plan produces content plans that are designed to be more similar to those produced by humans. It utilises contextual information from wider environment in which events take place. Several environmental contextual information is identified through domain knowledge and represented using specific features. These features, along-with general event representation is used by CBR-Plan to generate contextually-aware content plans for event summaries. The evaluation demonstrates that CBR-Plan is able to generate content plans with a similar structure to those generated by humans than current state-of-art neural methods.

8.1.5 O5: Develop a Context-Aware Surface Realization methodology to use temporal context for generating final event summary

Surface realisation is the D2T process that produces the final summary from the content plan. Current state-of-the-art D2T systems struggle to include temporal context and thus fail to generate inter-event content in their summaries. The distribution of inter-event content in system generated summaries is usually very low, with high accuracy errors. This makes such system generated summaries less interesting and unfaithful to read.

This objective is achieved by building a CBR-assisted context-aware surface realisation method, $\mathrm{MP}_{\mathrm{pe}}$, and is presented in Chapter 7. $\mathrm{MP}_{\mathrm{pe}}$ uses a neural network to write the initial summary according to the expected style and then a case-based dynamic template system to post-edit the neural network's summary, in order to improve its inter-event content accuracy. The input of neural network is also augmented with inter-event data selected from past-related events using a contextual inter-event data selection classifier.

8.1.6 O6: Evaluate the proposed methods in a complex domain using datasets similar to real-world problems

This objective is delivered with comprehensive experiments described and discussed in Chapters 4 to 7. A real-world sports domain dataset, SportSett (Thomson et al., 2020a), is used for experimentation. SportSett contains time-stamped events of basketball games played in the NBA league in USA[1]. The dataset contains the teams and players performances from a game as the input while a textual summary (on average 12-15 sentences long) as output.

The first evaluation is performed in Chapter 4 to evaluate the case-based dynamic template method, CBR-D2T. We used standard automated metrics from literature to evaluate the summaries generated by CBR-D2T and compared them against summaries generated from template- and neural- based systems. We found that even with one-third training data, the CBR-D2T method is able to achieve much higher accuracy than neural system whilst maintaining better fluency and diversity than template system. We also perform an ablation study to understand the benefit of several key steps used in the generation process such as: using a language model to rank the retrieved templates, or using another neural text fusion model to combine the filled templates into final summary.

Next, we evaluate the context aware content planning methodology, CBR-Plan, presented in Chapter 6. This evaluation also uses the SportSett dataset from sports domain and the results demonstrate the benefit of CBR-Plan against a template baseline and several neural benchmarks. An ablation study also examines the impact of different contextual features on content planning and finds the combination of all to work best. The content

[1]http://www.nba.com/

plan generated by CBR-Plan contains concepts more similar to that in human-written summaries as compared to other benchmarks (2+ F1 score in content selection and 2+ DLD score in content ordering). The content type distribution of these concepts is also more closer to human-written summaries.

Finally, in Chapter 7, we perform the evaluation of final context-aware summary generated by the context aware surface realiser. The proposed method in the chapter uses a hybrid neural and dynamic template method to generate a contextually aware summary with inter-event content. The proposed method also uses an inter-event data selection module which carefully selects contextual inter-event data from past related events. This module uses a binary classifier to decide which of the possible records (derived from past events data) can be used in the summary. We evaluate different possible machine learning classifiers based on their accuracy and macro-f1 score and choose the best one.

We use the surface realiser module from a state-of-the-art neural system, Macro-Plan (Puduppully and Lapata, 2021), to implement our idea. The empirical evaluation compares three methods: first, the base Macro-Plan (MP_{bas}) method as described in Puduppully and Lapata (2021); second, a setting where input data for each event is augmented with selected inter-event contextual data (MP_{aug}); and third, a method where output of MP_{aug} is post-edited using case-based dynamic template method to improve the accuracy of inter-event content (MP_{pe}). This evaluation is based on both automated metrics and human judgement which aim to study the inter-event accuracy of the three systems.

The results demonstrate that the post-editing method, MP_{pe}, produces six times less contradicting facts while six times more supporting facts as compared to input augmentation method, MP_{aug}. The input augmentation method is also able to produce three times less contradicting facts than the base model, MP_{base}. Other results also demonstrates that the improvement in accuracy isn't traded with a major drop in fluency. The content type distribution also improves with post-editing, as it allows the summary to include more inter-event content than that of generated from input augmentation method.

8.2 Limitations and Future Works

In this section, we discuss some of the limitations of the work presented in this thesis and outline some possible areas for consideration in future research directions.

Commissioning Templates The core idea presented in this work is post-editing an event summary generated by a neural system with some dynamically selected templates to improve the overall accuracy of the event summary. However, the main challenge with using templates is their procurement. In order to produce a good summary with our proposed method, the quality and quantity of templates is very important. One option is to produce some templates manually with the help of a domain expert. However, this can be very expensive and also be difficult to offer content/lexical diversity. A work-around is to automatically extract templates from previously solved examples. But this approach presents its own challenge where the extraction process is not 100% error-proof, as discussed in Section 4.4.3. In current work, we used a simple entity-value matching method to extract templates from sentences.

In future, an Information Extraction based approach can be explored to extract the templates. This will reduce the margin of error for identifying the incorrect slots in the sentences, hence improving the overall accuracy of text produced by these templates. Another option could be a collaboration between experts and AI systems. A collaborative setup where experts create a few templates to start with and then LLM-based generative models are used to produce different variations of those templates. LLMs are known to be good at creative tasks, thus their creative ability could be useful in augmenting the bank of expert-generated templates. However, these variations might only be able to offer lexical diversity and not any content-based diversity. For example, the variation of a template that describes the 'average points scored by player X in Y games' can say the same thing with different lexical variation. But it would be more interesting if the variations are also able to produce something such as 'total rebounds made by player X in Y games'. Note that changing 'average points' to 'total rebounds made' allows a different type of attribute to describe, allowing the system to be more diverse on the basis of content.

Domain Dependence One limitation of the work is its domain dependence in selecting contextual information. Extracting both temporal and environmental context requires a pre-analysis of problem domain. Some domain knowledge is required to identify environmental contextual features and ways to collect them. Similarly, identifying temporal contextual features will also require deep domain understanding. One way of identifying such features could be to asking domain experts. However, this can be expensive and thus in this work we have explored data analysis based approaches.

We argue that this type of corpus analysis is required for any domain regardless of type of text generation system being used. However, an understanding the different requirements and expectations of end-users from the generated summary is also necessary. Domain analysis such as these will also help in expanding the dimensions of end evaluation of the system.

Extended Evaluation In this thesis, we evaluated our proposed methods on mostly sports domain datasets. In future extensions, the method can be evaluated on multiple datasets from different domains. Problem domains such as finance and weather reporting will have similar properties as described in this work. The textual summaries in these domains are also influenced by contextual information, both temporal and environmental. Recent advancements in IoT and their use in health sector applications, such as care homes, provides another opportunity to apply this work in a new domain that require context awareness. Such problems monitor time-series data collect via IoT sensors, and summarise them in textual reports.

A complex error-analysis of the proposed method can also be explored in future extensions. In this thesis, we only evaluated the accuracy of inter-event sentences by counting the supporting and contradicting inter-event facts generated per sentence. We tried to understand errors in different types of entities, i.e., players and teams. However, a better understanding of the system can be achieved by analysing the generated texts in regard to richer error taxonomies. Thomson et al. (2023) present a six-class taxonomy of errors in complex D2T problems which can be used to analyse the proposed methods further.

Personalisation from Writer's Perspective In different chapters, we touched upon using the writer's perspective to either understand content type distribution or prepare content plans. Further exploration in this direction can be an interesting area of research. Summaries could be personalised from reader's perspective as well as writer's perspective. In certain domains such as sports, a group of readers can prefer a certain writer's summary over others. This is common amongst sports league fans around the world. The summary could be toned to cater for particular fan groups by talking more about the players from their team rather than the others.

Bibliography

Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59.

Adeyanju, I., Wiratunga, N., and Lothian, R. (2010). Learning to author text with textual cbr. In *European Conf. on Artificial Intelligence*.

Allen, N. D., Templon, J. R., McNally, P. S., Birnbaum, L., and Hammond, K. (2010). Statsmonkey: A data-driven sports narrative writer. In *2010 AAAI Fall Symposium Series*.

Angeli, G., Liang, P., and Klein, D. (2010). A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512, Cambridge, MA. Association for Computational Linguistics.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Barzilay, R. and Lapata, M. (2005). Collective content selection for concept-to-text generation. In *Proc. of Human Language Technology Conf. and Conf. on Empirical Methods in Natural Language Processing*, pages 331–338. Association for Computational Linguistics.

Beck, D., Haffari, G., and Cohn, T. (2018). Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283.

Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Brill, E. and Moore, R. C. (2000a). An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, Hong Kong. Association for Computational Linguistics.

Brill, E. and Moore, R. C. (2000b). An improved error model for noisy channel spelling correction. In *Proceedings of the 38th annual meeting of the association for computational linguistics*, pages 286–293.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Castro Ferreira, T., van der Lee, C., van Miltenburg, E., and Krahmer, E. (2019). Neural data-to-text generation: A comparison between pipeline and end-to-end architectures. In *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP)*, pages 552–562.

Chen, L., Zaharia, M., and Zou, J. (2023). How is chatgpt's behavior changing over time?

Chen, S., Wang, J., Feng, X., Jiang, F., Qin, B., and Lin, C.-Y. (2019). Enhancing neural data-to-text generation models with external background knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3022–3032, Hong Kong, China. Association for Computational Linguistics.

Chen, Z., Eavani, H., Chen, W., Liu, Y., and Wang, W. Y. (2020). Few-shot NLG with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190, Online. Association for Computational Linguistics.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. (2022). Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311.*

Clark, E., August, T., Serrano, S., Haduong, N., Gururangan, S., and Smith, N. A. (2021). All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7282–7296, Online. Association for Computational Linguistics.

Colin, E., Gardent, C., M'rabet, Y., Narayan, S., and Perez-Beltrachini, L. (2016). The WebNLG challenge: Generating text from DBPedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167, Edinburgh, UK. Association for Computational Linguistics.

Dale, R. (2020). Natural language generation: The commercial state of the art in 2020. *Natural Language Engineering*, 26(4):481–487.

Dale, R. (2023). Navigating the text generation revolution: Traditional data-to-text nlg companies and the rise of...

de Vries, H., Bahdanau, D., and Manning, C. D. (2020). Towards ecologically valid research on language user interfaces. *CoRR*, abs/2007.14435.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*

(Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dong, Q., Li, L., Dai, D., Zheng, C., Wu, Z., Chang, B., Sun, X., Xu, J., Li, L., and Sui, Z. (2023). A survey on in-context learning.

Du, W. and Black, A. W. (2019). Boosting dialog response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 38–43, Florence, Italy. Association for Computational Linguistics.

Dubey, N., Chakraborti, S., and Khemani, D. (2018). Textual summarization of time series using case-based reasoning: a case study. In *Workshop on Reasoning about Time in CBR-RATIC*, pages 164–174.

Dudy, S., Bedrick, S., and Webber, B. (2021). Refocusing on relevance: Personalization in NLG. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5190–5202, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Dušek, O., Novikova, J., and Rieser, V. (2018). Findings of the e2e nlg challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328.

Feng, S. Y., Huynh, J., Narisetty, C. P., Hovy, E., and Gangal, V. (2021). SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Ganesan, D. and Chakraborti, S. (2018). An empirical study of knowledge tradeoffs in case-based reasoning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 1817–1823. International Joint Conferences on Artificial Intelligence Organization.

Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. (2017). The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Garneau, N. and Lamontagne, L. (2021). Shared task in evaluating accuracy: Leveraging pre-annotations in the validation process. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 266–270, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Garvin, P. L. (1968). *The Georgetown-IBM experiment of 1954: an evaluation in retrospect.* Mouton.

Gatt, A. and Krahmer, E. (2018). Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Iii, H. D., and Crawford, K. (2021). Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Gehrmann, S., Bhattacharjee, A., Mahendiran, A., Wang, A., Papangelis, A., Madaan, A., McMillan-Major, A., Shvets, A., Upadhyay, A., Yao, B., Wilie, B., Bhagavatula, C., You, C., Thomson, C., Garbacea, C., Wang, D., Deutsch, D., Xiong, D., Jin, D., Gkatzia, D., Radev, D., Clark, E., Durmus, E., Ladhak, F., Ginter, F., Winata, G. I., Strobelt, H., Hayashi, H., Novikova, J., Kanerva, J., Chim, J., Zhou, J., Clive, J., Maynez, J., Sedoc, J., Juraska, J., Dhole, K., Chandu, K. R., Perez-Beltrachini, L., Ribeiro, L. F. R., Tunstall, L., Zhang, L., Pushkarna, M., Creutz, M., White, M., Kale, M. S., Eddine, M. K., Daheim, N., Subramani, N., Dusek, O., Liang, P. P., Ammanamanchi, P. S., Zhu, Q., Puduppully, R., Kriz, R., Shahriyar, R., Cardenas, R., Mahamood, S., Osei, S., Cahyawijaya, S., Štajner, S., Montella, S., Shailza, Jolly, S., Mille, S., Hasan, T., Shen, T., Adewumi, T., Raunak, V., Raheja, V., Nikolaev, V., Tsai, V., Jernite, Y., Xu, Y., Sang, Y., Liu, Y., and Hou, Y. (2022). Gemv2: Multilingual nlg benchmarking in a single line of code.

Gkatzia, D., Hastie, H., and Lemon, O. (2014). Multi-adaptive natural language generation using principal component regression. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 138–142, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Gkatzia, D., Lemon, O., and Rieser, V. (2016). Natural language generation enhances human decision-making with uncertain information. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 264–268, Berlin, Germany. Association for Computational Linguistics.

Goldberg, E., Driedger, N., and Kittredge, R. (1994). Using natural-language processing to produce weather forecasts. *IEEE Expert*, 9(2):45–53.

Gong, H., Feng, X., Qin, B., and Liu, T. (2019a). Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3143–3152, Hong Kong, China. Association for Computational Linguistics.

Gong, H., Feng, X., Qin, B., and Liu, T. (2019b). Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3134–3143.

He, P., Gao, J., and Chen, W. (2023). Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L. (2022). Training compute-optimal large language models.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Hunter, J., Freer, Y., Gatt, A., Reiter, E., Sripada, S., Sykes, C., and Westwater, D. (2011). Bt-nurse:

computer generation of natural language shift summaries from complex heterogeneous medical data. *Journal of the American Medical Informatics Association*, 18(5):621–624.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Kale, M. (2020). Text-to-text pre-training for data-to-text tasks. *arXiv preprint arXiv:2005.10433*.

Kasner, Z., Mille, S., and Dušek, O. (2021). Text-in-context: Token-level error detection for table-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 259–265, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Kennedy, J. (2010). *Particle Swarm Optimization*, pages 760–766. Springer US.

Kondadadi, R., Howald, B., and Schilder, F. (2013). A statistical NLG framework for aggregated planning and realization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1406–1415, Sofia, Bulgaria. Association for Computational Linguistics.

Konstas, I. and Lapata, M. (2013). A global model for concept-to-text generation. *Journal of Artificial Intelligence Research*, 48:305–346.

Kukich, K. (1983). Design of a knowledge-based report generator. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 145–150. Association for Computational Linguistics.

Laha, A., Jain, P., Mishra, A., and Sankaranarayanan, K. (2020). Scalable micro-planned generation of discourse from structured data. *Computational Linguistics*, 45(4):737–763.

Lebret, R., Grangier, D., and Auli, M. (2016). Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, Austin, Texas. Association for Computational Linguistics.

Lee, B. Y. (2016). Doctors wasting over two-thirds of their time doing paperwork.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Liang, P., Jordan, M., and Klein, D. (2009). Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore. Association for Computational Linguistics.

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

Liu, T., Luo, F., Xia, Q., Ma, S., Chang, B., and Sui, Z. (2019a). Hierarchical encoder with auxiliary supervision for neural table-to-text generation: Learning better representation for tables. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6786–6793.

Liu, T., Wang, K., Sha, L., Chang, B., and Sui, Z. (2018). Table-to-text generation by structure-aware seq2seq learning. In *AAAI*.

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

Luccioni, A. S. and Rogers, A. (2023). Mind your language (model): Fact-checking llms and their role in nlp research and practice.

Mahamood, S. and Reiter, E. (2011). Generating affective natural language for parents of neonatal infants. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 12–21.

Malmi, E., Krause, S., Rothe, S., Mirylenka, D., and Severyn, A. (2019). Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065, Hong Kong, China. Association for Computational Linguistics.

Manning, C. D. (2015). Computational linguistics and deep learning. *Computational Linguistics*, 41(4):701–707.

Marcheggiani, D. and Perez-Beltrachini, L. (2018). Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9.

Massie, S., Wiratunga, N., Craw, S., Donati, A., and Vicari, E. (2007). From anomaly reports to cases. In *International Conference on Case-Based Reasoning*, pages 359–373.

Mei, H., Bansal, M., and Walter, M. R. (2016). What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 720–730, San Diego, California. Association for Computational Linguistics.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Moryossef, A., Goldberg, Y., and Dagan, I. (2019a). Improving quality and efficiency in plan-based neural data-to-text generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 377–382, Tokyo, Japan. Association for Computational Linguistics.

Moryossef, A., Goldberg, Y., and Dagan, I. (2019b). Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277, Minneapolis, Minnesota. Association for Computational Linguistics.

Nan, L., Radev, D., Zhang, R., Rau, A., Sivaprasad, A., Hsieh, C., Tang, X., Vyas, A., Verma, N., Krishna, P., Liu, Y., Irwanto, N., Pan, J., Rahman, F., Zaidi, A., Mutuma, M., Tarabar, Y., Gupta, A., Yu, T., Tan, Y. C., Lin, X. V., Xiong, C., Socher, R., and Rajani, N. F. (2021). DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online. Association for Computational Linguistics.

Narayan, S., Maynez, J., Adamek, J., Pighin, D., Bratanic, B., and McDonald, R. (2020). Stepwise extractive summarization and planning with structured transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4143–4159, Online. Association for Computational Linguistics.

Nema, P., Shetty, S., Jain, P., Laha, A., Sankaranarayanan, K., and Khapra, M. M. (2018). Generating descriptions from structured data using a bifocal attention mechanism and gated orthogonalization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1539–1550, New Orleans, Louisiana. Association for Computational Linguistics.

Nie, F., Wang, J., Yao, J.-G., Pan, R., and Lin, C.-Y. (2018). Operation-guided neural networks for high fidelity data-to-text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3879–3889, Brussels, Belgium. Association for Computational Linguistics.

Novikova, J., Dušek, O., and Rieser, V. (2017). The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pauws, S., Gatt, A., Krahmer, E., and Reiter, E. (2019). Making effective use of healthcare data using data-to-text technology. *Data science for healthcare: methodologies and applications*, pages 119–145.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Portet, F., Reiter, E., Gatt, A., Hunter, J., Sripada, S., Freer, Y., and Sykes, C. (2009). Automatic generation of textual summaries from neonatal intensive care data. *Artificial Intelligence*, 173(7-8):789–816.

Puduppully, R., Dong, L., and Lapata, M. (2019a). Data-to-text generation with content selection and planning. In *The Thirty-Third AAAI Conf. on Artificial Intelligence, AAAI 2019*, pages 6908–6915.

Puduppully, R., Dong, L., and Lapata, M. (2019b). Data-to-text generation with entity modeling. In *Proc. of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2023–2035.

Puduppully, R., Fu, Y., and Lapata, M. (2022). Data-to-text generation with variational sequential planning. *Transactions of the Association for Computational Linguistics*, 10:697–715.

Puduppully, R. and Lapata, M. (2021). Data-to-text generation with macro planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

Puduppully, R. S. (2022). *Data-to-text generation with neural planning.* PhD thesis, University of Edinburgh, Scotland, UK.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Rasmy, L., Xiang, Y., Xie, Z., Tao, C., and Zhi, D. (2021). Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):86.

Rebuffel, C., Soulier, L., Scoutheeten, G., and Gallinari, P. (2020). A hierarchical model for data-to-text generation. In *European Conf. on Information Retrieval*, pages 65–80. Springer.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Reiter, E. (2007a). An architecture for data-to-text systems. In *Proc. of the 11th European Workshop on Natural Language Generation*, page 97–104.

Reiter, E. (2007b). An architecture for data-to-text systems. In *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 07)*, pages 97–104, Saarbrücken, Germany. DFKI GmbH.

REITER, E. and DALE, R. (1997). Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems.* Cambridge University Press.

Reiter, E., Robertson, R., Lennox, A. S., and Osman, L. (2001). Using a randomised controlled clinical trial to evaluate an NLG system. In *Proceedings of the 39th Annual Meeting of the Association for*

Computational Linguistics, pages 442–449, Toulouse, France. Association for Computational Linguistics.

Richter, M. M. and Weber, R. (2013). *Case-based reasoning a textbook.* Springer Berlin.

Robin, J. and McKeown, K. (1996). Empirically designing and evaluating a new revision-based model for summary generation. *Artificial Intelligence*, 85(1):135–179.

Ruder, S. (2019). *Neural Transfer Learning for Natural Language Processing.* PhD thesis, National University of Ireland, Galway.

Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., Tow, J., Rush, A. M., Biderman, S., Webson, A., Ammanamanchi, P. S., Wang, T., Sagot, B., Muennighoff, N., del Moral, A. V., Ruwase, O., Bawden, R., Bekman, S., McMillan-Major, A., Beltagy, I., Nguyen, H., Saulnier, L., Tan, S., Suarez, P. O., Sanh, V., Laurençon, H., Jernite, Y., Launay, J., Mitchell, M., Raffel, C., Gokaslan, A., Simhi, A., Soroa, A., Aji, A. F., Alfassy, A., Rogers, A., Nitzav, A. K., Xu, C., Mou, C., Emezue, C., Klamm, C., Leong, C., van Strien, D., Adelani, D. I., Radev, D., Ponferrada, E. G., Levkovizh, E., Kim, E., Natan, E. B., Toni, F. D., Dupont, G., Kruszewski, G., Pistilli, G., Elsahar, H., Benyamina, H., Tran, H., Yu, I., Abdulmumin, I., Johnson, I., Gonzalez-Dios, I., de la Rosa, J., Chim, J., Dodge, J., Zhu, J., Chang, J., Frohberg, J., Tobing, J., Bhattacharjee, J., Almubarak, K., Chen, K., Lo, K., Werra, L. V., Weber, L., Phan, L., allal, L. B., Tanguy, L., Dey, M., Muñoz, M. R., Masoud, M., Grandury, M., Šaško, M., Huang, M., Coavoux, M., Singh, M., Jiang, M. T.-J., Vu, M. C., Jauhar, M. A., Ghaleb, M., Subramani, N., Kassner, N., Khamis, N., Nguyen, O., Espejel, O., de Gibert, O., Villegas, P., Henderson, P., Colombo, P., Amuok, P., Lhoest, Q., Harliman, R., Bommasani, R., López, R. L., Ribeiro, R., Osei, S., Pyysalo, S., Nagel, S., Bose, S., Muhammad, S. H., Sharma, S., Longpre, S., Nikpoor, S., Silberberg, S., Pai, S., Zink, S., Torrent, T. T., Schick, T., Thrush, T., Danchev, V., Nikoulina, V., Laippala, V., Lepercq, V., Prabhu, V., Alyafeai, Z., Talat, Z., Raja, A., Heinzerling, B., Si, C., Taşar, D. E., Salesky, E., Mielke, S. J., Lee, W. Y., Sharma, A., Santilli, A., Chaffin, A., Stiegler, A., Datta, D., Szczechla, E., Chhablani, G., Wang, H., Pandey, H., Strobelt, H., Fries, J. A., Rozen, J., Gao, L., Sutawika, L., Bari, M. S., Al-shaibani, M. S., Manica, M., Nayak, N., Teehan, R., Albanie, S., Shen, S., Ben-David, S., Bach, S. H., Kim, T., Bers, T., Fevry, T., Neeraj, T., Thakker, U., Raunak, V., Tang, X., Yong, Z.-X., Sun, Z., Brody, S., Uri, Y., Tojarieh, H., Roberts, A., Chung, H. W., Tae, J., Phang, J., Press, O., Li, C., Narayanan, D., Bourfoune, H., Casper, J., Rasley, J., Ryabinin, M., Mishra, M., Zhang, M., Shoeybi, M., Peyrounette, M., Patry, N., Tazi, N., Sanseviero, O., von Platen, P., Cornette, P., Lavallée, P. F., Lacroix, R., Rajbhandari, S., Gandhi, S., Smith, S., Requena, S., Patil, S., Dettmers, T., Baruwa, A., Singh, A., Cheveleva, A., Ligozat, A.-L., Subramonian, A., Névéol, A., Lovering, C., Garrette, D., Tunuguntla, D., Reiter, E., Taktasheva, E., Voloshina, E., Bogdanov, E., Winata, G. I., Schoelkopf, H., Kalo, J.-C., Novikova, J., Forde, J. Z., Clive, J., Kasai, J., Kawamura, K., Hazan, L., Carpuat, M., Clinciu, M., Kim, N., Cheng, N., Serikov, O., Antverg, O., van der Wal, O., Zhang, R., Zhang, R., Gehrmann, S., Mirkin, S., Pais, S., Shavrina, T., Scialom, T., Yun, T., Limisiewicz, T., Rieser, V., Protasov, V., Mikhailov, V., Pruksachatkun, Y., Belinkov, Y., Bamberger, Z., Kasner, Z., Rueda, A., Pestana, A., Feizpour, A., Khan, A., Faranak, A., Santos, A., Hevia, A., Unldreaj, A., Aghagol, A., Abdollahi, A., Tammour, A., HajiHosseini, A., Behroozi, B., Ajibade, B., Saxena, B., Ferrandis, C. M., McDuff, D., Contractor, D., Lansky, D., David, D., Kiela, D., Nguyen, D. A., Tan, E., Baylor, E., Ozoani, E., Mirza, F., Ononiwu, F., Rezanejad, H., Jones, H., Bhattacharya, I., Solaiman, I., Sedenko, I., Nejadgholi, I., Passmore, J., Seltzer, J., Sanz, J. B., Dutra, L., Samagaio, M., Elbadri, M., Mieskes, M., Gerchick, M., Akinlolu, M., McKenna, M., Qiu, M., Ghauri, M., Burynok, M., Abrar, N., Rajani, N., Elkott, N., Fahmy, N., Samuel, O., An, R., Kromann, R., Hao, R., Alizadeh, S., Shubber, S., Wang, S., Roy, S., Viguier, S., Le, T., Oyebade, T., Le, T., Yang, Y., Nguyen, Z., Kashyap, A. R., Palasciano, A., Callahan, A., Shukla, A., Miranda-Escalada, A., Singh, A., Beilharz, B., Wang, B., Brito, C., Zhou, C., Jain, C., Xu, C., Fourrier, C., Periñán, D. L., Molano, D., Yu, D., Manjavacas, E., Barth, F., Fuhrimann, F., Altay, G., Bayrak, G., Burns, G., Vrabec, H. U., Bello, I., Dash, I., Kang, J., Giorgi, J., Golde, J., Posada, J. D., Sivaraman, K. R., Bulchandani, L., Liu, L., Shinzato, L., de Bykhovetz, M. H., Takeuchi, M., Pàmies, M., Castillo, M. A., Nezhurina, M., Sänger, M., Samwald, M., Cullan, M., Weinberg, M., Wolf, M. D., Mihaljcic, M., Liu, M., Freidank, M., Kang, M., Seelam, N., Dahlberg, N., Broad, N. M., Muellner, N., Fung, P., Haller, P., Chandrasekhar, R., Eisenberg, R., Martin, R., Canalli, R., Su, R., Su, R., Cahyawijaya, S., Garda, S., Deshmukh, S. S., Mishra, S., Kiblawi, S., Ott, S., Sang-aroonsiri, S., Kumar, S., Schweter, S., Bharati, S., Laud, T., Gigant, T., Kainuma, T., Kusa, W., Labrak, Y., Bajaj, Y. S., Venkatraman, Y., Xu, Y., Xu, Y., Xu, Y., Tan, Z., Xie, Z., Ye, Z., Bras, M., Belkada, Y., and Wolf, T. (2023). Bloom: A 176b-parameter open-access multilingual language model.

Schlichtkrull, M., Kipf, T. N., Bloem, P., Van Den Berg, R., Titov, I., and Welling, M. (2018). Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Settles, B. (2009). *Active learning literature survey.* University of Wisconsin-Madison Department of Computer Sciences.

Sripada, S., Reiter, E., and Davy, I. (2003). Sumtime-mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10.

Thomson, C. and Reiter, E. (2020). A gold standard methodology for evaluating accuracy in data-to-text systems. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 158–168, Dublin, Ireland. Association for Computational Linguistics.

Thomson, C., Reiter, E., and Sripada, S. (2020a). SportSett:basketball - a robust and maintainable dataset for natural language generation. In *Proc. of the Workshop on Intelligent Information Processing and Natural Language Generation.*

Thomson, C., Reiter, E., and Sundararajan, B. (2023). Evaluating factual accuracy in complex data-to-text. *Computer Speech & Language*, 80:101482.

Thomson, C., Zhao, Z., and Sripada, S. (2020b). Studying the impact of filling information gaps on the output quality of neural data-to-text. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 35–40, Dublin, Ireland. Association for Computational Linguistics.

Upadhyay, A. and Massie, S. (2022). Content type profiling of data-to-text generation datasets. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5770–5782, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Upadhyay, A., Massie, S., and Clogher, S. (2020). Case-based approach to automated natural language generation for obituaries. In *Int. Conf. on Case-Based Reasoning*, pages 279–294. Springer.

van der Lee, C., Krahmer, E., and Wubben, S. (2018). Automated learning of templates for data-to-text generation: comparing rule-based, statistical and neural methods. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 35–45. Association for Computational Linguistics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28.

Wang, H. (2019). Revisiting challenges in data-to-text generation with fact grounding. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 311–322, Tokyo, Japan. Association for Computational Linguistics.

Werlen, L. M., Marone, M., and Hassan, H. (2019). Selecting, planning, and rewriting: A modular approach for data-to-document generation and translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 289–296.

Wiseman, S., Shieber, S., and Rush, A. (2017). Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263.

Zhang, J., Zhao, Y., Saleh, M., and Liu, P. (2020a). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339. PMLR.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020b). Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020.* OpenReview.net.

Zhao, C., Walker, M., and Chaturvedi, S. (2020). Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, volume 1.