

Final Version - January 14, 2024

Master Thesis Interaction Technology

From Pit Stops to Paragraphs

Automatic Generation of Formula One
Live Blogs based on Structured Race Data

Jetske Bonenkamp

January 14, 2024

Faculty of Electrical Engineering, Mathematics, and Computer Science

UNIVERSITY
OF TWENTE.



Final Version - January 14, 2024

Author

Jetske Bonenkamp (s2649330)

Graduate Student MSc Interaction Technology

Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS)

University of Twente, Enschede, the Netherlands

Graduation committee

dr. Lorenzo Gatti

Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS)

Human Media Interaction (HMI)

University of Twente, Enschede, the Netherlands

dr. Doina Bucur

Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS)

Datamanagement and Biometrics (DMB)

University of Twente, Enschede, the Netherlands

“The question of whether a computer can think is no more interesting than the question of whether a submarine can swim.”

– Edsger W. Dijkstra

Abstract

Live blogs have been undergoing a major increase in popularity in the field of *Formula One* in recent years. This thesis explores the generation of such blog posts based on structured race data by deploying natural language generation techniques. The study emphasizes the importance of accuracy and perceived level of entertainment in its objective. Using the context of *Formula One* and text generation techniques, as well as the principles of news writing and (sports) commentary, a system architecture was proposed that uses three main components: an event identification model, a blog generation model, and a rephrasing model. The first mentioned component used a rule-based approach to extract noteworthy events from a race data set, which was composed by extracting numerical data from *TFeed*, a Russian platform providing race statistics and visuals, and race control messages from the live timing feature of *Formula One*. It was evaluated by comparing the events mentioned in the posts on *Autosport.com* to the identified events, and achieved a 70% overlap, with the most crucial actions (overtakes, pit stops, retirements, and car events) being correctly identified on nearly all occasions. The output of the event identifier was used as input to the blog generation component, which was based on a large language model pre-trained for data-to-text generation. The model was fine-tuned to the *Formula One* live blog generation task on a set of input-output pairs, established using scraped posts from *Autosport.com* and a linguistic feature extraction model to extract structured data from those posts. The fine-tuning was realized with low-rank adaptation, resulting in blog posts that successfully covered approximately 95% of the subjects, actions, and objects in the data. This was manually evaluated by a point scoring system that compared the events in the input data to those present in the generated posts. Around 5% of the posts contained hallucinations, which can potentially be lowered in future work by applying reinforcement learning. The third model, a language model trained on mixed-quality data, took the informative blog posts as input and rephrased the text with the aim of making the content more entertaining. Readers' perception of the posts prior to and after rephrasing, as well as the perception of human-written posts, was evaluated via an online survey, in which post sequences were rated on entertainment, informativeness, and clarity. Applying a Mann Whitney U-test on the results pointed out that this rephrasing model improved the perceived level of entertainment of the generated posts significantly. No sufficient statistical evidence could be found to state that the posts differed on perceived level of informativeness nor clarity, which also holds for the differences in perceived entertainment between the rephrased and human-written posts. Therefore, further research with a larger number of sequences and/or respondents is recommended.

Preface



We live in an era where it is nearly impossible to imagine a world without computers in it. While our knowledge (i.e., us as humans) and skills have increased over the years, so has - thanks to these same humans - the skills and capabilities of computers. During my thesis, of which you are currently reading the preface, I have had the privilege of exploring the endless possibilities and fascinating performance of the language understanding and generating capabilities of these technological masterminds.

I have been intrigued by the structure of language, and the way humans use it to communicate to one another on a unique level compared to other species, since my teenage years. Combining this interest with my passion for (and knowledge of) programming, researching, and problem solving has made the past semester a truly enjoyable experience. The fact that I could even include another of my interests, i.e., *Formula One*, into the same project has made it a great fit for me and helped me grow as a Master's candidate, potential NLP specialist, but also as a person.

The experience and result would not have been the same without the excellent help and support of my supervisor, dr. Lorenzo Gatti. I highly appreciate the time and effort he spent on guiding me through the process, his enthusiasm during our meetings and corresponding brainstorming, and the way he sometimes reminded me to take small steps so I could 'see the trees through the forest' again and continue with a fresh view. In addition, I would like to thank the chair of my graduation committee, dr. Doina Bucur, for the valuable feedback, support, and rapid replies. I am very grateful to have experienced such great guidance of both committee members during my thesis.

Last but not least, I want to thank my partner, relatives, friends, and colleagues for being a 'listening ear' and/or giving advice when the work got slightly overwhelming or frustrating occasionally, and, of course, you, the reader! I hope that you will enjoy reading my work.

After you turn the page, I will take you along in the full process of the study, including the steps taken and insights gained along the way. The key takeaways of each of the chapters, which you will find in the pine-green boxes, are partially written in a *we* perspective, since I want you, the reader, to take part in the process while reading along.

What are you waiting for? It is time to turn the page! Enjoy the journey of reading through the study on *Automatic Generation of Formula One Live Blogs based on Structured Race Data...*

Jetske Bonenkamp

Contents

Abstract	v
Preface	vi
Contents	vii
THEORETICAL BACKGROUND	1
1 Introduction	2
1.1 Research objective	2
1.2 Thesis outline	3
2 Context	4
2.1 <i>Formula One</i>	4
2.2 Live reporting	5
2.3 Text generation	5
3 Principles	9
3.1 Storytelling	9
3.2 Blogging	9
3.3 News writing	9
3.4 Sports commentary	10
4 Related work	12
4.1 Sports data	12
4.2 <i>Formula One</i>	13
RESEARCH QUESTION RECAP I	15
DATA COLLECTION AND ANALYSIS	16
5 Live blog data	17
5.1 Preference survey	17
5.2 Structure analysis	18
5.3 Collection and processing	18
6 Race data	20
6.1 Requirements	20
6.2 Source selection	21
6.3 Collection and processing	22
6.4 Data overview	24
RESEARCH QUESTION RECAP II	27

SYSTEM ARCHITECTURE	28
7 Setup and components	29
8 Event identification model	30
8.1 Rule-based identification	30
8.2 Outlier detection	31
8.3 Novelty detection	31
9 Blog generation model	33
9.1 Structured data extraction	33
9.2 Language model	37
10 Rephrasing model	39
RESEARCH QUESTION RECAP III	40
EXPERIMENTATION	41
11 Event identification	42
11.1 Event detection rules	42
11.2 Event filtering	45
11.3 Term variety	46
12 Structured data extraction	48
12.1 Large language model	48
12.2 Linguistic feature extraction	49
13 Blog generation	52
13.1 Initial model	52
13.2 Fine-tuned model	52
14 Blog rephrasing	55
RESEARCH QUESTION RECAP IV	56
IMPLEMENTATION AND METHODOLOGY	57
15 Implementation	58
15.1 Event identification model	58
15.2 Text generation model	58
15.3 Rephrasing model	60
16 Evaluation setup	62
16.1 Accuracy	62
16.2 Readers' perception	65
RESEARCH QUESTION RECAP V	66

FINDINGS AND OUTCOME	67
17 Results	68
17.1 Generated blog posts	68
17.2 Evaluation	69
18 Discussion	75
18.1 Data	75
18.2 System architecture	76
19 Conclusion	80
RESEARCH QUESTION RECAP VI	82
Bibliography	83
APPENDIX	90
A Reader preference survey	91
A.1 Introduction	91
A.2 Setup	92
A.3 Results	92
B Research Topics	121
B.1 Data-to-text generation	121
C Live timing data	124
D Experiments	127
D.1 LLaMA	127
E Orientation on Large Language Models	130
F Analysis of blog structure	132
G Event identification model	134
H Experimentation	135
H.1 Structured data extraction with an LLM	135
H.2 Named entity recognition	136
H.3 Dependency parsing	137
I Evaluation survey	139
I.1 Content	139
I.2 Results	141

Acronyms

AI	artificial intelligence
AI-O	original (non-rephrased) AI-generated
AI-R	rephrased AI-generated
API	application programming interface
BGM	blog generation model
CD	content determination
CLM	causal language modeling
CO	content organization
CSV	comma-separated value
DL	deep learning
DP	dependency parsing
DRS	drag reduction system
EIM	event identification model
GPUs	graphics processing unit
HTML	hypertext markup language
HW	human-written
JSON	JavaScript object notation
LFE	linguistic feature extraction
LLM	large language model
LoRA	low-rank adaptation
ML	machine learning
MTL	multi-task learning
MVP	multi-task supervised pre-training
ND	novelty detection
NER	named entity recognition
NLG	natural language generation
NLP	natural language processing
NN	neural network
OD	outlier detection
PEFT	parameter-efficient fine-tuning
PoS	Part-of-Speech
RCM	race control message
RPM	rephrasing model
S2S	sequence-to-sequence modeling
SR	surface realization
SRL	semantic role labelling
SVM	support vector machine
TV	television
UA	untrained automatic
URL	uniform resource locator

List of Figures

6.1	Example of statistics table on the F1 TFeed dashboard during the Austin GP, 2023 [76]	22
6.2	Example of the data extracted from <i>TFeed</i> [76]	23
6.3	Example of extracted RCMs from the <i>live timing</i> API [6]	23
6.4	Example of circuit info file extracted from the <i>Ergast</i> API [5]	24
6.5	Example of driver info objects extracted from the <i>Ergast</i> API [5]	24
6.6	Example of constructor info objects extracted from the <i>Ergast</i> API [5]	24
7.1	Global visualisation of the components in the system architecture	29
8.1	Visualisation of an outlier in a sequence of lap times per driver	31
8.2	Visualisation of a new observation being an outlier	32
9.1	Visualization of the blog generation component	33
9.2	Example of extracting structured data from unstructured product review [81]	34
9.3	Example of the dependencies within a single-sentence blog post	35
9.4	Example of the extracted semantic roles from a blog post sentence	36
11.1	Result of events identified by the experimentation model in a single lap	46
12.1	Example of a structured data extracted from a blog post	50
12.2	Semantic roles identified in an example blog sentence using the AllenNLP demo tool [91]	51
16.1	Overview of the scoring system for evaluating the BGM accuracy	63
16.2	Overview of the scoring system for evaluating the RPM accuracy	64
17.1	Overview of the EIM evaluation results	69
17.2	Overview of the BGM evaluation results	71
17.3	Overview of the EIM evaluation results	72
H.1	Results of the named entities detected in a subset of the blog examples (1)	137
H.2	Results of the named entities detected in a subset of the blog examples (2)	137
H.3	Results of the named entities detected by a further trained NER model	137

List of Tables

5.1	Overview of the blog data set	19
6.1	Live race statistics per driver, per lap	20
6.2	General live statistics per race, per lap	20
6.3	General statistics per driver, per race	21
6.4	Overview of the <i>TFeed</i> time table data set	25
6.5	Overview of the <i>live timing</i> data set	26
6.6	Overview of the <i>Ergast</i> additional data set	26
8.1	Overview of event identification rules	30
9.1	Steps in text-to-data generation	37
13.1	Examples of posts generated by LLMs during experimentation	53
14.1	Examples of posts generated by the <i>OpenChat</i> LLM during experimentation	55
15.1	Implementation of the event identifier	59
15.2	Prioritization of race control events	60
17.1	Sample of blog posts generated by the LLM	68
17.2	U-values of the Mann-Whitney U-test on level of entertainment	73
E.1	Overview of Large Language Models	130
H.1	Results of data extraction experiment with Falcon LLM (1)	135
H.2	Results of data extraction experiment with Falcon LLM (2)	136
I.1	Results of the Jarque-Bera test on the evaluation survey results	142
I.2	U-values of the Mann-Whitney U-test on level of informativeness	142
I.3	U-values of the Mann-Whitney U-test on level of clarity	142

THEORETICAL BACKGROUND

1

Introduction

What are we trying to achieve?

1.1 Research objective	2
1.2 Thesis outline	3

1: We use the term *live blogs* to refer to textual update posts that are shared throughout the race to communicate important events to readers.

The motorsports discipline *Formula One* has seen a large increase in popularity throughout the past few years [1] [2] [3]. This has led to higher costs of streaming provider subscriptions and more races being added to the calendar, limiting the possibility to visually watch the races for many race fans [4]. This has led to an increased public desire for access to news, standings, and other information related to the sport. Within *Formula One*, (live) data sets - containing records such as lap times and intervals - are publicly available via various sources [5] [6], while visual statistics are provided by online platforms as well. It can be challenging and time-consuming, however, for people to extract the information that is important.

Live blogs¹ have emerged as alternatives, but creation of such content can be time-consuming, error-prone, and biased towards certain teams or drivers. These flaws can potentially be overcome by deploying natural language generation (NLG) techniques, as is seen more and more in (sports) reporting nowadays [7] [8]. NLG is a subfield of the artificial intelligence (AI) branch natural language processing (NLP). It focuses, as its name indicates, on generating natural language [9] [10]. It has seen an increase in popularity among researchers over the last decade [11], with the technologies continuing to advance and the capabilities of existing NLG systems increasing, mainly thanks to the rising amount of data available and the emerge of machine learning (ML) models.

Automating the writing procedure not only has potential of speeding up the process [12], it also comes with economic benefits: AI systems can generate text at larger scale - and therefore lower cost - than human writers [13]. Furthermore, it has potential to improve the quality and readability of the text. Computer systems are less likely to overlook facts than humans and are less error-prone, mainly because they are not exposed to the negative effects of, e.g., getting tired [12]. Although the expectation formulated by Levy [14] in 2012 (i.e., that more than 90% of news would be computer-generated by 2025) is not likely to be met, the advantages of automatically generated text, especially in fields that involve structured and rich data, among which sports, are seen as human-outperforming candidates - especially for routine tasks [15].

In short, the study covered in this thesis aims at finding a solution for automatically generating live blogs based on structured data retrieved during *Formula One* races by deploying NLG techniques. In sections 1.1 and 1.2, further details about the objective of the study and the outline of the document can be found.

1.1 Research objective

To achieve this aim, a main research question has been formulated, supported by four sub-questions of which the answers can together lead to an answer to the main question, which is indicated by its **bold** font.

RQ1. How can accurate and entertaining blog posts be generated based on structured, *Formula One* race data?

RQ2. What is needed for the generated posts to be accurate?

- 2.1. When can a blog post be considered as accurate (enough)?
- 2.2. How can the accuracy of the blog posts be evaluated?

2.3. To what extent can the accuracy of human-written blog posts be matched?

RQ3. What is needed for the generated posts to be entertaining?

- 3.1. When can a blog post be considered as entertaining (enough)?
- 3.2. What level of detail is preferred by potential readers?
- 3.3. What writing style is preferred by potential readers?
- 3.4. How can the level of entertainment of the blog posts be evaluated?
- 3.5. To what extent can the level of entertainment of human-written blog posts be matched?

RQ4. What main components are needed in the system architecture?

- 4.1. What task(s) does each component perform?
- 4.2. How do the system components interact?
- 4.3. How can the performance of each component be evaluated?

RQ5. What (language) model is most suitable for data-to-blog generation?

- 5.1. What (types of) models are available?
- 5.2. What input (format) does the model take?
- 5.3. What is needed for the model to fit the desired task?

RQ6. What steps are needed for collecting and processing the data?

- 6.1. What are the requirements for the data?
- 6.2. What sources contain the desired data?
- 6.3. What data processing techniques are needed?
- 6.4. How can the impact of the data on the system's performance be evaluated?
- 6.5. To what extent can the data be parsed in real time?



The research questions we have presented help us to make clear what we are trying to achieve during the thesis work. Using the questions, which can each be seen as separate objectives, we want to end up with a framework for automatically generating blog posts based on structured data that is extracted during *Formula One* races. Along the way, we want to gain insights in the overall guidelines and methods for creating *accurate* and *entertaining* content, the tools to evaluate these characteristics, and potential readers' preferences. Furthermore, we want to study the possibilities for the system architecture, and determine what components and tasks are needed, what data is needed, and how the data can be collected and processed.

i To highlight key takeaways of certain chapters and/or sections, we will use these pinpointed boxes. Besides, we will use them to reflect on the question we have asked ourselves in the beginning of the chapter, and to point out what impact the new information has on the remaining part of the study.

1.2 Thesis outline

The thesis is split up into six parts. In the current chapter, up to chapter 4, a theoretical background to the study is provided, where an explanation of the context of related topics is given, followed by a summary of principles that are of use in text generation, and the thesis part is closed by a chapter that discusses previous work with similar intentions.

In the second part, the process of collecting, analyzing, and processing data is discussed, followed by a third part that involves the setup of the system architecture. Once the latter has been defined, the subsequent part will dive into experiments that were done to get insights in the feasibility of the proposed system and defines the steps to be taken during the fifth part, which is about implementing the system. Besides implementation, the regarding part discusses the evaluation setup in chapter 16.

In the final part, the results and findings of the study can be found. After interpreting and discussing the results in chapter 18, a final conclusion of the work and a summary of the insights gained during the thesis are provided in the closing chapter.

2

Context

What are relevant topics for our study and why?

2.1 <i>Formula One</i>	4
2.2 Live reporting	5
2.3 Text generation	5

Busy schedules of many individuals have led to a desire for increased efficiency in spread of news articles and/or entertaining content. The popularity of the verb *multi-tasking* has risen: doing multiple things at once while staying up-to-date about as much as possible without wasting time has become the norm. In the field of *Formula One*, this phenomena leads to more and more people following races via news and/or social media, rather than scheduling their Sunday afternoon free to watch the race on a television (TV) screen. Currently, many of these people read human-written live blogs via one or more online platforms. In an ideal future scenario, such live blogs would also be available in automatically generated form.

In this chapter, the main concepts that are related to the objective are introduced, where background information that helps to identify the problems and understand the intentions is provided.

2.1 *Formula One*

The car racing discipline *Formula One* can be seen as the 'elite top' of motor racing. Constructors (or teams) and drivers compete against one another on over 20 races per year via so-called Grands Prix. The discipline has been existing for over 70 years, with large developments in mechanical, aerodynamical, and technological areas that have made it to be the highly advanced sport it is nowadays [16]. Required pit stops, various strategies, challenging overtakes, differences among cars on different tracks, and way more factors make it interesting for many people to follow the sport closely. The combination of racing at extremely high speeds and continuous analysis until the tiniest detail by the teams' specialist engineers and strategists is what, according to many, distinguishes the sport from other racing disciplines and makes it one of the biggest sports events worldwide¹ [19].

Besides the spectators physically present at the Grands Prix, the average number of people following races via a visual live stream is 70,3 million and is still increasing [18], while it is popular on social media² as well. The latter indicates that people not only show interest in the races themselves, but also in other available media related to the sport. On top of the race footage, race viewers are informed via live audio commentary and provided with statistics via boxes that are displayed on their TV screen. In addition, they have access to separate, more detailed real-time statistics via online platforms, such as the *F1 TV* platform [20]. Examples of the data and statistics shared to the public are drivers' positions, lap- and sector times, tyre compounds, nationality, championship standings, number of pit stops done, overtake predictions, and more.

1: In 2006, the average number of spectators was over 120,000 [17], and in 2021 multiple races welcomed over 300,000 individuals on track [18].

2: *Formula One* even was the 'fastest-growing major sports league' [18] with nearly 50 million followers in 2021.

3: Those that are publicly available.




Section 2.1 has taught us that *Formula One* is a popular - and still growing - sports discipline that engages people by its wide range of facets impacting performance and results. We now also know that live data availability is an important aspect of the sport. A new insight we now have is that the general purpose of such live statistics³ is to make it easier for people to follow and understand the events going on during a race.

2.2 Live reporting

Online textual content has seen a large rise since the early 2000s [21]. Over the years, a shift from longer news articles and blogs to a new term called *micro-blogging*, especially via Twitter⁴, was seen [23]. The cause for the popularity of this new phenomenon was the ability to read and share breaking news and gossip in a fast and simple way [24]. In the professional industry, such as for news platforms or sports federations, this type of content spreading also became a standard [25]. Updates during important events or sports games were shared within minutes to keep anyone that was interested updated without the need to be present or follow it visually. While some organisations solely used Twitter for this purpose, others created their own separate platform or used a part of their website to share updates via a new type of content: live blogs [26].

Nowadays, live blogs are very common in various types of sports, ranging from football [28] to equestrian disciplines [29] and motor sports. In the latter, especially in *Formula One*, such updates are mainly offered via news platforms specialized in car racing [30] [31] [32] [33]. They are offered in addition to their regular news articles, which are posted at any moment. The *live* blogs, however, as the name indicates, are posted close to real-time during and shortly after the Grands Prix. During any session, the editors of the platform keep a close eye on what is happening in the race and try to update the reader as fast and accurate as possible via blog posts ranging from a single sentence to a short, few-sentence long story. Example 2.1 shows such a blog post during the Grand Prix in Las Vegas, 2023.

 19 November at 07:24

PIT-STOPS

On Lap 16, Russell dives in for a fresh set of Hards as Verstappen defends from Leclerc into Turn 5.

These live blogs are ideal for race fans who do not have the access or time to watch the live stream, as they can simply keep an eye on the blog feed and stay up-to-date about what is going on in the race. However, due to the fact that the editors try to inform their readers as fast as possible, human errors are not uncommon in the blogs. In addition, there will always be a delay of a few minutes as the posts need to be written and checked manually.





From section 2.2, we have learned that live blogs are a popular way of sports reporting, with various platforms offering such content during *Formula One* races. We now know that it takes several minutes for those human-written blog posts to be published after the event occurs: this can be seen as one of the points where an automatic blog generation system might outperform a human writer. To develop a qualitative system, however, knowledge on the preferences of readers - and their reasoning to read or skip certain blog platforms - is needed. Therefore, we know that some further research, e.g., a survey, is needed to base our further choices on.

2.3 Text generation


Humans communicate primarily through language, which is complex due to its rich grammar, vocabulary, syntax, semantics, context, and expressions. In contrast, computers operate on structured data and strict rules, making it challenging to understand human language and generate it on the same level as a human. Fortunately,

⁴ Twitter has been rebranded and is named X since 2023 [22].

 Part of the content in section 2.2 is reused material from the *Research Topics* report [27].

 A *Formula One* Grand Prix includes multiple sessions: free practice, qualifying, and (sprint) races [16].

Example 2.1: Example of blog post on RacingNews365 [31]

 In many countries, one needs to have a subscription to a streaming platform to be able to watch the races [34].

many years of NLP research have created a bridge between human language and computer systems, enabling computers to learn to understand human language patterns and opening doors to domains like information retrieval and sentiment analysis [35].

Advancements in the NLP field go back to the second half of the previous century. While in the early stages, the techniques relied on manual processing rules, a shift took place over the years where the methods are mainly data-driven and make use of probabilistic models, which allowed for more qualitative processing of language. This shift was supported by the rise of large textual data sets and evaluation metrics for assessing processing performance [35]. These advancements have, together with the rise of deep learning (DL) approaches and large language model (LLM) techniques, lead to an expansion of the domains and applications within the field. While most research previously involved language understanding, more studies have been diving into NLG nowadays [11] [36]. NLG techniques have shown success for various applications, such as machine translation, summarization, and dialogue systems [37].

In below subsections, the stages in state-of-the-art NLG systems are discussed, followed by more context to the briefly mentioned LLMs.

2.3.1 Stages of language generation

Three main steps are present in common NLG architectures: content determination (CD), content organization (CO), and surface realization (SR) [38]. The first two steps are often referred to as a single stage: *content planning*. Below paragraphs discuss the details of these two stages.

Content planning

In general, readers prefer lower-quality content that is correct and appropriate over nicely styled text with inappropriate or inaccurate content [39]. This highlights the importance of the first step mentioned: CD, which involves the decision on the information to be communicated in the text. CD can be used to choose suitable content from the input data based on the semantic message to be shared. This can often be achieved by various content units representing the same communicative goal [38]. To support the message, the system often also needs external data, for example via knowledge bases with background information [40].

Ideally, CD would be based on deep interpretation of the goal, the readers' intentions, and the context [41], which is difficult to achieve in a robust manner. Prior studies on finding effective CD solutions were mainly based on experience and did not prove success in more than a single application, which led to Sripada et al. [39] proposing a general CD architecture using two reasoners. The domain reasoner interprets the initial data and creates an overview that includes supportive knowledge, while the communication reasoner takes the new overview together with the initial data (and potential constraints) to feed a final content unit to further parts of the NLG architecture. The importance of the data overview combined with additional knowledge is emphasized in the mentioned study, which is inspired by the process that human experts would follow when selecting content.

Once the content has been determined, the content needs to be organized before it can be used to generate text directly. The content is grouped so that the units form phrases. These phrases are used to form sentences and paragraphs, which are then ordered within their group. In addition, the dependencies between the different groups are determined in the stage of CO [38].

Surface realization

By the end of the CO stage, the content is organized, but not yet in natural language form. SR takes care of representing the content in a syntactically correct way, with appropriate wording to present the intended communicative goal [38]. SR can be divided into three smaller components: syntactic realization, morphological realization, and orthographic realization. The first is responsible for identifying the input data structure. The latter two focus on wording, punctuation, and formatting [42]. SR can be achieved via various approaches, among which the use of LLMs has been gaining popularity, especially in comparison to the more traditional rule-based approach [42] [43]. In section 2.3.2, such language models are discussed in further detail.



The knowledge we have obtained in subsection 2.3.1 has highlighted the advantages of splitting the system into different tasks and modules. In our system, e.g., one module could be responsible for filtering and organizing noteworthy data records, while another takes care of the text generation task. Our new insights will particularly be used in the thesis part about system architecture.

2.3.2 Large Language Models

Whereas template-based and planning-based approaches⁵ to NLG used to be the "go-to" method, state-of-the-art models in the field of NLG are based on *Transformer* architectures. Such architectures have outperformed prior NLG approaches by large margins [44], which gave a boost to its popularity and drove the rise of LLMs.

In simple terms, an LLM is a ML-based model that has the core task of predicting and generating human language [45] that is an appropriate fit to the context [46]. The models make use of parameters, i.e., the weights that the model learns while it is trained. Distinguishing a 'small' language model from an LLM is based on the number of parameters it contains. A model with around 100 million parameters is already seen as an LLM, but the number can range up to hundreds of billions for extremely advanced models [45]. Due to them being trained on large-scale text corpora, LLMs are able to learn patterns of high complexity, linguistic distinctions, and grammatical relationships within text [46].

Training an LLM from scratch is an extremely costly task, which is therefore often done by large institutes, i.e., tech companies and universities, after which the models can be adopted and fine-tuned by smaller parties [47]. Many LLMs are available online⁶, each with their own capabilities and specialism. Whereas an LLM with tens of billions of parameters can be ideal for all-purpose chatbot or complex NLG/dialogue applications, narrow tasks and computational limitations can lead to smaller LLMs being better suited in others.

Besides the size of an LLM, there are also different *types* of LLMs available, which depends on the method used during training. For the blog generation task, two methods that are potentially a good match are causal language modeling (CLM) and sequence-to-sequence modeling (S2S). LLMs based on CLM are auto-regressive and generate language by predicting the next token given the previous tokens. S2S models, on the other hand, use encoders to process the input and decoders to generate the output sequences. Overall, CLM is mainly used for text generation and summarization, while S2S is best suited for question-answering and summarization [49]. In appendix E, a comparison of some (relatively large) LLMs can be found.

5: In the *Research Topics* report [27], various approaches for data-to-text generation were introduced and compared. The content of the regarding section can be found in appendix section B.1.

6: Many LLMs are open source, but the more high-advanced models such as GPT-4 [48] require paid subscriptions.

Both of the mentioned LLM training methods have their own (dis)advantages in certain contexts. Considering computational and cost-related limitations of the thesis work, using excessively large and/or costly models is written off. Various models, including both CLM and S2S trained ones, are still among the options. Based on the theoretical study (appendix E), good potential candidates are the Open Pre-trained Transformer models (*OPT*) [50], which are open source CLM LLMs available with a wide range of parameters, or *ProphetNet* [51], an S2S based model that is known to perform well in text generation, particularly in abstract summarization [49].



Section 2.3.2 provided us with the insight that there are two LLM types that might help reaching our objective: CLM- and S2S based models. Although this theoretical research has guided us into a certain direction, experimentation with various models is needed to conclude what model works best for our task in a later stage. In the experimentation part of the thesis, we will get back to this topic.

2.3.3 Evaluation

In broad lines, the quality of an NLP system can be determined by the level of conciseness, the extent to which the input data is covered, and the absence of false information [52]. In the case of live *Formula One* blogs, a high quality blog would, therefore, be one that (1) solely covers events present in the input data (i.e., it does not include random other drivers or an imaginary action), and (2) covers *all* interesting events present in the input data.

In addition, there are some factors related to the perception of the readers that impact the system's performance. Overall, the main factors to evaluate include credibility (i.e., how informative the text is perceived, but also the accuracy as described in the previous paragraph) and readability (i.e., whether the reader finds the text clear, coherent, and does not get bored by it) [53] [12].

Measuring whether the generated blogs meet these criteria can be challenging due to the open-ended nature of most NLG tasks. This leads to, in contrast to many other fields, human evaluation still being the standard evaluation approach for NLG systems, in which human judges are asked to rate the generated texts and/or compare them to human-written ones [54]. In recent years, two alternative methods have emerged, which do not require human labour: untrained automatic (UA) metrics and ML metrics. The UA approach simply measures the similarity and/or overlap of machine- and human-generated text based on equal input data. The other approach also measures similarity, this time using ML, making it come closer to human judges compared to UA evaluation.

In general, human evaluation gives more reliable insight in the performance of a model [54]. UA or ML evaluation, however, can provide more quantitative evaluation results when there is limited time or money for an extensive human evaluation process. In such cases, a combination of both approaches can lead to the most useful results.



Our work can be seen as an example of one that has limitations regarding time and money for evaluation. Therefore, human evaluation - potentially supported by some ML-based methods - is seen as the most appropriate approach in our scenario. This can be done in the form of a survey gathering readers' opinions on the generated blog posts and/or a session with (a) human judge(s) to evaluate the accuracy of the events included.

To ensure readers follow updates during races, it is crucial to consider writing style principles. This chapter, therefore, discusses the principles of writing stories, blog items, news articles, and (sports) commentary, of which relevant concepts are combined to set guidelines for the blog generation system.

3.1 Storytelling	9
3.2 Blogging	9
3.3 News writing	9
3.4 Sports commentary	10

3.1 Storytelling

Writing a fictional story differs significantly from writing an informative blog post. For instance, a story is longer, with themes, characters, and conflicts, while an informative blog post is shorter and based on factual data. However, both require storytelling concepts to capture and hold readers' attention. Storytelling principles that need to be taken into account for live blog writing include achieving engagement, researching the topic properly, including accurate facts, and emphasizing relevant and entertaining aspects [55]. Furthermore, it is crucial to have a clear subject, i.e., that it is transparent who or what is the main aspect of the sentence/paragraph, and to ensure a logical text structure, so that a clear and engaging reader experience can be established.

3.2 Blogging

When talking about a blog, one generally refers to regularly updated web pages that give insight into a specific topic. The popularity of blogs is caused by its potential to realize online communities where individuals can feel a part of [56]. Considering that there are many bloggers worldwide¹, with each their own communication style, it can be a challenge to formulate what distinguishes a good blog from a poor one. What is often recommended, however, is to hook the attention and stimulate curiosity of the reader [58]. It is important that this is done directly, preferably in the first sentence, to avoid the reader aborting their plan to read the blog beforehand.

1: In the United States only, the number of individual bloggers already exceeds 30 million [57].

3.3 News writing

In journalism, the concepts of storytelling - as discussed in section 3.1 - play an important role in distinguishing between *good* and *great* news articles. The foundation of good news is finding and validating what information is or can be interesting to the public, followed by presenting it in a way that captures the reader [59]. A commonly made mistake is to consider news to be of value without studying it from the perspective of the reader [60]. Within the content of a news article, there needs to be a balance between newsworthiness and *noteworthiness* [60]. By the term newsworthy, one refers to a topic or event being of sufficient importance for reporting in the media, whereas noteworthy often refers to a smaller piece within that topic or event that either impacts the bigger picture, or is key information to follow the line of the article.

Determining what information is news- and noteworthy can be a challenge, which also holds for live blog posts, as the data that represents an interesting event needs to be identified. There are a few factors that should be present in a story to make it more newsworthy: *impact*, *conflict*, *accidents*, *proximity*, *prominence*, and *novelty* [61]. These factors emphasize that events that have consequences for the future, involve conflicts

or accidents, have some relation to the reader, involve well-known persons or parties, or are unusual draw readers' interest.

3.4 Sports commentary

Sports commentary can be divided into several forms, ranging from the audio you hear while watching a game to columns of various paragraphs where the author discusses and reflects the course of a game that happened in the past. In general, it is referred to as content that 'reads as if it were being spoken' [62]. The overlap between sports commentary and live blogs is that they both involve written updates given during a match, game, or race. According to Lewandowski [63], sports commentary can be seen as a combination of oral TV commentary and written reports. In below subsections, the characteristics of each of these types of commentary are discussed.

3.4.1 Written reports

Sports reports written after the event can be split up into two phases of discourse [64]. First of all, the reports contain an objective part, in which solely the events and actions are described. In addition, a reflection and the opinion or personal view of the author is included. This type of commentary is usually addressed to a large audience and can be simply seen as a written monologue [63]. Generally, the author expects some shared knowledge within the domain from the readers, so the presence of specialist terms is relatively high. Furthermore, compared to oral commentary, the vocabulary is relatively small, the language is more formal, and the syntactic patterns used are more complex [63].

3.4.2 Oral commentary

According to Ferguson [65], oral sports commentary can be seen as the verbal reporting of an event while it is taking place, where the reporting is combined with and supported by related background information and interpretations of the commentator. Overall, such commentary can be divided into two parts: *play-to-play* commentary, which involves updating listeners on the occurring actions, and *colour* commentary, which involves the moments between interesting actions, which is filled with entertaining speech [66].

2: This is in contrast to written reports, where nearly all phrases are in Past Tense [63]

Oral sports commentary tends to be done in a universal setup, where two commentators are working together, among which there is a commentator and an expert. The commentator simply reports the actions, while the expert supports by reflecting on them. Remarkable in this type of commentary is the use of domain-specific terms (including slang), the high occurrence of phrases in Present Simple Tense², the use of passive structures, subject simplification, copula *be* deletion (i.e. *person disqualified* instead of *person is disqualified*), and the use of demonstrative pronouns like *this* or *that* [67]. Overall, known guidelines include that the commentators speak fast and concise, use informal language, include excitement in their voice, and follow the principles of *vignette-quality* language, i.e. make it similar to how an image caption would look [63].



From the discussed principles of storytelling and blogging, we will take into account the importance of background research and/or topic knowledge, and the need for capturing and maintaining readers' attention in the first sentence. Furthermore, we learned that the blog posts should include events that (likely) impact the remainder of the race, are conflicts or accidents (e.g., steward investigations or crashes), involve prominent drivers (e.g., the reigning world champion), or are unusual (e.g., a backmarker³ leading the race). Lastly, we gained the insight that textual updates should mainly consist of Present Simple Tense phrases. We should specifically take the knowledge we have obtained in this chapter into account during the data selection stage.

3: The *Formula One* term *backmarker* refers to a driver that is usually lower in the standings.

4

Related work

What have others done in our field already?

4.1 Sports data	12
4.2 Formula One	13

In the crossroad of NLP and sports, automatic generation of textual language (often based on structured data sources) has gotten more and more attention among researchers over the years. In the current chapter, the first section will dive into what has previously been done in the sports field in general, followed by a more narrowed down section where work on automatic text generation in the field of racing sports and *Formula One* is discussed further.

4.1 Sports data

Whereas sports used to rely on human judgment and spectators physically present at the scene, and was mainly played locally, one can now follow whatever sports they are interested in live, via visual live streams, radio, TV, or news platforms. In addition, detailed statistics, such as rankings, championship standings, records, timing data, and many more, are available all over the internet for people to analyze them [68]. This data is not only used for direct analysis: transformations in data science and AI have lead to such data being used for many more purposes, including sports reporting. As pointed out by Galily [7] and Latar [8], sports reporting can be majorly transformed by AI, specifically considering new data collection and analysis approaches and automated content creation. Automated generation of sports content can solve the issue of lack of time to meet the desire of readers to get immediate updates [7], improve content quality, generate material faster, fill in coverage gaps, and offer insights that human journalists might overlook [8]. However, drawbacks mentioned are that automatically generated sports articles might produce biased or deceptive information [7] and it may lack inventiveness and originality compared to human-written content [8].

Prior studies have proposed systems that generate content automatically in the sports field. Wang et al. [69] studied the automatic generation of sports summaries based on structured data with an ML approach, where they trained a pipeline to retrieve the semantic meaning from input data, which has the option of fine-tuning it for new statistics and/or other sports. They compared their results to the more traditional template-based approach on perplexity scores and were able to conclude that the narratives generated by their ML model performed better in terms of readability and preservation of key information. Another study involving the template-based approach is the study of PASS by van der Lee et al. [70]. By automatically inserting football match data in pre-defined templates, accurate text could be generated, although it was experienced to lack in engagement and personalisation [70].

Gong et al. [71] used NLP techniques combined with knowledge rules for creating a system to produce sports articles. Their findings demonstrated that the system was capable of producing news stories that were both educational and cohesive. However, they do mention that using such knowledge rules may restrict adaptability and suitability for other sports or news events. Instead of the knowledge rules of Gong et al., Aires [72] described a strategy where NLP and ML methods were combined to produce sports news articles. The steps discussed in their paper involve data collection (sports data gathered from various sources, including official websites and social media platforms), text normalization, entity recognition, event detection, and text production.

♻️ Part of the content in sections 4.1 and 4.2 is reused material from the *Research Topics* report [27].



From section 4.1, there are a few aspects we can consider as ‘key takeaways’. First of all, we can now conclude that using software instead of human reporters for creating sports-related news articles offers opportunities for making the process faster and potentially even making the content better. Although various approaches can generate accurate content, ML based approaches have most potential for factual and engaging stories. However, we must pay attention to precautions on biased or deceptive claims being generated, and we should stimulate inventiveness and originality of the content by fine-tuning the model on data with enough variety and originality in writing style. Last but not least, we should focus on engagement and personalisation, which can be achieved by the principles previously discussed in section 3.4.

4.2 Formula One

Prior work on NLG applications in motorsports is limited. Some studies can be found which focused on automatic text generation for racing *games*, however, such as the work done by Ishigaki et al. [73]. In their work, they attempted to automatically generate commentary based on vision, numerical, and textual data obtained during a racing game. They discuss that live commentary "should describe each important event in the race at the moment when the event occurs, within a short period of time" [73] and that the two main tasks for the generation of live commentary involve timing identification and utterance generation. For their experiment, the researchers collected structured telemetry data including the cars’ speeds and steering wheel angles, and they collected transcripts of spoken commentary and corresponding racing game recordings, created by hired workers. The utterances are eventually generated by inputting tuples of a video frame, structured telemetry data, and previous commentary in textual form into an architecture with a multimodal encoder and a duo of decoders if the corresponding timepoint is classified as positive. A point in time is seen as positive if it is considered to be relevant to produce an utterance there, which is the case when the probability output of a linear transformation and softmax function on that timepoint exceeds a threshold or if no utterance has been generated in over 7 seconds to avoid excessively long pauses in commentary. The utterances are generated using an LSTM character model, which was the most commonly used neural network (NN) technique for generating text before the rise of Transformer architectures, as discussed in section 2.3.2. The results of the study show that it is challenging to combine multiple factors into single utterances and that it is crucial to get a clear vision on what makes good commentary/reporting. Another important aspect mentioned in the study of Ishigaki et al. is that timing identification is a highly important task for commentary during races, because races cannot simply be divided into segments, which is in contrast to team sports like soccer or baseball.

More related to report generation in *Formula One* is the bachelor thesis of Krijnen [74]. In 2021, Krijnen did his bachelor thesis on a similar topic, where he presented a system based on the language model GPT-2 to automatically generate reports of *Formula One* races. The process is divided into three main phases: collecting the data, fine-tuning the language model, and finally evaluating the generated content. For data collection, Krijnen used the Ergast application programming interface (API), which contains all *Formula One* race data from the second half of the last century up until 2023 (the current season at the time of writing) [5]. The API was chosen because it contained most of the data needed and it was publicly available. The data that is extracted from this API involves data from specific races and includes the top three drivers’ names, the name of the constructor that won the race, the name of the race itself, the location of the race, the year in which the race took place, and other data that was not used for the project.

To eventually generate the text for the report, Krijnen scraped nearly 400 existing reports from a medium called *Crash.net* [32]. The choice for this website is based on the ease of scraping and the language used [74]. Using these reports, the GPT-2 language model could be fine-tuned so that it was able to understand the specific writing style of race reports, including field jargon and names of the teams and drivers. During the evaluation, the generated reports were tested on "catchyness, factuality, repetitiveness, fluency, grammaticality, and cohesiveness" [74]. Based on the results, Krijnen claimed to demonstrate that it is feasible to produce reports that are more fluent and as grammatically correct, catchy, and coherent compared to reports manually written by humans, although it had some shortcomings in terms of sticking with the facts and tended to generate repetitive content. The reliability of the results, however, is questionable. Krijnen compared a very limited number of reports and the number of respondents in his survey cannot be found in the report. Therefore, no valid conclusions can be drawn regarding the actual quality of the reports generated by the language model in his work. Furthermore, as Krijnen himself also mentioned in his thesis, the generated reports were short of length. This was mainly due to the limited amount of data that could be extracted from the Ergast API [5] and the limited time available for the thesis work.



Taking into account the work done by Ishigaki et al., we know that identification of timing plays a crucial role in establishing a successful commentary generation model. Correct timing identification can, again, be achieved by carefully training and fine-tuning a model so that it learns on what patterns in the structured data human writers created posts and how frequently these should be generated. Furthermore, the questionable outcome of the work of Krijnen following the limited support for the claims made in the paper, it is safe to say that there is room for improvement and expansion of his work, which highlights the potential of our study.



Research question recap

Part I - Theoretical Background

🔗 RQ2. What is needed for the generated posts to be *accurate*?

✅ 2.1. When can a blog post be considered as accurate (enough)?

When the blog post solely contains *factual* information, retrieved by the generation model through either the input data or the knowledge obtained through finetuning.

🔗 2.2. How can the accuracy of the blog posts be evaluated?

By comparing the events covered in the generated blog posts to the input data and checking if it did not include imaginary events and/or information.

🔗 RQ3. What is needed for the generated posts to be *entertaining*?

✅ 3.1. When can a blog post be considered as entertaining (enough)?

When the level of detail and writing style of the blog post is of equal quality compared to those of popular human-written live blog posts.

✅ 3.4. How can the level of entertainment be evaluated?

Via a survey that lets participants rate blog posts and compare the scores of the generated posts to human-written ones.

🔗 RQ4. What main components are needed in the system architecture?

🔗 4.1. What tasks does each component perform?

At least one component is responsible for the task of determining and filtering interesting content from the structured race data, and at least one different component is responsible for generating the blog posts in natural language.

🔗 RQ5. What (language) model is most suitable for data-to-blog generation?

🔗 5.1. What suitable (types of) models are available?

LLMs pre-trained with either a CLM or S2S approach have potential to perform data-to-text generation.

🔗 5.3. What is needed for the model to fit the desired task?

Regardless of the specific model, it needs to be fine-tuned to the *Formula One* and live blogging domain.

i We will be using these *research question recap* pages to reflect on what we have learned in the previous part of the thesis and what questions can potentially already be answered based on the knowledge we have so far.

i A question preceded by a 🔗 icon indicates that the question is *partially* answered. A ✅ icon indicates the question has been answered fully.

DATA COLLECTION AND ANALYSIS

To generate update posts based on structured race data, a language model needs to be fine-tuned on existing, human-written blog posts so that the model learns in what style it should generate content. To realize this fine-tuning process, a corpus of live blog examples is needed. Since such a data set does not exist at the time of writing, it is established using web scraping techniques.

Multiple racing news platforms offer live blogs that each use their own styles and approaches. Some focus on essential information, others use story-like posts, and some use multiple media types like embedded content or images. The platforms considered for the study are those providing updates in purely textual form. Research on available platforms that provide live blogs, with the main decisive factors being (1) English language, (2) limited posts including external sources (e.g., videos, images, embedded content, or links to other pages), and (3) clear and concise posts of which the events can overall be, in theory, extracted from structured data, has led to an initial selection of three sources: *RacingNews365* [31], *PlanetF1* [75], and *Autosport.com* [33]. In the following sections, further research on these sources is discussed, leading to a final choice for the blog data set.

5.1 Preference survey

To get an understanding of the needs and preferences of potential readers, an online survey has been shared via various platforms. In this questionnaire, the 25 respondents answered questions about their opinion on writing style of and content covered in blog posts from the three platforms introduced in the previous paragraph. Nearly all respondents (which are all regular *Formula One* followers) indicated that they know about the concept of live blogs and know how and where to access them. The official *Formula One* platform, together with *RacingNews365*, were chosen as the most popular platforms based on prior experience.

Based on the blog post ratings, however, *Autosport.com* came out as the clear winner. Their posts were praised for their writing style, level of detail, readability, and information covered. *PlanetF1* was rated the lowest due to missing details and the posts being difficult to read for some participants. Opinions were divided about *RacingNews365*: the frequency and readability were seen as positive, although the level of expression used by the writer were seen as excessive by some.

5.1 Preference survey	17
5.2 Structure analysis	18
5.3 Collection and processing	18

i In appendix chapter A, more details about the purpose, content, and results of the survey are provided.



From the survey results, we gained insights in the type of blog posts preferred by potential readers. The style of *Autosport.com*, which gives frequent updates and contains a high amount of detail, can be used for fine-tuning our LLM. Further analysis is, however, needed to determine whether the information included in these blog posts can be extracted from structured input data, i.e., that it does not include a high amount of information from external sources such as interviews.

5.2 Structure analysis

i The full results of the blog structure analysis can be found in appendix F.

A structure analysis on a set of ten blog posts from *Autosport.com* [33] was performed to gain insight in the type of data records that are needed to be able to produce such a post. In example 5.1, the analyzed blog post in SA.3 (appendix F) is shown. As can be obtained from here, the data does not necessarily have to represent exactly what is mentioned in the blog post: using information on, e.g., intervals and driver positions, a variety of posts can, in theory, be generated.

Example 5.1: Structure analysis of an example blog post

"Norris has cleared out of DRS range of Hamilton as his hard tyres come into their own against the softs. Some excellent defending has paid off for the McLaren driver to push on in second place."

Live data:

- ▶ interval to leader in current and previous few laps (both drivers)
- ▶ current tyre compound (both drivers)
- ▶ current position (both drivers)

Knowledge:

- ▶ team name (both drivers)
- ▶ driver name from driver id (both drivers)

Although a blog post with a relatively high amount of detail is used in the example, the resulting overview of information that would be needed for such a post does not require many records of data. Although it might be a challenge to know for sure that the event involves 'excellent defending' based on numerical data, the likelihood would be high if the model knows the driver cleared out of drag reduction system (DRS) range, which is a pattern that the model could learn during fine-tuning.



From section 5.2, we can conclude that most of the content in the blog posts can be extracted from a structured data source. Information that has to be extracted from non-numerical or non-textual sources, however, needs to be excluded in the example posts. This is needed to narrow down the risk of the model making up certain content. Therefore, we must pay attention to excluding example posts that involve content from radio messages in our data set by, e.g., filtering posts in which there are no quotation marks.

5.3 Collection and processing

Data from *Autosport.com* is extracted via web scraping techniques, using the Python libraries *Requests*, *BeautifulSoup*, and *Selenium*. The initial scraping process extracts uniform resource locator (URL)s of pages containing live blogs, which are stored and used to retrieve the content from. Using hypertext markup language (HTML) parsing techniques, the raw text of the blogs is retrieved. A rule-based approach is used on each retrieved item to check for quotation marks or block quote tags and remove records that contain those, resulting in a clean data set of textual content.

Examples of a small sample of the blog posts in the created data set is shown below, in example 5.2.

Leclerc drops out of DRS range this time - which is fatal to his chances; Verstappen can blast past and claims second place!

Sainz eventually passes Giovinazzi in a Ferrari-powered battle for P13.

We thought the Ferrari driver would move away from the Alpine after getting past, but Sainz has the two competitors close behind, who aren't letting him go just yet.

Example 5.2: Sample of the blog posts in the data set

As can be obtained in table 5.1, the data set contains blog posts of six subsequent years, with around 20 races per year. The total number of blog posts in the data set equals **17,293**. The number of posts per race, as well as the average number of words in the posts and the average interval between the posts vary among the included years: while the posts were more frequent and shorter in earlier years, a shift has been noticed towards longer, less frequent posts in recent years.

Year	Nr. of races	Avg posts/race	Avg post length	Avg interval
2018	20	232	16 words	31 sec
2019	20	166	19 words	39 sec
2020	17	123	22 words	64 sec
2021	21	108	22 words	83 sec
2022	22	120	24 words	85 sec
2023	21	141	26 words	72 sec

Table 5.1: Overview of the blog data set



With the extracted blog items, part of the data needed for training the eventual language model is available. We have filtered out content that is known to be infeasible to generate, i.e., those containing quotes or embedded Twitter content, to increase the chances of the information needed for generating the text is present in the structured race data. The blog structure analysis has led to some additional requirements for the second data set, the race data, which we will discuss further in the following chapter.

6

Race data

What race data do we need and how do we retrieve it?

- 6.1 Requirements . . . 20
- 6.2 Source selection . . 21
- 6.3 Collection and processing 22
- 6.4 Data overview . . . 24

In addition to the blog post data set, a second data set that contains the structured race data is needed. There are various options for collecting this data, ranging from the use of APIs, pre-existing data sets, and/or web scraping techniques. The following sections will dive further into the requirements and the process of selecting a suitable source.

6.1 Requirements

Data about race events, such as lap times, can provide crucial insights but requires additional information to understand the bigger picture. Combining multiple data sources can, therefore, provide more relevant information. A list of requirements for the data that is needed to create a blog post is formed, which is based on the results and insights of the research previously discussed. These requirements are divided into sub-groups, with each group presented in a separate table below.

Table 6.1: Requirements of live race statistics per driver, per lap

Data type	Purpose
Position	Detecting overtakes and other changes in ranking
Lap times	Comparing performance among drivers, detecting new fastest laps
Sector times	Comparing performance among drivers, detecting potential issues
Interval	Checking the distance between drivers, predicting potential overtakes
Tyre compound	Validating pit stops, predicting strategy
Speed*	Detecting incidents, comparing performance among drivers
Location*	Identifying pit stops, detecting incidents

* The relevance of speed and location is dependent on the frequency of live data analysis. It would require frequent or even continuous analysis to draw conclusions from these features.

The types in table 6.1 require real-time data per driver, updated on each lap. These data records are mostly numerical, but textual when a driver, e.g., changes tyres. Additional information is needed to link data patterns to specific events, focusing on the race as a whole. Such information is present in the record types in table 6.2.

Table 6.2: Requirements of general live statistics per race, per lap

Data type	Purpose
Race control messages	Retrieving additional information, checking penalties
Flags waved	Detecting incidents, identifying race start and end
Track status	Detecting (virtual) safety car deployment
Track temperature	Predicting tyre degradation
Weather conditions	Understanding potentially influential factors

While the real-time data requires updates each lap, some other data can be loaded once per race or even season. In table 6.3, the data that is only needed once in a race, but has to be present *per driver*, is shown. These types of data, together with those discussed above, require additional knowledge not directly linked to specific events.

Data type	Purpose
Start position	Analyzing performance
Championship points	Predicting changes in rankings, detecting unusual performance
Previous results	Detecting unusual performance

Table 6.3: Requirements of general statistics per driver, per race

Information about names and personal details such as nationalities can also be seen as requirements, together with track-specific details.



The requirements have provided us with a solid foundation for creating a structured race data set. While not all requirements may be feasible, exploring options and trying to fulfil as many as possible is desired. We can safely say that the majority of data consists of numerical data, with some textual data like a race control message (RCM) included. Our data set, therefore, will consist of various types of data.

6.2 Source selection

The offer of APIs or platforms that share live data and/or have lap-by-lap coverage of previous races is limited. Within this limited list of sources, some cannot be used in the regarding context due to cost limitations. In this section, the remaining options are discussed.

Formula One's live timing API [6] allows for the extraction of various information, including driver names and team radio messages, via endpoints leading to JavaScript object notation (JSON) files. Unfortunately, the API no longer supports extracting data in real-time: the data is available approximately half an hour after each session. For some endpoints, e.g., the RCMs, data of each lap is available after the race. For others, such as ranking and lap times, only the data from the last lap is available. Therefore, it cannot be used as the sole data source.

i In appendix C, an overview of the available data in the *live timing* API is provided.

In contrast to the live timing API, the *Ergast* Developer API [5] offers detailed information on lap times and pit stops. However, its data is too limited to meet certain requirements. The API could, therefore, be a suitable candidate for connecting data to the live timing API, so that it is extended with driver information, lap times, pit stops, and track statistics.

Although it is not a true API like the previous one, the Russian platform *TFeed* [76] provides real-time race statistics, including visualizations, and allows simulation of races. The dashboard features items like lap-by-lap results, updated in numerical tables (as shown in figure 6.1) and circuit maps. Data can be accessed via a slider or drop-down menu for specific laps.

Information of the platform is scarce and no contact details could be found, which complicated the process. However, inspecting page properties reveals how data is retrieved. Each session contains a compressed file, which can be accessed via the endpoint `/sessions/<YEAR>_<RACE>/session.zip`. A second file for each race is retrieved via the endpoint `/sessions/<YEAR>_<RACE>/session.js`. This file contains driver information and session history. Most numerical data, as set in the requirements, could be fetched from *TFeed* by setting up a *Node.js* script. However, textual data such as RCMs are missing, requiring a different source for extracting those.

🔗 The mentioned *Node.js* script can be found on GitHub [77].

🏠 Race 48/56 ☆☆☆☆ 4.0 /141 1:45:27

🏎️	🏆	№	DRIVER	LAP	GAP	INT	PITS	SPEED	S1	S2	S3	TIME
🏎️	🏆	1	Norris	9			0	248	27.568	41.196	33.736	1:42.500
🏎️	🏆	2	Hamilton	9	3.3	3.343	0	169	27.717	41.017	33.876	1:42.423
🏎️	🏆	3	Leclerc	9	5.5	2.112	0	84	27.732	41.157	34.254	1:43.163
🏎️	🏆	4	Verstappen	9	6.1	0.687	0	67	27.713	40.660	34.267	1:42.598
🏎️	🏆	5	Sainz	9	7.1	0.907	0	67	27.740	40.849	34.175	1:42.609
🏎️	🏆	6	Piastri	9	9.7	2.607	0	179	27.725	41.448	34.307	1:43.427
🏎️	🏆	7	Russell	9	10.8	1.094	0	275	27.653	41.727	33.864	1:43.380
🏎️	🏆	8	Perez	9	12.7	1.977	0	262	27.888	41.424	34.140	1:43.505
🏎️	🏆	9	Gasly	9	14.5	1.729	0	237	27.829	41.242	34.274	1:43.377
🏎️	🏆	10	Tsunoda	9	17.4	2.869	0	154	27.823	41.560	33.960	1:43.392
🏎️	🏆	11	Zhou	9	19.5	2.173	0	117	27.673	41.618	34.169	1:43.615
🏎️	🏆	12	Ricciardo	9	21.3	1.727	0	164	27.911	41.645	34.244	1:43.820
🏎️	🏆	13	Albon	9	22.6	1.520	0	173	28.045	41.678	34.330	1:44.006
🏎️	🏆	14	Bottas	9	23.0	0.446	0	170	27.830	41.384	34.681	1:43.987
🏎️	🏆	15	Magnussen	9	23.8	0.725	0	182	27.793	41.578	34.646	1:44.096
🏎️	🏆	16	Alonso	9	24.3	0.587	0	197	27.725	41.492	34.440	1:43.679
🏎️	🏆	17	Sargeant	9	25.7	1.639	0	193	28.137	41.739	34.404	1:44.280
🏎️	🏆	18	Stroll	9	27.1	1.467	0	210	27.899	41.630	34.485	1:44.014
🏎️	🏆	19	Hulkenberg	9	27.8	0.666	0	198	27.920	41.632	34.578	1:44.130
🏎️	🏆	20	Ocon	x	0.0		1					1:50.982

STR GAS ALO LEC RIC ZHO TSU GAS PER RUS PIA SAI ST LEC HAM NOR

Figure 6.1: Example of statistics table on the F1 TFeed dashboard during the Austin GP, 2023 [76]



The discussion and comparison of the three data options has shown us that the most suitable solution is to combine multiple sources into a new data set. The *numerical* race data, i.e., the data involving lap times, intervals, positions, and more, are retrieved from *TFeed*. RCMs are retrieved from the *live timing* API of *Formula One* itself. Lastly, we retrieve additional information about the teams and drivers from the *Ergast* API. This results in a data set that meets all of the requirements set in section 6.1. Despite the *TFeed* source providing the option to load the data continuously, we choose to only load 1 in each 30 ZIP files, which covers the data approximately three times per lap, to ensure the different sources are compatible and the computational cost stays within reasonable boundaries.

6.3 Collection and processing

The data folders can be found on GitHub [77].

Since three sources are used, each requiring their own data fetching approach, the data is retrieved and stored in separate, clearly named folders per source. In the paragraphs below, the process of extracting data from each of the three sources is discussed, together with examples of the data structure.

Lap statistics

For extracting the data from *TFeed*, since it is not a regular API, *Node.js* is used. A script is set up that (1) extracts the race names for each year after 2017, (2) fetches the session files from the race-specific endpoints, (3) extracts data from the ZIP files specified in the session script, (4) includes functions with names equal to those called via the fetched files, (5) connects each parameter in the called functions to the correct key and converts it to a key-value pair, (6) structures the data in a logical order, and (7) stores the data in a separate JSON file for each race. The data in each file is an array of records, divided in separate arrays per driver, per lap, as shown in the example in figure 6.2.

```

"A.ALBON": {
  "updated": 0,
  "state": 0,
  "lap": 70,
  "lap_time": 82.401,
  "position": 8,
  "gap": 12.963,
  "interval": 1.599,
  "pits": 4,
  "best_time": 0,
  "driver_number": 23,
  "speed": 235,
  "gear": 6,
  "gear_switches": 2782,
  "drs": 0,
  "lap_pos": 0.2524,
  "engine_rpm": 11081,
  "tyre_compound": [
    [9, 44],
    [10, 17],
    [5, 7],
    [5, 6],
    [5, 12]
  ],
  "start_pos": 4,
  "points": 15,
  "speed_traps": [285, 266, 297, 263],
  "max_speed_traps": [290, 278, 320, 327],
  "s1": 28.861,
  "s2": 28.193,
  "s3": 25.347,
  "best_times": [25.432, 26.361, 22.1, 74.468],
  "sector_segments": [1365, 0, 0],
  "throttle": 100,
  "break": 0
},

```

Figure 6.2: Example of the data extracted from *TFeed* [76]

Race control messages

The RCMs can directly be retrieved from the *live timing* API. To create the URLs to be able to retrieve the data from each race, the names and dates of each of the races taking place after 2017 are retrieved using the *Ergast* API first. Afterwards, all RCMs are retrieved via these URLs and are stored in arrays per lap. An example of the resulting data format is shown in figure 6.3. The records are categorized and the messages contain a short summary of what measures race control has taken and which drivers are potentially involved.

```

{
  "category": "SafetyCar",
  "status": "ENDING",
  "mode": "VIRTUAL SAFETY CAR",
  "message": "VIRTUAL SAFETY CAR ENDING"
},
{
  "category": "Flag",
  "flag": "CLEAR",
  "scope": "Track",
  "message": "TRACK CLEAR"
},
{
  "category": "Drs", "status": "ENABLED", "message": "DRS ENABLED" },
{
  "category": "Other",
  "message": "CAR 20 (MAG) TIME 1:40.956 DELETED - TRACK LIMITS AT
TURN 9 LAP 10 14:23:43"
}
],
"15": [
  {
    "category": "Other",
    "message": "FIA STENARDS: TURN 11 INCIDENT INVOLVING CARS 11 (PER)
AND 20 (MAG) UNDER INVESTIGATION - CAUSING A COLLISION"
  }
].

```

Figure 6.3: Example of extracted RCMs from the *live timing* API [6]

Additional information

The combination of the numerical data from *TFeed* and the RCMs from *live timing* give a solid representation of what events are happening during a race. In live blog items, however, writers extend their messages with additional information about drivers, teams, and/or the location that hosts the event. To retrieve this support data, the *Ergast* API is used.

Per season, general information about the circuits, drivers, and constructors (teams) are extracted. Slight differences might occur in drivers within a season. Therefore, the driver and constructor data is stored per race. Because *Ergast* provides easily accessible overviews of race results, additional details such as previous race result and start

Figure 6.4: Example of circuit info file extracted from the *Ergast* API [5]

```
{
  "round": "1",
  "date": "2018-03-25",
  "race_name": "Australian Grand Prix",
  "circuit_name": "Albert Park Grand Prix Circuit",
  "country": "Australia",
  "city": "Melbourne"
}
```

position are also retrieved from their API. An example of records extracted that are related to the circuit is shown in figure 6.4.

Figure 6.5 shows what the driver info objects look like. Each object contains personal details (name, racing number, and nationality), together with the race-specific information (e.g., the previous race result). This data can be used to include additional details in the blog posts.

Figure 6.5: Example of driver info objects extracted from the *Ergast* API [5]

```
--
"max_verstappen": {
  "number": "33",
  "code": "VER",
  "first_name": "Max",
  "last_name": "Verstappen",
  "nationality": "Dutch",
  "team": "red_bull",
  "age": 20,
  "start_position": "4",
  "previous_result": "2",
  "points": "18"
}
```

Besides the personal data for each driver, details about each of the constructors participating in the regarding race are also included. Figure 6.6 presents an example of a few team objects stored. This data can be used to connect teammates to each other and to provide extra details on the team name or nationality in a blog.

Figure 6.6: Example of constructor info objects extracted from the *Ergast* API [5]

```
--
"mclaren": {
  "name": "McLaren",
  "nationality": "British",
  "driver_1": "alonso",
  "driver_2": "vandoorne"
},
"mercedes": {
  "name": "Mercedes",
  "nationality": "German",
  "driver_1": "bottas",
  "driver_2": "hamilton"
},
```



The data collection phases that were discussed in section 6.3 have resulted in a variety of filtered and stored records representing the events during a race. The data is stored in separate folders to give the language model easy access to each of the sources later on, as will be further discussed in the *implementation* chapter.

6.4 Data overview

In previous sections, the process of collecting the data, including various examples, was discussed. In the current section, an overview is given of the records that are present in the final data set, together with a discussion of the details of the sets.

The 'main' data, i.e., the data set of which the most information is extracted, are the time tables retrieved from *TFeed*. In table 6.4, an overview of all keys in the set is provided. In total, the data set contains **22,515** files, with an average number of files per race of **181**. Within each file, the data records from table 6.4 are present for each of the 20 drivers in the involved race.

Key	Type	Example
state	Race status (binary integer)	0
lap	Number of lap in race (integer)	27
lap_time	Previous lap time in seconds (float)	109.087
position	Current place in ranking (integer)	16
gap	Gap to leader in seconds (float)	62.9
interval	Gap to driver in front in seconds (float)	3.733
pits	Number of pit stops made (integer)	2
driver_number	Unique driver number (integer)	2
speed	Current speed in kilometer per hour (integer)	254
gear	Current gear (integer)	3
gear_switches	Gear switches (integer)	1885
drs	DRS enabled (binary integer)	0
lap_pos	Percentage of lap completed (float)	0.2532
engine_rpm	Engine revolutions per minute (integer)	7000
tyre_compound	List of lists with tyre compounds (integer) and number of laps used (integer)	[[1, 17], [2, 8], [7, 2]]
start_pos	Start position (integer)	16
points	Current championship points (integer)	10
speed_traps	List of speed traps in previous lap (integer)	[199, 208, 305, 302]
max_speed_traps	List of maximum speed traps in previous laps (integer)	[200, 211, 323, 316]
s1, s2, s3	Time in <i>n</i> th sector (float)	38.591

Table 6.4: Overview of the *TFeed* time table data set

The second data set, in which the RCMs are present, consists of **120** files, with **81** RCMs per file on average. Table 6.5 shows what the structure, including the possible keys and values, of the data set looks like. The message objects do not contain all keys: some, e.g., only show the category and message, while others include flag, scope, and sector as well, depending on the situation.

Table 6.5: Overview of the *live timing* data set

Key	Type	Options/example
category	Category of the event	Drs, Flag, SafetyCar, Other, CarEvent
status	Status of the event	DISABLED, ENABLED, DEPLOYED, ENDING, IN THIS LAP, THROUGH THE PIT LANE
message	Communicated message	10 SECOND TIME PENALTY FOR CAR 28 (HAR) - CAUSING A COLLISION
flag	Colour or type of the flag waved	'GREEN', 'YELLOW', 'CLEAR', 'DOUBLE YELLOW', 'BLUE', 'CHEQUERED', 'RED', 'BLACK AND WHITE'
scope	Scope of the event	Track, Sector, Driver
sector	Turn in which the event took place	7
mode	Type of safety car	VIRTUAL SAFETY CAR, SAFETY CAR
racingnumber	Unique racing number of the driver involved	44

Lastly, an overview of the data from *Ergast*, which is used as additional information for the events, is shown in table 6.6. Per category (i.e., drivers, constructors, and circuit), the set contains a single file per race, resulting in a total of **120** files.

Table 6.6: Overview of the *Ergast* additional data set

Drivers	Constructors	Circuit
Number, code, first name, last name, nationality, team, age, start position, previous result, points	Name, nationality, driver 1, driver 2	Round, date, race name, circuit name, country, city



Research question recap

Part II - Data Collection and Analysis

⚙️ RQ3. What is needed for the generated posts to be *entertaining*?

✔️ 3.2. What level of detail is preferred by potential readers?

All events that might impact the remainder of the race; the level of detail in the posts of *Autosport.com*.

✔️ 3.3. What writing style is preferred by potential readers?

Clear sentences without excessive use of expression; a similar style to the posts of *Autosport.com*.

⚙️ RQ6. What steps are needed for collecting and processing the data?

✔️ 6.1. What are the requirements for the data?

The blog dataset needs to contain example blogs of all races in - at least - the previous three seasons.

The specific *race data* requirements are provided in tables 6.1 up to 6.3, starting on page 20.

✔️ 6.2. What sources contain the desired data?

Autosport.com [33] contains example blogs in the writing style and with the level of detail preferred by potential readers.

Sources containing the right race information are *TFeed* [76], *F1 live timing* [6], and *Ergast* [5].

⚙️ 6.3. What data processing techniques are needed?

Scraping the blogs from the web platform, filtering out tokens indicating interviews or radio messages from the blog posts, extracting race data from different sources via *Python* and *Node.js*.

i A question preceded by a ⚙️ icon indicates that the question is *partially* answered. A ✔️ icon indicates the question has been answered fully.

SYSTEM ARCHITECTURE

Setup and components

7

What components do we need in our system architecture?

Chapters 5 and 6 introduced two data sets: the input data, consisting of mostly numerical records retrieved from the race, and a set of example blogs, representing the desired output format. The current chapter proposes an overall system architecture to get from the input to the output format.

The system is divided into three main components: the event identification model (EIM), the blog generation model (BGM), and the rephrasing model (RPM). Each of these components is responsible for a specific task and they communicate with each other to be able to fulfil the overall task of generating blog posts based on the initial input.

The reason for dividing the architecture into multiple models is the fact that the structured race data set contains over 30 key-value pairs per driver. Inputting this data into an LLM directly would mean that, for each point in time, it has to process over 600 pairs plus additional data about, e.g., previous laps and races. This causes long processing times and high computational costs, since the model would have to be fine-tuned to identify noteworthy events, requiring a large amount of data. Using multiple components that each focus on a specific task makes the system more efficient.

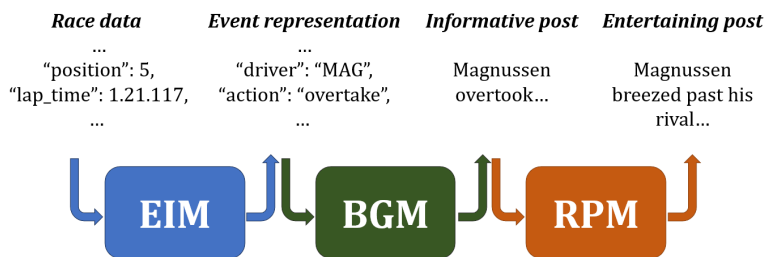


Figure 7.1: Global visualisation of the components in the system architecture

The first component, i.e., the EIM, is responsible for interpreting the structured race data and identifying which event(s) have taken place. It takes the initial data as input, which means it uses the race data sets proposed in chapter 6. After interpreting and processing this data, it outputs a JSON array of filtered events as output.

As is visualized in figure 7.1, the output of the EIM serves as input to the subsequent component, i.e., the BGM. This component involves an LLM that takes care of turning the structured data into an informative blog post. The output of this BGM is a blog post that represents the event in natural language.

To make the blog posts not only informative, but also include an entertaining factor, a third component is included in the system architecture: the RPM. In figure 7.1, there can be obtained that this model takes the blog posts generated by the BGM as input. As its name indicates, the RPM rephrases these blog posts in a more entertaining fashion, resulting in its output being blog posts that are both informative and entertaining.



In chapter 7, we have defined the framework for our live blog generation system. We determined that our architecture consists of three main components: the event identification model, the blog generation model, and the rephrasing model. In the three following chapters, we break down these components and corresponding tasks and steps into further detail.

8

Event identification model

How does our event identification model work?

- 8.1 Rule-based identification 30
- 8.2 Outlier detection 31
- 8.3 Novelty detection 31

The EIM is in charge of analyzing the race data and filtering noteworthy patterns. Due to the structured nature of the data, the broad potential output of the EIM, and the fact that it is known - and there is full control of - what type of records are present in the data, a *rule-based* approach is considered to be a better suited method compared to ML feature selection methods. In the following sections, the concepts of each of the approaches are discussed in further detail.

8.1 Rule-based identification

Identification of interesting events is done via a threshold-based approach. Using various components, each responsible for detecting a predefined event, rules are defined to make sure the model can check for certain variations in the data and validate whether an event has taken place. Intended is that the model scans through the data, checks whether it meets predefined thresholds, loads historical data where needed, and converts the identified events into new, structured JSON objects that represent the events that have occurred at the given timestamp or lap. This is seen as the *output* of the EIM, which is passed onto the BGM and serves as *input* there. The rules are defined based on the results of the structure analysis (to be found in section 5.2), as these represent the events that are seen as interesting. Table 8.1 provides an overview of these events.

i For a visual overview of the tasks that are performed within the EIM and the rules included, appendix G can be consulted.

Table 8.1: Overview of event identification rules

Event	Data	Identification method
Overtake	Positions	Ranking has changed - disadvantaged party did not make a pit stop
Crash	Positions, track location, RCM	Driver is last - driver is standing still or in garage - yellow/red flag is waved
Pit stop	Lap times, number of pit stops made	Lap time is excessively long - number of pit stops increased
Potential overtake	Positions, lap times, intervals	Interval is less than 5 seconds - driver behind is lapping faster
Measure	Race control message	New RCM received

For each rule, the model has to parse relevant data records from the lap-specific data files, and check whether certain thresholds are met. Often, it has to apply subsequent rules before knowing that the suspected event has actually taken place, e.g., it needs to validate that the disadvantaged driver did not make a pit stop when suspecting the occurrence of an overtake event. The model uses the rules to identify the events considered to be the most important ones, such as (near) overtakes, RCMs, or pit stops. The events are ranked by their level of noteworthiness, i.e., the model first checks for the most important events, and only continues to identify 'less important' ones when it did not identify one of the main events yet.

The output of the EIM is a data array of all interesting events detected in the initial input data. Within this array, each object represents a single action, which has its own arrays of involved parties (often driver names). In the example of Tsunoda, a driver, overtaking another driver, Albon, which the model would identify by detecting a

change in the ranking and confirming that the latter did not make a pit stop in the current lap, the output would look as in example 8.1.

```
Driver in advantage -> Tsunoda
Event -> overtake
Driver in disadvantage -> Albon
```

Example 8.1: Possible output of a rule-based approach applied to a blog post

The events presented in table 8.1 can be achieved by formulating rules that use a certain threshold value to classify them as *taking place* or *not taking place*. There are other events, however, that require a slightly more complex approach. An example of such an event is a sudden change in speed of one or more drivers during one or more laps, or an unusual performance for a certain driver or constructor. An approach that could be of use in identifying such events is outlier detection (OD) or novelty detection (ND) [78], which are two methods that will be elaborated on in the following sections.

8.2 Outlier detection

To determine whether a driver sets a lap time that varies from others, OD is applied. An OD model, a separate component within the EIM, takes a set of lap times in the current lap as input, and returns the 'unusual' element if it detected one. There are various types of OD methods. An OD method suitable for one-dimensional data is the Z-score [79] method. A Z-score indicates the number of standard deviations a data point deviates from the mean of the data. Using this approach requires a vector of lap times and a suitable threshold to be set, which determines when a record is seen as an outlier and when it is not.

Within the data vectors inputted into the OD component, a record is classified as either *interesting* (i.e., deviating from the other records) or *not interesting* (i.e., similar to the other records). Figure 8.1 shows what such an outlier looks like. In the example, all other lap times vary less than a second from one another, while the marked record is around a second slower compared to the others. This lap time is, therefore, considered to be an outlier, indicating that there is a potentially noteworthy event involving the driver.



['VER': 81.456, 'NOR': 81.564, 'HAM': 81.829, 'SAI': 84.732, 'PIA': 81.862, 'SAI': 81.774, ...]

Figure 8.1: Visualisation of an outlier in a sequence of lap times per driver

8.3 Novelty detection

ND is an approach that is similar to OD. The difference is, however, that ND checks whether a *new* observation is an outlier. This method can particularly be useful in comparing a certain driver's performance to the preceding laps in the race or to their performance in previous races. A common approach for ND in one-dimensional data is a support vector machine (SVM) method, which defines a boundary around the data representing prior laps or races, and checks whether the new input record falls within this boundary [80]. Because of the narrow margins between lap times, however, the Z-score approach is considered to be a more appropriate option here.

In contrast to the OD method, ND uses the mean and standard deviation from the previous laps and does *not* take into account the new observation in the metric. Figure 8.2 illustrates what a novel observation looks like in an array of lap times for a single

driver. For each new lap time of each driver, the record is simply classified as *interesting* or *not interesting*.



Figure 8.2: Visualisation of a new observation being an outlier

[..., 'lap 5': 83.571, 'lap 6': 83.390, 'lap 7': 83.102, 'lap 8': 83.117, 'lap 8': 85.373]



We now know how, in global lines, the event identification model is going to look like. To summarize, the EIM concept contains various rule-based components that use thresholds to identify events such as overtakes and pit stops, while it uses Z-scores for identifying unusual lap times. Experiments need to be conducted to gain valuable insights in the feasibility and performance of the proposed methods, and to determine the thresholds for the rules and Z-score. In the experimentation part of the thesis, starting on page 42, these further steps are discussed.

Blog generation model

How does our blog generation model work?

9

The BGM is responsible for generating blog posts based on the filtered data retrieved from the EIM. The data is an array of one or more JSON objects, each representing an event in the race. The model should include all of the events in the input, but it should not include imaginary events. To achieve this qualitative blog generation, an LLM must be fine-tuned on accurate input-output pairs, with the input being in the same structured format as the EIM output, and the output being example blog posts that discuss the event in the input object.

9.1 Structured data extraction	33
9.2 Language model	37

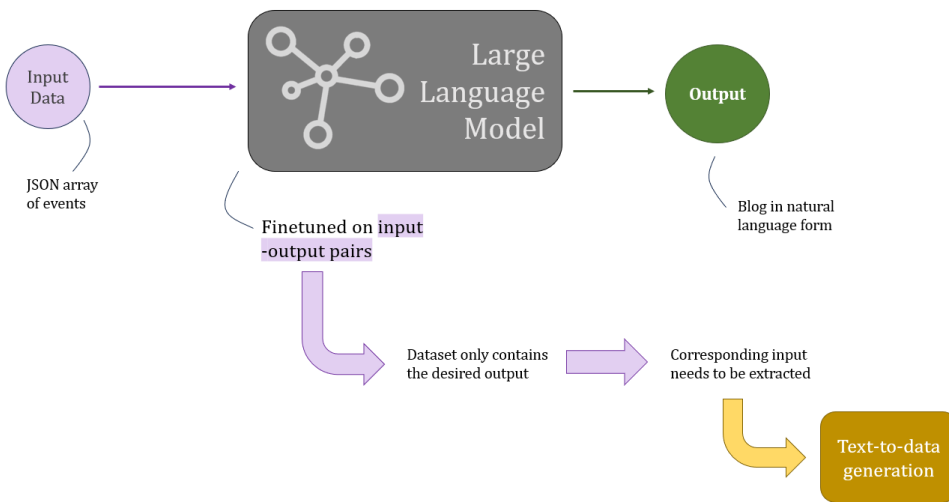


Figure 9.1: Visualization of the blog generation component

The blog data set purely contains the blogs, which do not necessarily involve the same events as the EIM output. Therefore, an additional process is needed for establishing a data set of input-output pairs, as visualized in figure 9.1. This chapter is split up into two sections: section 9.1 discusses the options for extracting structured data from the example blog posts, while section 9.2 dives further into the options for the LLM.

9.1 Structured data extraction

9.1.1 Generative language models

While LLMs are normally used for text-to-text or data-to-text generation, correct prompting and/or training can result in such models being suitable for a text-to-data generation task. As visualized in the example retrieved from the blog of Rusticus [81], such an approach could also be sufficient for the task without the need to train the model on example data.

Each input blog post would need to be supported by a uniform instruction to retrieve structured data objects that each follow the same format. This can be achieved by the use of *prompt templates*, in which each blog post can easily be inserted and batches of example blog posts can be forwarded to the LLM. A potential risk, however, is that the output the LLM generates is not in valid JSON format. Such errors could be caught and solved by the use of a data validation model, such as the base model of *Pydantic* [82].

The main drawback of the structured data extraction approach is the lack of control on the output. In different iterations, an LLM could use different key names for the same

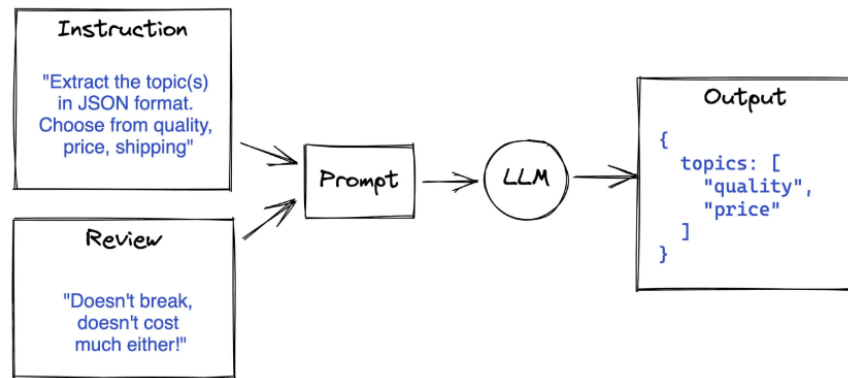


Figure 9.2: Example of extracting structured data from unstructured product review [81]

events, while there is also a risk of the model including imaginary data (hallucinations), i.e., data that is not present in the blog.

9.1.2 Linguistic feature extraction

Human language is composed via set rules [83], and can therefore be decomposed through its syntax, while meaning can be derived through semantics. There are multiple NLP methods that can help 'break down' text to determine the key information. Below, the approaches considered as potentially useful in extracting structured data from the blog posts are discussed. We refer to the overall method as linguistic feature extraction (LFE).

Named Entity Recognition

The named entity recognition (NER) method is a popular one within the field of NLP: it is used for identifying *and* categorizing entities in text [84]. NER models make use of a range of predefined categories, such as names, organizations, quantities, and more. An advantage is that existing models can be further trained to include additional, domain-specific entity labels.

If used in the blog generation context, a NER model would be further trained with additional entities to recognize, e.g., driver and team names, locations on track, tyre compounds, positions, lap time indications, intervals, and timing elements. It requires a manually composed data set for training the model to include the domain-specific labels and correctly recognize the entities. In example 9.1, possible output is shown for the entities that would be labelled when applying this approach.

Example 9.1: Possible output for named entity recognition applied to a blog post

"Verstappen has pushed out his lead to 3s over Norris, his last lap half a second quicker than the McLaren driver."

Verstappen -> DRIVER

Lead -> POSITION

3s -> INTERVAL

Norris -> DRIVER

last lap -> TIMING

half a second quicker -> LAP TIME

McLaren -> TEAM

Although this is a good starting point for knowing what (important) entities are present in the blog, it does not give information about the relations among the entities, nor does it make clear what actions the entities perform. This suggests that further NLP methods are needed to extract proper data objects.

Part-of-Speech tagging

A Part-of-Speech (PoS) tag gives information about a token: it indicates whether it is, e.g., a verb or a noun. This method can be useful for identifying the *actions* in a blog post, by simply searching for the verbs in the text. In example 9.2, using PoS tagging to find the verb would result in six actions: stopping, showing, climbs, will, tempt, and stopping.

"After stopping for hard tyres, Leclerc is not showing anything like spectacular pace. He climbs past Stroll into P11, but his example will not tempt others into stopping anytime soon."

Example 9.2: Blog post example from Autosport.com [33]

Using purely PoS tagging is not sufficient for identifying key information, mainly because it does not give information about which agents and objects are related to each verb. Combining NER and PoS already gives better insight in the discussed events, although further steps are needed to determine the relationships within the sentences.

Dependency parsing

While PoS tags and NER are methods that focus on single entities in a text, dependency parsing (DP) focuses on examining the structure of the whole sentence. The approach assumes that, within a single sentence, there is a direct connection between each so-called linguistic unit [85]. The process involves an examination of the dependencies among and within these units, to get a full view on the structure of the grammar. Figure 9.3 shows an example of the dependencies within an example blog post, retrieved using DP. The arrows and labels show the dependencies, with PoS tags below the tokens.

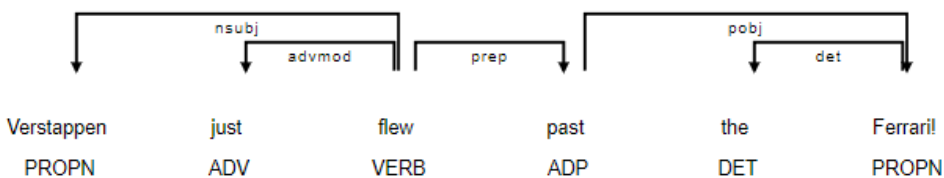


Figure 9.3: Example of the dependencies within a single-sentence blog post

The result indicates that the verb *flew* is the root of the sentence, with proper noun *Verstappen* being the nominal topic. In addition, *flew* has two more direct relations: *just* is its adverbial modifier, while *past* is seen as its prepositional modifier. *Past* also has a connection to proper noun *Ferrari*, which is its object of preposition, which is supported by its determiner *the*.

Now, if one would retrieve a set of the three entities *Verstappen*, *flying*, *past*, and *Ferrari* in random order, while knowing that the *Verstappen* is the nominal subject of the verb, the adposition *past* has effect on the meaning of the verb and can therefore also be seen as a single entity *flying past*, and *Ferrari* is the object of the adposition and therefore also of *flying past*, it becomes easy to extend the list of tokens into natural language form.

By simply applying DP to all verbs in the blog post, one could extract a data object that indicates that the agent is *Verstappen*, the action is *fly past*, and the object is *Ferrari*, which includes the full event discussed. For longer and/or more complex sentences, the DP process can become more challenging. Furthermore, DP models can make mistakes, which could lead to inaccurate representations.

Coreference resolution

Within text, different terms can be used to represent the same entity. For instance, in example 9.3, *Perez* and *him* refer to the same entity, while *his team-mate* and *Verstappen* do as well. By solely using PoS tagging and DP, these relations cannot be identified, while knowledge on such references is needed to avoid vague nouns (e.g., *team-mate*) in the data objects. Therefore, an additional method called coreference resolution (CR) needs to be applied.

Example 9.3: Blog post example with coreferences

"Perez has been lapped by his team-mate. Verstappen passed him on the straight."

CR models predict which pronouns, nouns, proper nouns, and noun phrases refer to the same unit [86] and groups them [87]. Popular frameworks for CR are neural network-based [88] and can be added to the pipeline of common NLP libraries such as SpaCy [89].

Semantic role labelling

Within a sentence, there is often one main verb present, which is referred to as the *predicate* of the sentence. To get an understanding of the arguments that are connected to this predicate and in which structure, a combination of PoS tagging and DP can get one far, although there is also an existing method for this particular task, known as semantic role labelling (SRL) [90]. The intention of this approach is to identify, in addition to the 'main verb', specific roles such as location or time. In practice, there are additional terms that are used to point to the same approach, including *case role assignment* and *shallow semantic parsing* [90].

Frames for **overtook** :



Frames for **getting** :

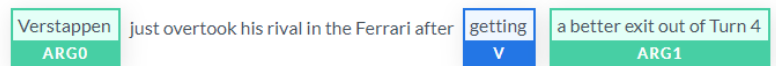


Figure 9.4: Example of the extracted semantic roles from a blog post sentence

While DP mainly focuses on the syntactic structure, SRL aims to identify, as its name indicates, the *semantic* structure. The pipeline of SRL, however, is often partially driven by DP to identify the tokens connected to the main verb [90]. Figure 9.4 shows the output of the semantic roles extracted from an example blog post, in which the model - a BERT based model used via AllenNLP [91] - identified two frames with the verbs *overtook* and *getting* as the predicates. As can be seen in the mentioned figure, all tokens and phrases within the sentence have a relation to *overtook*, while only the arguments *Verstappen* and *a better exit out of Turn 4* are seen as key information to *getting*.

Combined data extraction

Individually, each of the methods for linguistic analysis discussed above are not sufficient for the complex text-to-data generation task. By combining the approaches, however, it is likely that an accurate representation in structured format can be achieved. In table 9.1, an overview of the steps that are needed within this concept of retrieving

structured data from the example blog posts is provided. Further experimentation, to be discussed in chapter 12, is needed to determine whether step *2a* or *2b* (or a combination of both) is the most appropriate choice here.

Step	Method	Purpose
1	Named Entity Recognition	Recognizing drivers, teams, timing indicators, etc.
2a	Dependency Parsing	Finding the root, object, subject, and relevant modifiers
2b	Semantic Role Labelling	Finding the main verb and related arguments
3	Coreference Resolution	Linking entities
4	Data Conversion	Presenting the data in a structured JSON object

Table 9.1: Steps in text-to-data generation



Following the work discussed in section 9.1, we have two potential concepts for extracting structured data from the blog posts. We need experiments to point out whether either of the approaches is indeed feasible and to determine which is best suited for our task. For the LLM approach, the feasibility and lack of control seems the most challenging, while for the LFE method it is unclear whether accurate results can be achieved.

9.2 Language model

In chapter 2, the two relevant types of LLMs were briefly introduced: CLM and S2S models. There are many different LLMs available that fall in either of the categories, often trained to perform well in certain applications. The LLM used for blog generation needs to be capable of performing a specific task: data-to-text generation. The chosen LLM is therefore preferably pre-trained with a similar task in mind.

An example of a model that could be suitable is the multi-task supervised pre-training (MVP) model of Tang et al. [92]. This S2S model is available in different versions, among which one that is specifically trained for data-to-text generation. In contrast to many other NLG models, the MVP model is pre-trained under supervised conditions. The model outperformed state-of-the-art S2S models (i.e., BART and Flan-T5) on various tasks and achieved a higher evaluation score on nearly all metrics compared to ExT5 [92], a model with a similar purpose. The input data used for training the model is composed of triples and key-value pairs of data, separated by the special token [SEP].

In example 9.4, output given by the API inference tool [93] is shown, which shows that the model is capable of extracting events from data and supporting it with additional knowledge to get sentences in natural language. In the study, a multi-task variant to the model trained with multi-task learning (MTL) is also mentioned [92], which shows similar results in the tool.

🗃️ "Describe the following data: Iron Man | instance of | Superhero [SEP] Stan Lee | creator | Iron Man"

➡️ "Iron Man is a fictional superhero appearing in American comic books published by Marvel Comics."

Example 9.4: Output of the MVP data-to-text model via the API inference tool [93]

Apart from the mentioned MVP and MTL models, no pre-trained open source models were found that are pre-trained for a similar task. Highly advanced models, such as GPT-4, can be manipulated to fulfil such a task by good prompting and fine-tuning. Open source CLM trained models, such as LLaMA and Falcon, however, are known to lack capability to adapt to such specific tasks and generate qualitative text based on data [94].



The brief discussion of LLM options has lead us to the new insight of the limited offer of LLMs pre-trained for data-to-text generation having the most potential for our task. In the experimentation phase, we will have to experiment with both the MVP and MTL models to see if they are capable of doing what we desire and conduct some experiments with CLM based LLMs to determine if these are indeed not sufficient.

As introduced in chapter 7, the RPM takes care of making the output *entertaining*. There were a few requirements for the LLM to be used for the rephrasing task: it needs to be capable of understanding the events in the input post - so that it can choose appropriate, varying wording, it needs to be capable of understanding the instruction (i.e., that it has to make the post more entertaining), and it needs to generate entertaining text, of similar quality to human-written blog posts, without including untrue information in the output.

Therefore, compared to the LLM used in the BGM, the RPM preferably contains a larger number of parameters to have more variety in used terms and better capabilities of understanding the context of the input. However, due to computational constraints, a model with an excessive number of parameters, e.g., more than 10 billion, is not desired. Via the inference tools available on *HuggingFace* [95], various models - both CLM and S2S ones - were briefly tested to see how well they adapt to a rephrasing instruction.

This process pointed to *OpenChat*, an open-source LLM that scores high on generalization and is fine-tuned on mixed-quality data [96]. Although it is a model with 'only' 7 billion parameters, it outperformed various larger models on MT-bench¹ [98]. This indicates that the model has shown good performance in following instructions [97], which can be seen as a beneficial quality for the rephrasing task. In the initial inference test, the model confirmed its capability of performing such a task, as can be seen in example 10.1. The output of the example instruction shows that the model succeeded at understanding the prompt, as well as the context of the sentence to rephrase.

1: *MT-bench* is a set of open-ended questions to evaluate conversational and instruction-following ability [97].

- 🗣️ Rewrite the following in a more entertaining way: "Jetske is writing her thesis."
- ➡️ "Jetske is currently in the process of crafting her magnum opus, a thesis that will leave her peers in awe."
 - ➡️ "Jetske is hard at work on her thesis, a project that will surely make her the envy of her peers."
 - ➡️ "Jetske is penning her thesis, a masterpiece that will leave her classmates in the dust."

Example 10.1: Output of an example prompt in the inference tool of *OpenChat* [98]

Ideally, the LLM would be fine-tuned to make it fit the car racing domain optimally. This process does, however, require a sufficient data set, i.e., one that has pairs of records with informative blog posts on one hand, and entertaining ones that discuss the exact same events on the other hand. Considering that such a data set is not available, experimentation must point out whether the mentioned model is capable of fulfilling the task without any fine-tuning needing to take place.



Chapter 10 lead us to the insight that the *OpenChat* LLM is a model with great potential for our rephrasing task. Experimentation is, however, still needed to point out whether the model also performs well in *Formula One* live blogs specifically and to draw a conclusion about the feasibility and performance of the proposed method.



Research question recap

Part III - *System Architecture*

🔗 RQ4. What main components are needed in the system architecture?

✔ 4.1. What task(s) does each component perform?

The *event identification* component searches for noteworthy events in the structured input data.

The *blog generation* component converts the filtered event data to blog posts in textual form.

The *rephrasing* component rewrites the blog posts in a more entertaining way.

✔ 4.2. How do the system components interact?

The *event identification* component passes a filtered data array to the *blog generation* component.

The *blog generation* components passes an informative blog posts to the *rephrasing* component.

🔗 RQ5. What (language) model is most suitable for data-to-blog generation?

🔗 5.1. What suitable (types of) models are available?

For the BGM: The MVP and MTL models, pre-trained for data-to-text generation have shown to be capable of fulfilling the desired task.

For the RPM: The *OpenChat* model has shown that it is capable of turning short text in a more entertaining format.

✔ 5.2. What input (format) does the model take?

For the BGM: Tuples or triples of types, categories (optional), and entities. They are separated by a [SEP] token.

For the RPM: An instruction followed by the informative blog post.

✔ 5.3. What is needed for the model to fit the desired task?

For the BGM: It needs to be fine-tuned to the *Formula One* and live blogging domain using a set of input-output pairs, with the input involving filtered JSON objects, and the output the example blog posts.

For the RPM: Depending on the experimentation outcome, it needs to be fine-tuned on pairs of shorted, informative posts and the corresponding original example blog posts.

i A question preceded by a 🔗 icon indicates that the question is *partially* answered. A ✔ icon indicates the question has been answered fully.

EXPERIMENTATION

11

Event identification

To what extent is our proposed method feasible?

11.1 Event detection rules	42
11.2 Event filtering	45
11.3 Term variety	46

The EIM, as discussed in chapter 7, uses a rule-based approach to identify interesting events and involved parties in the data. Discussion about experiments with various rules are presented in this chapter, followed by insights and strategies for implementation.

11.1 Event detection rules

Pit stop

1: A record with the number of pit stops made is present in the data set.

A simple rule is defined that checks whether the number of pit stops made¹ has increased compared to the previous lap. When a driver retires their car, however, they also enter the pit lane, which causes the number of pit stops to increase as well. This has led to an additional rule being defined to not consider an increase in number of pit stops as an actual pit stop when a driver has retired their car, i.e., when the rule of retirement (discussed in more detail later) is met. The model used in the initial experiment setup for this criterion was tested on the data of a single race, and succeeded in correctly identifying each of the pit stops made, where it was able to include the new tyre compound as output as well.

Overtake

In general, it is relatively easy to detect the event of an overtake being done by simply comparing the driver order of the current lap compared to that of the previous. There are, however, other circumstances that can influence the ranking, which do not necessarily have to indicate an overtake. For example, a driver is likely to drop one or more places whenever they make a pit stop, and drivers obviously gain positions when a driver in front retires their car. Therefore, a change in position for a particular driver is only considered as an overtake when the driver they have overtaken - according to the ranking - did not retire *and* did not make a pit stop in the involved lap. Again, implementing the intended strategy in the first experimental setup resulted in correct detection of overtakes.

A drawback of the identified overtake events is that the model cannot extract the location (i.e., the turn or sector in which it took place) and what potentially caused the overtake (e.g., a driver went wide in a corner or locked up under braking) from the data, since this is simply not present in the information extracted from the various data sources. The likelihood of an overtake being caused by a driver error, however, can be determined by using the lap times and, potentially, the outlier detection method that will be discussed later. For instance, if the overtaken driver sets an unusually slow lap while the overtaker did not, one (and therefore the model) can assume that the disadvantaged driver made an error. Similarly, if both drivers (i.e., the overtaker and the overtaken driver) set an unusually slow lap time, it is likely that they had either a tough fight on track or (depending on the deviation of the lap times compared to the others) even made contact.

Besides overtakes taking place, rules are also defined to identify when there is serious potential for an overtake in the coming lap. This is done by checking whether a driver is in the DRS zone of the driver in front, which is the case when the gap between them is less than a second. It is not uncommon for drivers to have DRS enabled for multiple

laps in a row. To avoid the same information being sent to the BGM over and over, the experimentation model has been adjusted to only forward the event if it did not do so yet in the preceding laps. By implementing the changes to the model during experimentation, it was eventually able to identify interesting 'near overtakes'.

Although the DRS zone gives a proper indication of an overtake that can happen soon, a driver that laps significantly faster than its competitor ahead, while the gap is only a few seconds, an action described as *approaching* event can be identified. An algorithm has been set up to check if the gap between a pair of drivers is less than 5 seconds and the driver has gained more than a second in the current lap, it detects this event, indicating that an overtake might take place a few laps ahead. Again, it is adjusted to exclude the events when they occurred in the lap(s) before.

Retirement

Apart from yellow flags or even a safety car being waved, which is communicated through the RCMs, the data set does not directly provide information about retirements, whether it is caused by a crash, car failure, or other circumstances. Fortunately, records can be found in the data that shows the lap that a driver is in *and* the percentage of the lap they have completed. If these records stay the same in all data items for a given lap, the model can assume a driver is out of the race. Experimentation has pointed out that this simple method is effective in detecting a retirement, although there are no records in the data that can help retrieving what caused the retirement: this would require, e.g., radio messages or video frames. The only thing the model can derive from it is whether the driver retired their car on track or it drove into the pit lane by themselves, by simply checking if the number of pit stops has also increased. In addition, the model can assume a retirement on track or crash when a yellow flag is waved or the safety car is deployed.

Race control message

Deriving actions and parties from the set of RCMs turned out to be a slightly more complex task than expected. The race control stewards likely aim to communicate as short and clear as possible. This means the messages are to-the-point, but are often not composed in a grammar that can easily be understood by a computer. Therefore, the phase of experimentation lead to the insight of needing regular expressions to extract key information from the messages and categorize them correctly.

Using a set of rules, the experimentation model looks for messages of specific categories: *car events*, *flags waved*, and *other*, which are the labels already present in the data. In some of the messages, one or more drivers are involved. These are seen as the subjects of the action and are extracted via a regular expression that finds them by their 3-letter uppercase name codes, which are always surrounded by brackets, which simplifies the identification process.

For some of the message categories, the car events in particular, the actions and parties could be extracted by using a combination of DP and PoS tagging. Experimentation pointed out that, by searching for the verb present in the message and retrieving its direct object, the action and main object could be identified in an accurate manner. Taking into account that all RCMs within a category follow the same structure and have a limited set of options, this accuracy is likely to maintain for new data records.

Besides the object that is (potentially) extracted via DP, there is often additional relevant information present in the message, such as the location of the event. Analysis of the RCMs made clear that the mentioned location is always either a turn or a sector. The location object could, therefore, easily be identified using a simple regular expression.

In example 11.1, an overview of the approaches used for retrieving an action from an example message is shown.

Example 11.1: Overview of the approaches used for retrieving an action from a blog post

"CAR 14 (ALO) MISSED THE APEX OF TURN 5"
 Driver -> 'ALO' -> regular expression
 Action -> 'miss' -> PoS tagging
 Object (main) -> 'apex' -> dependency parsing
 Object (location) -> 'turn 5' -> regular expression

Race control, i.e., the race stewards, inform the public when they hand out a penalty or investigate an incident. Whenever this is the case, the RCM always contains the token *penalty* or *incident*. In addition, the reason behind the penalty or investigation is often stated behind a hyphen. Identifying such events is, therefore, achieved by simply searching for the occurrence of words in the message and using PoS tagging and DP to check if there is an event present in the cause behind the hyphen. If the latter is the case, an additional data object is composed that represents the action that leads to the message.

Example 11.2: Events extracted from a RCM

"FIA STEWARDS: 5 SECOND TIME PENALTY FOR CAR 55 (SAI) - SPEEDING IN THE PIT LANE"
 -> Sainz speeds in the pit lane
 -> Sainz gets 5 second time penalty

Example 11.2 illustrates this process. From the RCM, not only the measure (i.e., the penalty) is seen as an event: the event of the driver speeding in the pit lane is also extracted as an event, since this is something that the model cannot extract from the numerical race data by itself. Without including the additional event, the blog reader might be confused about the cause of the penalty.

Unusual lap time

2: The *Autosport.com* platform was used for this comparison.

To distinguish 'regular' lap times from potentially unusual ones, OD is performed by using Z-scores. During an experimentation session in which the threshold value was adjusted iteratively, while comparing the resulting events marked as noteworthy to similar actions mentioned in existing live blogs², a value of 1.5 turned out to provide a correct balance of noteworthy lap times being recognized while not including an excessive number of lap times in the filtered data. Example 11.3 shows a data object of the lap time of a specific driver that is considered as unusual by the experimentation model. Since it uses the Z-score with a threshold of 1.5, it knows that this particular time is an outlier as the difference between the driver's time and the average time in the regarding lap exceeds the standard deviation by more than 50%.

Example 11.3: Data object representing an unusual lap time

```
{'subject': [{'driver': 'LAT'}], 'action': 'drive', 'object': {'main': ['1.361s slower', 'than rest', 'on average'], 'timing': ['lap 13', 'lap_time': ['96.683']]}}
```

If a driver, for example, makes a pit stop, their lap time increases with around 20 seconds in that lap. Not only would these times be seen as outliers, they also impact the lap statistics and including these lap time values therefore negatively influences the detection of unusual lap times. Considering this, the lap times of drivers that have made a pit stop are excluded from the array that is used to calculate the mean and

standard deviation, as well as the times of the retired drivers as these always remain stuck on the lap time of their last completed lap.

The Z-score method is also used for classifying a *new* lap time for a driver as unusual and therefore interesting. The main difference between the two methods, however, is that in OD all lap times, including the ones evaluated, are taken along in the mean and standard deviation calculation, while in ND the new lap time is *not* included.

```
[[{'subject': [{'driver': 'NOR'}], 'action': 'slow down', 'object': {'main': 'massively', 'lap_time': [115.077]}]]
```

Example 11.4: Data object with token indicating the difference

As shown in example 11.4, an additional token is included to indicate how much the new record deviates from the average lap times in the previous few laps: for the experimentation model to note that the driver slowed down 'massively', the lap time has to differ more than 4 seconds, which is seen as a large difference in *Formula One*. In the case of example 11.4, the average lap time of the involved driver in the preceding laps was around 107 seconds, which the displayed 115 seconds deviates nearly 8 seconds from: the experimentation model, therefore, included the term *massively* in the filtered data object.



At this stage, we have a good view on what rules we can use to retrieve key information from the race data. The methods and insights we have discussed in section 11.1 can be used to experiment with creating the actual data objects that are suitable for forwarding to the BGM. In the section hereafter, we will discuss further experiments and insights that involve an overall model that implements multiple rules and events.

11.2 Event filtering

For many race fans, what makes *Formula One* an interesting sport is the number of events that can be going on at the same time. For the event identification task, however, this can be challenging. When applying all the rules discussed in the previous section, the experimentation model returned an array of over 10 events in some of the busier laps in a race. Because this is not likely to result in short, coherent, and interesting blog posts, some filtering needs to take place.

To limit computational cost, the steps of filtering the data based on the relevance of the events is done before retrieving all of them. First, the model checks for presence of events that are considered to be the most important ones: overtakes, pit stops, retirements, and car events. If the model did not detect any of these, it continues by extracting the information that is seen as slightly less important, but still noteworthy: new fastest laps and RCMs. Afterwards, if it still did not identify events, while it also does not detect a DRS or approaching event, the final step is to check for an unusual sector or lap time. The point in time is skipped if there are still no events identified.

The proposed concept approach implemented in the experimentation model eventually lead to the model extracting a limited number of interesting events, as shown in an example of one of the results in figure 11.1. If this data object would be forwarded to the BGM, it could combine it into a single blog item, such as the one in example 11.5.

"While the engineers of Gasly and Albon are busy changing their tyres, Ricciardo gets the move done on Ocon in the Alpine to move up into the points."

Example 11.5: Blog post example retrieved from Autosport.com [33]

Figure 11.1: Result of events identified by the experimentation model in a single lap


```
{
  "Lap 15": [
    {
      "subject": [{ "driver": "RIC" }],
      "action": "overtake",
      "object": { "driver": ["OCO"], "position": ["10"] }
    },
    {
      "subject": [{ "driver": "GAS" }],
      "action": "pitstop",
      "object": { "timing": ["lap 15"], "lap_time": ["100.169"] }
    },
    {
      "subject": [{ "driver": "ALB" }],
      "action": "pitstop",
      "object": { "timing": ["lap 15"], "lap_time": ["100.786"] }
    }
  ]
},
```



The results of the experimentation steps we dived into in section 11.2 have shown us that our rule-based approach - which starts by scanning for the most important events and checking for less interesting ones only when no events are yet detected - is a strong method for working with our specific data set. The events that are outputted by our current experimentation model are, however, lacking some variety in words used. In section 11.3, we will discuss what we can improve to end up with an even-better performing model.

11.3 Term variety

The model that resulted from the experiments discussed in the previous section was able to create accurate data objects representing actions that take place in each lap. Unfortunately, because of the rule-based nature of the model, the results are limited in variety. Considering that it is likely for the terms used in the input and output of the fine-tuning data set for the LLM to be similar as well, since these are extracted from the original posts, it is desired to have varying terms in the data to ensure more variety in the eventually generated posts. Therefore, additional experiments were conducted that explored the use of additional techniques to vary in used terms, especially since the data extracted from the example blogs, which is used for fine-tuning the BGM, also includes various terms representing the same (or similar) events.

 The synonym set can be found on [GitHub \[77\]](#).

Using NLP methods such as similarity scores or synonyms did not result in the desired outcome due to the high amount of *Formula One*-specific jargon used for describing events, e.g., overtakes and pit stops. Experimentation pointed out that the use of a manually written synonym data set, from which the model randomly picks an option, is the most appropriate approach to achieve more variety in the terms used in the data objects. As shown in example 11.6 - retrieved after the final adjustments to the experimentation model, the output contains varying terms for representing the same events. For instance, the event *stop for tyres* is the same as *make pit stop* in the example.

Example 11.6: Data objects with varying event terms

```
Lap 15
{'subject': [{'driver': 'RIC'}], 'action': 'fly', 'object': {'main': ['past'], 'driver': ['OCO'], 'position': ['10']}},
{'subject': [{'driver': 'GAS'}], 'action': 'stop', 'object': {'main': ['for tyres'], 'timing': ['lap 15'], 'lap_time': ['100.169']}},
{'subject': [{'driver': 'ALB'}], 'action': 'make', 'object': {'main': ['pit stop'], 'timing': ['lap 15'], 'lap_time': ['100.786']}}
```



After section 11.3, we have learned that the use of a manually created set of synonyms for domain-specific jargon is a suitable method for realizing variety in the filtered event data. Experimentation with the LLM, however, is needed to conclude whether this approach results in sufficient originality in the blog posts, which will be discussed in chapter 13.

12

Structured data extraction

What method is most suitable for creating our fine-tuning data set?

12.1 Large language model	48
12.2 Linguistic feature extraction	49

The two proposed methods for extracting structured data objects from the unstructured example blog posts discussed in chapter 9 are tested and evaluated to determine the most suitable approach for establishing a set of key-value pairs for fine-tuning the LLM. In below subsections, the relevance, setup, and results of each of the experiments is discussed, followed by a comparison and selection of the methods.

12.1 Large language model

To evaluate whether an LLM could be a suitable method for the desired task, an experiment was conducted in which a model was prompted to extract data from a small set of four example posts. In the experiment, a pre-trained *Falcon* [99] model with 7 billion parameters was used. Four different blog posts, each with the same instruction (i.e., that the model should generate a JSON object of the action discussed in the blog, without specific details about which keys to use), were prompted to the model.

1: The results can be found in appendix section H.1, table H.1.

The resulting JSON objects were not uniform on multiple levels¹, such as the key names, amount of detail, and use of full names and last names. In addition, the model included some info in the structured representations that cannot be found in nor derived from the content of the blog posts: it has chosen a random race with incorrect dates and included Vettel as Hamilton's team-mate, while these two drivers have, in fact, never been part of the same team.

To get a better understanding of the capabilities of the LLM, experimentation was continued with prompt engineering, in which a more specific template for the JSON object was defined in the prompt. Providing an example to the model, either in the form of a fully complete JSON object or just an overview of the keys, unfortunately did not lead to improvement of the results. Now, the model included a full sentence (i.e., "Max Verstappen overtook Leclerc after the Safety Car restart") rather than a short representation of the action (which should be "overtake" in the example), left all fields blank, simply returned an exact replicate of the example JSON object - which had nothing to do with the event discussed, or included actions, drivers, or teams that did not occur in the provided blog post.

Example 12.1: Instruction to include specific information in data object

"In the JSON object, please include the type of action, drivers and teams involved, and indicate which driver and/or team has an advantage in the situation"

2: The results can be found in appendix section H.1, table H.2.

By explaining in the instruction what information should be included in the data object in textual form, i.e., adding the sentence in example 12.1, the resulting structured representations of the blogs improved on some points: the actions/events were now identified correctly for all four prompts and the connected subjects were also included in the output data objects². However, the keys used were not uniformal (i.e., *typeOfAction*, *action*, *type*, and *event* for representing an action), the data included one or more hallucinations in three out of the four results, and the data did not make clear who was (dis)advantaged, despite this being stated in the instruction.



In the results of section 12.1, we have seen that the resulting JSON objects did not meet our desired quality. This highlights the importance of fine-tuning, even for the data extraction task. For fine-tuning, however, we need a sufficient amount of training data. In our case, such training data requires manual creation of the input-output pairs, which is time-consuming and does still not guarantee a well-performing model for data extraction. Deploying LLMs for creating the input objects is, therefore, no longer considered as an appropriate option.

12.2 Linguistic feature extraction

The exclusion of deploying an LLM as one of the options, as discussed in the previous section, put all hopes on the LFE concepts. Iterative experiments pointed out that training a NER model to identify a wide range of entities did not result in accurate performance due to the limited availability of training data, implying that using NER as the only method is not a solid option. Therefore, the labels were reduced to a set of 7 unique entities (driver, team, position, timing, gap, tyre, and location) so that the NER approach could be used as support for the other LFE methods.

Furthermore, initial experiments with DP showed that solely using the root, subject, and object did not come near an accurate representation of the discussed actions, especially for more complex sentences. This method, therefore, requires inclusion of multiple methods as well. The subsequent experiment focused on a combination of LFE methods. As mentioned, there are two variations for such a combined model considered: one with DP, the other with SRL. Hereafter, experimentation with - and comparison of - these options is discussed.

12.2.1 Dependency parsing

By using the LFE model with DP, it is intended to find representations of an *action*, its *subject*, and its *object*. Below, the steps that are involved in this approach are listed.

1. Replace all tokens that have a named entity in their cluster with that entity using *coreference resolution*
2. Replace all pronouns that do not have a named entity in their cluster with a (proper) noun in their cluster using *coreference resolution*
3. Collect all verbs in the sentence using *Part-of-Speech tagging*
4. For each verb, get the nominal subject using *dependency parsing*
 - ▶ If there is no subject linked to the verb, skip the verb for now
5. Get the full form of the subject by checking for compound dependencies (e.g., *Aston Martin*)
6. Get additional subjects by checking for conjuncts (e.g., *Zhou and Albon*)
7. Check for an open clausal complement dependency of the verb, and if this is present, replace the verb with this complement (e.g., ~~managed to~~ *overtake*)
8. Retrieve important related tokens of the verb by checking for negation modifiers (e.g., *not improving*), particles (e.g., *comes out*), prepositions (e.g., *stops to*), and adverbial modifiers (e.g., *goes wide*) and store these in an array representing the action
9. For each token in the action array (e.g., [*comes, in, to*]), retrieve the objects of the action by checking for object or attribute dependencies (e.g., *comes out in front of Leclerc in 2nd place*)
10. For each token in the subject and object arrays, check if they are a named entity (i.e., they have an entity label)
11. For all entities with label *driver*, retrieve their unique name code via the *Ergast* API
12. Store the subjects and objects in JSON objects together with their entity labels ('subject': [{'driver': 'LEC'}])

i A discussion of the NER experiments can be found in appendix section H.2.

i A discussion of the dependency parsing experiments can be found in appendix section H.2.

Recap of abbreviations

LLM - Large Language Model
LFE - Linguistic Feature Extraction
NER - Named Entity Recognition
DP - Dependency Parsing
SRL - Semantic Role Labelling

3: The prepositions were stored in the action array to extract all linked objects, but they are not needed to represent the action itself.

13. Store the action by joining the lemmatized version of the tokens in the action array that are not prepositions³

As can be seen in figure 12.1, the data extracted from the blog posts using the proposed method is not without flaws. The actions of the driver Zhou being lucky and the car ending up on the back of the tyre barrier are identified correctly, while the event of his car flipping and ending up in the gravel is left out. This is caused by the rule of solely including an action if either the subject or the object includes a named entity, which is not the case for the subject *car* and the object *gravel trap* here.

```
"Replays show just how lucky Zhou was, as his car is flipped, skidded on its
roll hoop before digging into the gravel trap. The car eventually ends up
between the back of the tyre barrier and catch fencing.": [
  {
    "subject": [{ "type": "DRIVER", "name": "ZHO" }],
    "action": "be",
    "object": [{ "type": "unknown", "name": "just how lucky" }]
  },
  {
    "subject": [{ "type": "unknown", "name": "car" }],
    "action": "end up",
    "object": [
      { "type": "unknown", "name": "back" },
      { "type": "TYRE", "name": "tyre barrier" }
    ]
  }
],
```

Figure 12.1: Example of a structured data extracted from a blog post

To improve the representation of actions, more named entity labels and retraining the NER model may be necessary. However, this may result in a higher number of actions in the structured data, potentially excluding key information. Including non-labelled entities, on the other hand, is also not desired. Fine-tuning the model on such unclassified data is not useful, as it exposes the risk of creating blogs with a high amount of imaginary actions, subjects, and objects. An alternative method is, therefore, to use manual rules to connect tokens to appropriate categories, such as 'car', to avoid overloading JSON arrays with uninteresting actions.

In the example shown in figure 12.1, this would mean that, e.g., the object that includes the string *car* is labelled to be in a new defined category 'car', as naming a car in either the subject or object often indicates a noteworthy event. This would result in three additional objects being included in the data of figure 12.1, that would be formatted similar to example 12.2.

Example 12.2: Format of data objects after including manual labelling

```
{'subject': {'type': 'CAR', 'name': 'his car'}, 'action': 'flip'},
{'subject': {'type': 'CAR', 'name': 'his car'}, 'action': 'skid', 'object': {'type': 'unknown', 'name': 'roll hoop'}},
{'subject': {'type': 'CAR', 'name': 'his car'}, 'action': 'dig', 'object': {'type': 'unknown', 'name': 'gravel trap'}}
```

Adding this data to the original result in figure 12.1 leads to a structured representation of the blog in high detail. There is, however, still an excessive amount of data present that is not necessarily key information and/or data that could be passed on by the EIM.

12.2.2 Semantic role labelling

As mentioned in chapter 7, SRL can give other insights into the actions and agents within a sentence compared to the approach focusing on DP. Using the AllenNLP library [91], the method was tested on a subset of the example blog data set.

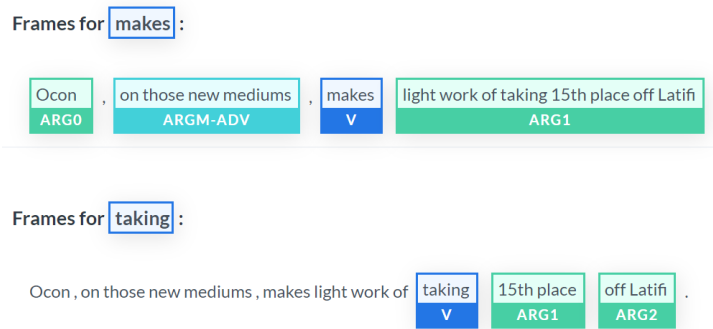


Figure 12.2: Semantic roles identified in an example blog sentence using the AllenNLP demo tool [91]

From the result of one of the examples shown in figure 12.2, one could say that the approach combines a relatively large number of tokens in a single argument and does not always accurately link the right arguments to a verb, as *Ocon* would in theory also be an argument of *taking*. Especially for longer blog examples, such as example 12.3, the resulting overview of semantic roles contains multiple long phrases as arguments for five different verbs that are seen as the predicates, which is not suitable for using in a short, structured data object.

"People's champion Vettel, denied a chance to fight for big points due to his slow pit stop, is 12th and closing up to a train of cars led by Albon in ninth."

Example 12.3: Challenging blog post for SRL



While both LFE options can provide valuable insights into the syntactic and semantic relations between tokens and phrases in a sentence, the SRL method seems less suitable for extracting a structured representation due to the high number of tokens included in a single argument. Therefore, we consider the LFE approach that uses DP (supported by NER and PoS tagging) to have most potential for our case. During implementation of the model, we need to pay attention to the use of uniformal terms for certain actions⁴ and addition of manual rules to categorize certain entities. Furthermore, we need to perform proper evaluation on the resulting blog posts to get insight in the LFE model's performance, as this can have impact on the performance of the BGM as well.

4: E.g., seeing the actions *overtake*, *fly past*, and *breeze past* as the same action.

13

Blog generation

What model is most suitable for generating the posts?

13.1 Initial model . . .	52
13.2 Fine-tuned model	52

To get an idea of the capabilities of the BGM component and the steps that potentially need to be taken before actually implementing the system, experimentation was performed with both discussed (i.e., MVP and MTL) models. The experiments and new insights are discussed in the following sections.

13.1 Initial model

Using the inference tool, the performance of the MVP and MTL data-to-text models was assessed, prior to any fine-tuning. When inputting relatively simple data representing a *Formula One* event, both models generated similar text that described the event, although they both failed in converting name codes to names, and the output quality was poor when using domain-specific jargon, e.g., the action *pitting for hard tyres* in the second example of table 13.1.



The results of the original data-to-text models showed us that, for the model to be able to work with common *Formula One* terms and driver names, fine-tuning is a crucial step. We will dive further into experimentation with fine-tuning in section 13.2.

13.2 Fine-tuned model


Both data-to-text models were fine-tuned on the input-output pairs that were realized while experimenting with the LFE method discussed in the previous chapter. The process was performed in Python, using - among others - the *Transformers* library [95]. The efficiency and speed of the process was optimized by working in *JupyterLab*, a web-based environment available via the *University of Twente* [100], and applying low-rank adaptation (LoRA), a parameter-efficient fine-tuning (PEFT) method, which has shown promising results in fine-tuning LLMs without the need for highly advanced computational resources [101]. LoRA speeds up the fine-tuning process by freezing the weights set during pre-training, which reduces the number of trainable parameters, and therefore the graphics processing unit (GPUs) memory, significantly [102].

When prompting the fine-tuned models to generate live blogs based on the EIM output, with the input data converted into the format of the pre-training data of the model, in a similar fashion to example 13.1, it became clear that the output of the MVP model was limited to exactly the data present in the input (table 13.1, example 3), while the MTL model included imaginary content, i.e., the part about the Red Bull Drivers' Championship (table 13.1, example 4). Considering that there is no such championship, nor is either of the involved drivers part of the Red Bull team, such information inclusion is undesirable.

"Write an entertaining live blog post describing the following event in a Formula 1 race: Agent1 | driver | VER [SEP] Action1 | lost [SEP] Object1 | main | 14 positions [SEP] Object1 | main | since start [SEP] Object1 | position | current [SEP] Object1 | position | 4 [SEP] Object1 | timing | lap 15"

Example 13.1: JSON object converted into prompt format

When prompted with data that includes a single action, such as in the third example of table 13.1, the MVP model was able to produce posts that covered the full event in a logical structure. In contrast, as seen in table 13.1, example 5, hallucinations increased (e.g., driver *Verstappen* is mentioned, while he cannot be found in the data), while readability decreased (e.g., it is not clear who *0.411s* refers to), when prompting arrays of multiple data objects, i.e., multiple actions. During a brief analysis of the posts generated throughout a full race, similar mistakes were obtained in multi-event data arrays.

 An overview of the eventually generated posts can be found on [GitHub \[77\]](#).

Agent	Action	Object	Output	
<i>Before fine-tuning</i>				
1	Verstappen (D)	overtake	Norris (D), 2 (P)	Verstappen overtook Norris for second place.
2	VER (D)	pit	hard tyres (M), 2 (P)	VER was in 2nd position when the main object was hard tyres in
<i>After fine-tuning</i>				
3	HAM (D)	gain	position (M), RAI (D), 1 (P)	Hamilton has gained a position on Raikkonen, who is now 1st.
4	HAM (D)	gain	position (M), RAI (D), 1 (P)	Hamilton gains the position from Raikkonen to take the lead of the Red Bull Drivers' Championship.
	NOR (D)	has	DRS (M), SAI (D), 15 (P)	Norris is on the tail of Sainz, 0.411s behind, in the drs zone.
5	MSC (D)	has	DRS (M), MAG (D), 12 (P)	Magnussen is 0.559s behind and Verstappen is driving on his tail.
	VET (D)	has	DRS (M), MAG (D), 13 (P)	

Table 13.1: Examples of posts generated by LLMs during experimentation

In table 13.1, the following label abbreviations are used:
D - driver
O - organization
M - main
P - position
L - location

The EIM is designed to use various terms for the same events. Considering that the data objects generated with LFE include the exact same action descriptions as the corresponding blogs, fine-tuning has led to the LLM using the exact term used in the data object, and has not learned that, e.g., *breeze past* and *overtake* have a similar meaning. In table 13.1, example 5, the model has, therefore, not understood that the three actions (each represented with randomly chosen action representations) are the same, which likely caused the lack in output quality. The inclusion of the RPM could be a solution to this 'flaw': in chapter 14, this will be discussed further.

While experimenting with the fine-tuning setup, various LoRA configurations were tested for both of the models, as well as various training arguments. A substantiation of these experimentation choices, as well as the final decisions, can be found in the implementation chapter, i.e., chapter 15.



The outcome of the experiments and the iterative adjustments done based on new insights have shown us that fine-tuning the MVP model on the input-output pair data set improves the quality of the generated blogs and enables it to generate blog posts based on data provided by the EIM. The model struggled with producing readable content for multi-event input, which can potentially be solved by the RPM. We will discuss this matter in the chapter hereafter.

In chapter 10, the *OpenChat* model was introduced as an option for rephrasing the output of the BGM in a more entertaining manner. In this chapter, experiments that were conducted to assess the capabilities of the model in the *Formula One* field are discussed, which lead to a better view on the steps to be taken during the implementation of the overall system.

To kick off the RPM experimentation, a set of informative blogs, established during experimentation with the BGM, was inputted to the model using the *Transformers* [95] library. Remarkable here was that, despite the fact that it was not told that the text involved *Formula One*, the model was able to recognize the sports discipline and even knew correct facts about the drivers - such as first names and nationalities. The included information, however, seemed somewhat outdated, as can be obtained in the second example in table 14.1. Stroll has been in *Formula One* for multiple years, which means he is no longer a rookie, and Verstappen is no longer particularly young compared to the others.

Besides the outdated details, the model made a few slightly incorrect interpretations. For instance, it used the phrase *slams on the brakes* for representing the action *stop*, which should refer to a pit stop and not a 'braking' stop. This same lack of understanding in pit stop events is seen in the third example (table 14.1), where the post implies that *undercutting* another drivers means they want to overtake *on track*, while this is, in fact, done via a pit stop in such cases.

	Input	Output
1	Sirotkin stops for hypersoft tyres and stays in 12th.	Sirotkin slams on the brakes and swaps to hypersofts, maintaining his 12th position.
2	Verstappen overtakes Stroll to move up to 16th.	Verstappen, the young and talented F1 driver, pulled off an impressive move to pass Stroll, the rookie, and secure his position at 16th on the track.
3	Stroll pits from P19 to undercut Gasly, who is still out.	The Canadian driver, Lance Stroll, is making a bold move by attempting to undercut Pierre Gasly, who is still out on the track. Stroll is currently in P19 and is looking to make a move up the grid.

Table 14.1: Examples of posts generated by the *OpenChat* LLM during experimentation



The experiments we discussed in this chapter showed that the model we have selected performs well on the rephrasing task itself, while it also has basic understanding of *Formula One* concepts. Proper insights into its performance in terms of improving perceived entertainment require evaluation, which will take place after implementing the model, which is to be discussed in section 15.3.



Research question recap

Part IV - Experimentation

✓ RQ5. What (language) model is most suitable for data-to-blog generation?

✓ 5.1. What suitable (types of) models are available?

The MVP model, pre-trained for data-to-text generation have shown to be most capable of generating posts with a limited amount of imaginary information.

⚙️ RQ6. What steps are needed for collecting and processing the data?

✓ 6.3. What data processing techniques are needed?

Scraping the blogs from the web platform, filtering out tokens indicating interviews or radio messages from the blog posts, extracting race data from different sources via *Python* and *Node.js*. Furthermore, a structured data extraction model is needed to create a set of input-output pairs from the blog dataset.

IMPLEMENTATION AND METHODOLOGY

15

Implementation


How are we implementing the proposed architecture?

15.1 Event identification model	58
15.2 Text generation model	58
15.3 Rephrasing model	60


Using - among other obtained insights - the results of the experimentation phase, the components in the system architecture were implemented. Below sections provide a discussion of the steps taken, together with a description of the choices made.

15.1 Event identification model

The EIM has been implemented using various *Python* objects. The main object of class *Interpreter* is responsible for calling other action objects, which are each responsible for identifying a specific action (or set of actions), and it interprets the data returned by those. In table 15.1, an overview is given of the actions and corresponding criteria that the EIM can identify, sorted according to priority. If one or more actions of a higher-priority level are identified, the model does not continue to identify any further events and returns solely these events. However, to prevent the data lacking variety in case of, e.g., many *priority 1* events in a short time frame, the algorithm decides to include an additional event of one of the other priority levels on random basis, which happens approximately 30% of the cases.

 The code of the EIM can be found on [GitHub \[77\]](#).

Furthermore, again to prevent lack of variety in the data objects, a function is implemented that adjusts the terms used for certain actions. For instance, the action *retire* can be represented as - among other terms - 'retire' (action) 'car' (object) or 'is' (action) 'out of race', which is randomly chosen by the model.

 Examples of data generated by the EIM can be found on [GitHub \[77\]](#).




Implementation of the EIM has lead to us having a set of filtered data objects that we can use for testing and evaluating our system, as well as a method to identify events in future race data, which enables the opportunity for race simulation and/or real-time implementation.

15.2 Text generation model

In this section, the implementation of the component responsible for generating the text is discussed, where the first subsection focuses on the creation of the fine-tuning data set, while subsection 15.2.2 discusses the process of fine-tuning the model on this data.

15.2.1 Structured data extraction

Taking into account the insights gained during experimentation, the LFE model has been optimized and implemented. To do so, a *SpaCy* [89] model with an additional coreference pipeline has been used.

 The scripts of the LFE approach, as well as the resulting data set, can be found on [GitHub \[77\]](#).

The steps described in subsection 12.2.1, extended with some manual rules to optimize the terms used and entities identified, were used for the implementation of the data extraction model. In summary, the model uses DP, PoS tagging, NER, and CR to store verbs that are (in)directly linked to a named entity (as either object or subject) and stores these, together with the subject(s) and/or object(s) in JSON format. To enable the

LLM to learn to vary in terms used for representing the events, similar to the approach discussed for the EIM, additional rules are included that randomize the terms used for the actions. This has led to a data set of input-output pairs of all races from 2018 to 2023, each containing an array of one or more action representations per lap.

Action	Criteria	Agent	Object
<i>Actions with priority I</i>			
Overtake	Position is lower than in previous lap, driver overtaken did not pit or retire	Driver that gained position(s)	Driver that lost position(s), new position of agent
Pit stop	Number of pit stops increased, driver did not retire	Driver that pitted	New position, tyre compound, driver subject to undercut (optional)
Safety car deployed	RCM that mentions 'safety car' and 'deploy'	Safety car	Timing (lap)
Safety car ending	RCM that mentions 'safety car' and 'ending' or 'in this lap'	Safety car	Timing (lap or 'soon')
Car event	RCM of type 'Car event'	Driver mentioned in RCM	Dependent of objects in RCM
Retirement	Relative position and lap did not change during lap	Driver that retired	Timing (lap), part of lap
<i>Actions with priority II</i>			
New fastest lap	Lap time is smaller than all previous lap times	Driver that set the time	Lap time
Race control event*	RCM that is not a (safety) car event	See table 15.2	See table 15.2
<i>Actions with priority III</i>			
In DRS zone	Gap smaller than second, not in DRS zone in previous lap	Driver in DRS zone	Driver in front, position, gap
Approaching	Gap smaller than 5 second, lap time at least 0.5 second faster	Driver that is approaching	Driver in front
<i>Actions with priority IV</i>			
Outlier	Z-score exceeds 1.5 compared to other drivers, no pit stop made	Driver with outlier time	Fast or slow, timing (lap), lap time
Novelty	Z-score exceeds 1.5 compared to previous laps, no pit stop made	Driver with novelty time	Fast or slow, lap time

Table 15.1: Implementation of the event identifier

15.2.2 Fine-tuning

As discussed in the experimentation chapter, the pre-trained MVP data-to-text model has been selected for the generation task. The data set established using LFE was used for fine-tuning, where the JSON objects are converted into the same data representation the model was pre-trained on. A server with two GPUss of *JupyterLab* has been used

Table 15.2: Prioritization of race control events

Action	Criteria	Agent	Object
<i>Actions with priority I</i>			
Penalty	RCM that mentions 'penalty'	Driver receiving penalty or FIA	Cause of penalty
Off track	RCM that mentions 'off track'	Driver off track	Info mentioned in RCM
<i>Actions with priority II</i>			
Non-blue flag	RCM of type 'Flag' with a colour that is not 'blue'	Driver(s) involved or FIA	Flag colour, location (optional)
Incident	RCM that mentions 'incident'	Driver(s) involved or FIA	Short incident description
<i>Actions with priority III</i>			
Blue flag	RCM of type 'Flag' with colour 'blue'	Driver for who the flag is waved	Blue flag, timing (lap)
Track limits	RCM that mentions 'track limits'	Driver exceeding limits	Track limits, timing (lap), location

 The fine-tuning script can be found on GitHub [77].

for the fine-tuning process. The main two *Python* libraries used are *Transformers* [95] and PEFT.

Given the relatively small size of the fine-tuning data set, which might expose the risk of overfitting when passing the data through the algorithm too often, fine-tuning was performed with a total of 3 epochs. This was considered to be a solid balance point in preventing the model from overfitting, while providing it with enough resources to learn patterns in the data. In addition, to limit overfitting effects further, a dropout rate of 0.1 was added to the LoRA configuration.

Although the original pre-trained model was already capable of extracting patterns from data and converting these into natural language, it does not have sufficient knowledge and capabilities of understanding certain language patterns that are related to the *Formula One* domain. Also taking into account, again, the limited data set size, the model turned out to benefit from a somewhat low rank and alpha [103], i.e., 8 for both. This LoRA setting resulted in a total of 1,179,648 trainable parameters, which is 0.25% of the total number of parameters¹.

Throughout the epochs, a decrease in loss (i.e., 0.91 to 0.58) was obtained, indicating that the model succeeded in improving its output predictions during fine-tuning.



At this point, we have established a fine-tuning data set and used this to fine-tune a pre-trained LLM, resulting in a model that is, in theory, able to generate the desired blog posts, although this still needs to be assessed by evaluating the results later on.

15.3 Rephrasing model

Given that the rephraser involves a pre-trained LLM that will not be fine-tuned, mainly due to data restrictions, the process of implementation is a relatively simple one. Similar to the approaches for the other two models, the implementation was done in *JupyterLab* [100], where the *OpenChat* [98] model was loaded via the *Transformers* library in Python.

 The notebook of the RPM can be found on GitHub [77].

Using manually defined functions, sets of blog posts were forwarded to the LLM by extending each of them with the instruction "*Rewrite the following in a more entertaining way:*". To easily import the files in the evaluation stage, the input and output prompts were saved as pairs in comma-separated value (CSV) files.



After we implemented the RPM, the full architecture has been realized and there are units that are ready to evaluate. In the chapter following hereafter, we will discuss how we will perform this evaluation.

16

Evaluation setup

How will we evaluate our system?

16.1 Accuracy 62
16.2 Readers' perception 65

This chapter revolves around the methods used for evaluation of the system. It is divided into two main subjects: evaluation of *accuracy* and evaluation of *entertainment*, which are discussed separately in below sections.

16.1 Accuracy

To determine the accuracy of the model, there are three main criteria to consider, as listed below:

1. Does the filtered data cover the most noteworthy events?
2. Do the posts contain all information from the input data?
3. Do the posts solely contain the information of the input data?

1: The posts from Autosport.com [33] are used for comparison.

Here, the first question can be seen as evaluation of the EIM, whereas the other two assess the performance of the generation component. To determine the quality of the EIM, a manual comparison of generated and human-written posts¹ of sequences of 10 laps in 10 different races is performed. In each season between 2018 and 2023, excluding 2020 due to Covid cancellations, two races are selected randomly, with the sequences of 10 laps also being randomly selected. In below subsections, the evaluation process of the EIM and BGM are discussed separately.

16.1.1 Event identification

For evaluating how well the EIM has identified noteworthy events in the data, two sources are needed, each representing the same sequence of 10 laps in the same race. In short, the filtered data objects (i.e., the output of the EIM) during those laps are compared to the sequence of human-written blogs written during those same laps.

For each event in the filtered data, the blog sequence is manually checked for the same event being mentioned. For each event that is mentioned in the blog posts, as is the case for the pair in example 16.1, a point is awarded. Furthermore, the events that are mentioned in the blog posts but are not found in the data are counted. In the end, this leads to a percentage of events that are noteworthy (i.e., the percentage of events in the data that are also found in the posts) and a percentage of events that are missed by the EIM (i.e., the percentage of events that are not found in the data).

Example 16.1: Pair of a data object and a blog post representing the same event

```
{'subject': [{'driver': 'NOR'}], 'action': 'overtake', 'object': {'driver': ['BOT'], 'position': ['P7']}}  
"The defence of the Finnish driver did not last long. He is overtaken by Norris and drops to 8th."
```

It is important to note that the events that simply cannot be extracted from the input data, such as the precise reasons for a retirement or radio messages, are not taken into account in the scores.

16.1.2 Blog generation

In contrast to the EIM evaluation, the human-written posts are not used for evaluating the performance of the BGM. Here, the input (i.e., the filtered events) is simply compared to the newly generated posts. A score between 0 and 1 is awarded for the subjects, actions, and objects separately. These three factors are rated individually to simplify the process of identifying where the main flaws and/or points of improvement can be found. In addition, so-called penalty points are given when mistakes are noticed that are not necessarily connected to those three aspects or when imaginary data is included in the posts. Figure 16.1 shows the principles of this point system.

Subject mistakes	
Name of different driver	0.5
Imaginary name	0.25
Subject missing	0.5
Wrong entity used as subject	0.75

Action mistakes	
Incorrect number of actions	0.33
Action missing	0.25-1
Incorrect interpretation	0.75
Similar actions not recognized	0.25

Object mistakes	
Name of different driver	0.5
Imaginary name	0.25
Driver missing	0.5
Position missing	0.33
Other object missing	0.25
Odd/incorrect preposition	0.25
Wrong entity used as object	0.75
Other	0.25-1

Penalties	
Wrong cause assumption	0.5
Mix-up of driver order	0.75
Imaginary action	1
Imaginary object/subject	1
Other	0.25-1

Figure 16.1: Overview of the scoring system for evaluating the BGM accuracy

For a flawless interpretation of the subject, action, and/or object, they get a score of 1 each. The points shown in the mentioned figure are subtracted from 1 for each mistake, with a minimum score of 0. The penalty points, on the other hand, are subtracted from the sum of the scores of the other three. This means that, e.g., a post that scores the full points on all three aspects but mentions imaginary information twice gets a total score of 1 instead of 3. This is illustrated in example 16.2, where neither the soft tyre compound nor the driver Verstappen are found in the data.

```
{'subject': [{'driver': 'NOR'}], 'action': 'overtake', 'object': {'driver': ['BOT'], 'position': ['P7']}}
```

"The defence of the Finnish driver did not last long. He is overtaken by Norris and drops to 8th on his soft tyres, with Verstappen on his tail now."

Example 16.2: Pair of a data object and a blog post with imaginary information

As can be obtained in figure 16.1, the scores vary, depending on the type of mistake made. The errors that receive the highest punishment are hallucinations (i.e., imaginary entities), followed by those that involve the use of the wrong entity as subject or object - as this might completely turn around the event mentioned, incorrect interpretation of an action - since that can have large impact on the way the reader will interpret the event as well, and a mix-up of driver order: this would imply an incorrect ranking, misleading the reader.

Following thereafter, the errors that lead to a subtraction of half a point are missing entities, incorrectly interpreted driver names (e.g., *Alonso* instead of *Albon*), and wrong cause assumptions (e.g., example 16.3). These are seen as impactful on the readers' understanding of the blog, but are considered to be less crucial compared to those mentioned in the previous paragraph. Lastly, the scores are affected the mildest by errors involving incorrect, non-existing names (e.g., *Tsuoda* instead of *Tsunoda*), odd/incorrect prepositions, and similar actions not being recognized as such (e.g.,

pitting and *making a stop* not seen as a similar event): these errors mainly impact perceived quality, and are therefore seen as 'mild' generation flaws.

Example 16.3: Example of a wrong cause assumption

"Sainz is in the mirror of his rival, **but** he is only 2 seconds behind."

16.1.3 Rephrasing

To get an understanding of how the RPM scores in terms of being accurate, a third evaluation method is needed. Considering that the RPM assumes that the input is correct, i.e., it takes the output of the BGM as truth, the performance is evaluated by comparing the events discussed in the input to those present in the rephrased item.

Similar to the evaluation discussed in subsection 16.1.2, a point-scoring system is applied to guarantee a uniform way of evaluating the individual rephrased posts. Figure 16.2 shows how these points are divided. Again, in case of a mistake involving the subject, action, or object, the scores are subtracted from 1, whereas the penalty points are subtracted from the total score.

Subject	
Different subject used	0.75
Incorrect additional information	0.75
Outdated additional information	0.33
Subject missing	0.5

Object	
Different object used	0.75
Incorrect additional information	0.75
Outdated additional information	0.33
Object missing	0.5

Action	
Incorrect action used	0.75
Incorrect interpretation	0.5
Odd synonym/description	0.5
Incorrect synonym/description	0.75
Action missing	0.5
Incorrect additional information	0.75

Penalties	
Imaginary action	1
Imaginary object/subject	1
Incorrect assumption	0.75
Questionable assumption	0.5
Odd term used	0.25
Incorrect term used	0.5
Other	0.25-1

Figure 16.2: Overview of the scoring system for evaluating the RPM accuracy

Again, the division of the scores is based on to what extent the mistakes impact the reader's understanding and perception. Here, the most crucial errors are hallucinations, after which the use of incorrect subjects, objects, or actions follow, together with mistakes in additional information, synonyms, and incorrect assumptions. An example of the latter is provided in example 16.4, where the model 'assumes' Perez struggles to keep his car on track, which is not mentioned in the original post and therefore seen as incorrect. Finally, outdated information (e.g., a driver mentioned to be a rookie, while he has been in Formula One for multiple years) and the use of odd terms are seen as the least crucial mistakes, and therefore receive the lowest penalty points.

Example 16.4: Example of an incorrect assumption

"Alonso overtakes Perez in Turn 4."

"Perez struggles to keep his car on track, as he is overtaken by Alonso in Turn 4."



We now know that, for evaluating the accuracy of our system, we are comparing sequences of data objects to posts of Autosport.com [33] at the same points in time to get an understanding of how well the EIM identified noteworthy events, while we need to manually compare the data objects to the generated post to check what events are covered.

16.2 Readers' perception

In addition to accuracy, the way the generated text is perceived by (potential) readers is a crucial factor in evaluating the model. To get an understanding of this so-called reader perception, human evaluation is required. This is achieved via an online questionnaire, in which respondents are asked to rate sequences of blog posts on three criteria. First, they are asked to indicate whether they find the posts *not entertaining* (score -2), *entertaining* (score 2), or anything in between with steps of 1. The same goes for the other two scales, which range from *uninformative* to *informative* and *unclear* to *clear*.

This process is repeated for a total of six sequences of posts. The posts presented to the respondents are divided into three categories: human-written (HW), original (non-rephrased) AI-generated (AI-O) (i.e., the posts generated by the initial BGM), and rephrased AI-generated (AI-R) (i.e., the posts rephrased by the RPM, of which two sequences of 5-10 blog posts are included per category. An example of what a question within the survey looks like is given in example 16.5.

Sainz wants to overtake Verstappen, but the gap is 0.85s. Magnussen takes DRS and Ricciardo is 12th.
 Ricciardo has come in to pit with a lap-time of 99.926s.
 Albon has lost 16 positions since the start of the race. His current position is 18 on lap 34.
 ...

To what extent does the blog sequence above meet the following criteria?

Not entertaining (-2) - Entertaining (2)
 Uninformative (-2) - Informative (2)
 Unclear (-2) - Clear (2)

Example 16.5: Part of a question in the evaluation survey

It is desired to limit the potential negative effect of comparing sequences of the human-written posts and the AI-generated posts during the time period, as the accuracy can, in that case, also influence the readers' perception, which is undesirable as these aspects are evaluated separately. The sequences are, therefore, selected to be within the same race, but not during the same laps.

i The exact sequences presented to the participants can be found in appendix section I.1.

The results of the survey will be used to evaluate how the model performs on the mentioned aspects compared to a human blog writer. Since the previous survey² pointed out that the blog posts of *Autosport.com* [33] score high on each of these aspects, these posts are used for the human-written sequences in the questionnaire.

²: The survey used to get insight in potential readers' preferences, as discussed in section 5.1.



In section 16.2, we explained our method for evaluating readers' perception, giving us an idea of how we can eventually draw conclusions about the performance of the system. We will use human evaluation so that we not only get insight in the technical performance, but also in to what extent potential readers like the content so that it becomes clear where the main points for improvement are.



Research question recap

Part V - Implementation and Methodology

🔗 RQ2. What is needed for the generated posts to be *accurate*?

✔ 2.2. How can the accuracy of the blog posts be evaluated?

By a manual comparison of the events covered in the generated blog posts and the input data. A point system is used that indicates how well the model interpreted the *subject*, *object*, and *action* and penalty points are awarded in case of other errors such as hallucinations.

✔ RQ4. What main components are needed in the system architecture?

✔ 4.3. How can the performance of each component be evaluated?

The EIM is evaluated on accuracy by comparing the filtered events to human-written blog posts during the same time period. The BGM is evaluated on entertainment by comparing survey scores to those of human-written posts, and on accuracy by comparing the events present in the data to those mentioned in the generated posts.

✔ RQ6. What steps are needed for collecting and processing the data?

✔ 6.4. How can the impact of the data on the system's performance be evaluated?

For the generated posts, biases, incorrect interpretations, and imaginary information can be linked to the blog dataset if the accuracy in case of no flaws in the EIM output. Missing events that should have been identified by the EIM can be addressed to the initial race data.

✔ 6.5. To what extent can the data be parsed in real time?

Currently, *TFeed* is the only data source that can be parsed while an event is taking place, which means the race control messages cannot be extracted in real time (yet).

FINDINGS AND OUTCOME


17

Results

What does our output look like?

17.1 Generated blog posts 68
17.2 Evaluation 69

Using the fine-tuned MVP model, a set of blog posts has been generated. To get an idea of to what extent these posts can be considered as *accurate* and *entertaining*, evaluation has been done using multiple methods. An overview of a sample of the results is given in section 17.1, followed by the evaluation process in section 17.2.

 The full results can be found in the Drive folder.

17.1 Generated blog posts

For a randomly selected sequence of 10 laps in 10 different races, blog posts were generated. Table 17.1 provides a sample of five of these posts, including the data that they were based on.

Table 17.1: Sample of blog posts generated by the LLM

Input	Output
{'subject': [{'driver': 'GRO'}], 'action': 'stop', 'object': {'main': ['for ultrasoft tyres']}}, {'subject': [{'driver': 'GRO'}], 'action': 'drop to', 'object': {'position': ['11']}}, {'subject': [{'driver': 'VER'}], 'action': 'pit', 'object': {'main': ['for ultrasoft tyres']}}, {'subject': [{'driver': 'VER'}], 'action': 'drop to', 'object': {'position': ['5']}}, {'subject': [{'driver': 'RIC'}], 'action': 'pit', 'object': {}}	"Grosjean pits for ultrasoft tyres, dropping to 11th. Verstappen pits for the same tyres and drops to 5th while Ricciardo pits."
{'subject': [{'driver': 'VER'}], 'action': 'set', 'object': {'main': ['new fastest lap'], 'lap_time': ['97.755']}}	"Verstappen sets a new fastest lap of 97.755s."
{'subject': [{'driver': 'ALO'}], 'action': 'have', 'object': {'main': ['DRS advantage'], 'driver': ['on VET']}}, {'subject': [{'driver': 'VET'}], 'action': 'might lose', 'object': {'main': ['P14'], 'driver': ['to ALO']}}	"Alonso has a DRS advantage on Vettel, who might lose P14 to Alonso."
{'subject': [{'driver': 'BOT'}], 'action': 'pit', 'object': {'main': ['from P19', 'to undercut'], 'driver': ['GAS']}}, {'subject': [{'driver': 'GAS'}], 'action': 'be', 'object': {'main': ['still out']}}	"Bottas pits from P19 to undercut Gasly, who is still out."
{'subject': [{'other': 'fia'}], 'action': 'wave', 'object': {'main': ['double yellow flag'], 'location': ['sector 20']}}, {'subject': [{'other': 'fia'}], 'action': 'wave', 'object': {'main': ['double yellow flag'], 'location': ['sector 19']}}	"The FIA has waved a double yellow flag for sector 20 and sector 19."

For each of the data objects provided to the language model, it was able to produce a blog post in logical English grammar. It generated blogs of around 16 words each, with little under 2 blogs per lap on average.



While going through the BGM output, we have seen that the model was able to produce *Formula One* blog posts, although the actual performance in terms of accuracy and level of entertainment remains unclear. Therefore, we need the evaluation processes discussed in the following section to get a better understanding of these aspects.

17.2 Evaluation

After randomly selecting the lap sequences and generating the posts, manual evaluation of both the EIM and BGM has been performed using Google Sheets. In below subsections, the results of the evaluations are discussed and partially presented.

The full results can be found in the *Evaluation of Accuracy* file in the Drive folder [104]

17.2.1 Event identification

In figure 17.1, the results of evaluating the EIM are shown. Here, the first column (i.e., number of events in data) shows the total number of events that were counted in the data records in the regarding race (such as in the data array of example 17.1). The middle column (i.e., events covered in posts) shows the total number of events mentioned in the *Autosport.com* posts, as well as in the data, together with the percentage of the events covered in both compared to the total number of events in the data (i.e., the left column). The column on the right end (i.e., events not covered in data) shows the number of events that were found in the *Autosport.com* posts, while they were not present in the data.

	Number of events in data	Events covered in posts	Events not covered in data
2018_monaco	16	12 75.00%	11 47.83%
2018_shanghai	26	18 69.23%	10.5 36.84%
2019_singapore	30	21 70.00%	14 40.00%
2019_spain	12	9 75.00%	10 52.63%
2021_zandvoort [1]	15	9 60.00%	3 25.00%
2021_austin	28	16.33 58.32%	2.5 13.28%
2022_saudi-arabia	24	15 62.50%	6.5 30.23%
2022_spa	35	19 54.29%	7 26.92%
2023_baku	9	9 100.00%	3 25.00%
2023_suzuka	18	11 61.11%	4 26.67%

Figure 17.1: Overview of the EIM evaluation results

For the identification of noteworthy events, large variations were seen in the scores between the different races, as shown in the mentioned figure. There can be obtained, however, that the number of events present in the data also contains significant variations, with the number of events of a 10-lap period ranging from 9 (Baku, 2023) up to 35 (Spa, 2022).

Example 17.1: Data array with two events

```
{'subject': [{'driver': 'VET'}], 'action': 'fly', 'object': {'driver': ['RIC'], 'main': ['past']}},
{'subject': [{'driver': 'VET'}], 'action': 'move up', 'object': {'position': ['3']}}, {'subject':
[{'driver': 'LEC'}], 'action': 'pass', 'object': {'driver': ['STR']}}, {'subject': [{'driver':
'LEC'}], 'action': 'move up', 'object': {'position': ['5']}}
```

For those less-eventful lap sequences, an increase in performance for including noteworthy events is seen, as all of the events in the data of Baku 2023 were also seen in the *Autosport.com* [33] blog posts, whereas nearly half of the events of the 35 data records in Spa could not be found in the posts.

This pattern in performance is not obtained for the percentage of the events in the blog posts that are not present in the filtered data. While over 85% of the blog post events were covered in the 28 data events of Austin, 2021, this was only the case for less than a third of the events in Shanghai, which included a similar number of events in the data.

It is important to note, however, that the number of posts of *Autosport.com* and their length changed over the years, with nearly double the amount of posts compared to the data objects in 2018 and 2019, in contrast to a similar number in the years after 2020. This shift in post style is also seen in the right column of figure 17.1, as the percentage of events not covered in the data is higher in the first two years and has seen a large decrease in the remaining three.

The type of events that are not covered in the data are mainly those that are used to 'fill gaps', such as noting the gap or tyre advantage between a pair of drivers, or events that can be seen as conclusions from a prior event, e.g., an undercut being successful or an explanation of a certain gap being caused by strategy or tyre choice.

High-priority events, such as overtakes, are correctly identified by the EIM on most occasions, with less than a handful of mistakes there. Other important events, such as pit stops, are picked up on all occasions, with even more being included in the filtered data than covered in the *Autosport.com* blog posts.

17.2.2 Accuracy

Similar to the evaluation of the EIM, large variations in the performance of the BGM were seen in the results of the different races, as shown in figure 17.2. The average scores for the different dependency types (i.e., subject, action, and object) were similar, although the model seems to score best on parsing the actions from the data, especially when leaving the outlier of Spain 2019 out.

The main causes for a suboptimal score are presented in figure 17.3. As can be obtained from here, most mistakes were related to the model mixing up entities, e.g., swapping the subjects and objects or simply picking the wrong driver name from the unique name code. Such name errors mainly occurred for drivers that only participated in one or two seasons or even only a few races. For instance, drivers Ericsson and Hartley¹ raced until the 2018 season, which also means they were only present in a small part of the training data.

1: These drivers were incorrectly recognized on various occasions in the generated posts of the races in 2018.

Another mistake that was seen often was the incorrect interpretation of the order of drivers. If a driver is approaching another - or is in their DRS zone, it always means that the subject driver is behind the object driver. The model often mistakenly mentioned it the other way around in the generated posts, such as in result 17.A, where the post implies that Bottas is behind, while he is in fact in front of Verstappen.

	Subject	Action	Object	Penalties	Total score	Max score
2018_monaco	13	14	14	2	39	42
	92.86%	100.00%	100.00%	85.71%	92.86%	
2018_shanghai	18.5	19	17.75	1.75	53.5	57
	97.37%	100.00%	93.42%	90.79%	93.86%	
2019_singapore	18	18.66	18.25	3	51.91	57
	94.74%	98.21%	96.05%	84.21%	91.07%	
2019_spain	10.5	9	10	1.5	28	33
	95.45%	81.82%	90.91%	86.36%	84.85%	
2021_zandvoort	13	13.75	14	1.25	39.5	42
	92.86%	98.21%	100.00%	91.07%	94.05%	
2021_austin	17.75	18.08	17.73	1	52.56	57
	93.42%	95.16%	93.32%	94.74%	92.21%	
2022_saudi-arabia	19	19.75	19	1	56.75	60
	95.00%	98.75%	95.00%	95.00%	94.58%	
2022_spa	15.4	15.74	16.65	1.75	46.04	54
	85.56%	87.44%	92.50%	90.28%	85.26%	
2023_baku	9	9	9	0	27	27
	100.00%	100.00%	100.00%	100.00%	100.00%	
2023_suzuka	13.25	13.75	13.33	1	39.33	42
	94.64%	98.21%	95.21%	92.86%	93.64%	

Subject	Action	Object
Avg. score	94.19%	Avg. score
Standard dev.	3.73%	Standard dev.
	6.19%	Avg. score
		Standard dev.
		95.64%
		3.34%

Figure 17.2: Overview of the BGM evaluation results

```
{'subject': [{'driver': 'VER'}], 'action': 'approach', 'object': {'driver': ['BOT']}}, {'subject': [{'other': 'gap'}], 'action': 'be', 'object': {'main': ['only', '2.69'], 'driver': ['BOT'], 'timing': ['this lap']}}
```

"Verstappen is approaching Bottas, who is only 2.69s **behind** this lap."

Result 17.A: Generated post with mix-up of driver order

Overall, the model performed best on generating posts about pit stop actions and overtakes, where it correctly mentioned the tyre compounds, new or old positions, and linked subjects in all cases, except for some name mistakes in the subjects. When there were more than 2 pit actions involved in a single data record, however, it occasionally failed to include the third action in the post or included a driver that was not present in the data, such as in result 17.B. The latter was the case for Verstappen twice, and Hamilton and Sainz once.

```
{'subject': [{'driver': 'PER'}], 'action': 'pit', 'object': {'main': ['for medium tyres']}}, {'subject': [{'driver': 'PER'}], 'action': 'drop to', 'object': {'position': ['3']}}, {'subject': [{'driver': 'LEC'}], 'action': 'pit', 'object': {'main': ['for hard tyres']}}, {'subject': [{'driver': 'LEC'}], 'action': 'drop to', 'object': {'position': ['4']}}, {'subject': [{'driver': 'HAM', 'RAI'}], 'action': 'pit', 'object': {}}
```

"Perez pits for medium tyres, dropping to 3rd. Leclerc pits for hard tyres and drops to 4th. Hamilton, Raikkonen and **Verstappen** also pit."

Result 17.B: Generated post that includes driver that is not present in data

17.2.3 Readers' perception

The questionnaire containing the AI-generated and human-written post sequences was shared via various online platforms², resulting in a total of 24 responses. Each of these respondents rated six sequences of blog posts, resulting in ratings for two sequences per category (HW, AI-O, and AI-R).

To determine whether there are differences in the ratings among the groups³, the *Jarque-Bera* test [108] was first performed on each group to assess whether the data is normally distributed. The results of this test, which can be found in appendix section

2: *Formula One* Facebook groups [105], SurveySwap [106], Instagram stories [107], and Whatsapp groups.

3: Each group involves the ratings of sequences of one category (human-written, non-rephrased AI-generated, or rephrased AI-generated) on one criterion (entertaining, informative, or clear).


	Total score	Most common mistakes
2018_monaco	39	Names not correctly recognized, mix-up of drivers being behind or in front.
	92.86%	
2018_shanghai	53.5	Drivers mistaken for other drivers, mix-up of driver being behind or in front.
	93.86%	
2019_singapore	51.91	Entities missing, incorrect use of 'but', incorrect subject-object links.
	91.07%	
2019_spain	28	Entities missing, mix-up of objects.
	84.85%	
2021_zandvoort	39.5	Subjects missing, mix-up of driver being behind or in front.
	94.05%	
2021_austin	52.56	Names not correctly recognized or mixed up, inclusion of drivers that are not in data.
	92.21%	
2022_saudi-arabia	56.75	Mix-up of entities.
	94.58%	
2022_spa	46.04	Mix-up of subjects and objects, incorrect interpretation of actions.
	85.26%	
2023_baku	27	
	100.00%	
2023_suzuka	39.33	Names not correctly recognized.
	93.64%	

Figure 17.3: Overview of the EIM evaluation results

4: This test is considered as an appropriate method since the variables are ordinal and the observations are independent.

Example 17.2: Hypotheses for the Mann-Whitney U-Tests

- H0:** The ratings on level of entertainment are equal between groups A and B.
HA: The ratings on level of entertainment are not equal between groups A and B.
- H0:** The ratings on level of informativeness are equal between groups A and B.
HA: The ratings on level of informativeness are not equal between groups A and B.
- H0:** The ratings on level of clarity are equal between groups A and B.
HA: The ratings on level of clarity are not equal between groups A and B.

 The full results can be found in the *Evaluation of Level of Entertainment* file in the *Drive* folder [104]

I.2, table I.1, showed that normal distribution could not be assumed for the majority of the groups.

Considering these results, together with the fact that the number of respondents is limited, a non-parametric statistical test was selected to gain insight in the differences among the groups: the Mann-Whitney U-test⁴ [109]. Pairs of groups are compared to one another with a new test each. The corresponding hypotheses are given in example 17.2. Here, groups A and B are all combinations that can be made among the groups for HW, AI-O, and AI-R sequences.

Since two different blog sequences were included in the survey for each of these groups, the results of each sequence all form a separate sub-group. A statistical test was conducted for each pair of sub-groups, for each of the factors. In all cases, the sample size is 24, leading to a critical value of 192 ($p < 0.05$), 164 ($p < 0.01$), or 132 ($p < 0.001$) [110]. In below subsections, the results are discussed per factor.

Entertainment

As shown in the average scores given by the respondents, presented in result 17.C, the average score on entertainment of both rephrased AI-generated sequences is higher than both scores of the other two categories. The non-rephrased sequences scored lowest, with a negative rating on average. To determine the significance of these differences, as mentioned, the Mann-Whitney U-test is applied.

Result 17.C: Average score on level of entertainment per category

HW - 1.167, 1.042
AI-O - -0.208, -0.792
AI-R - 1.375, 1.417

The results of this test on each of the sequences within the same category (e.g., HW 1 versus HW 2) showed that there is not enough evidence to state that there is a difference in the scores, as the critical value was exceeded for all three cases. The same holds for the comparison of the first human-written sequence to both non-rephrased AI-generated ones. For all others, as indicated by the bold numbers in table 17.2, the null hypothesis can be rejected, indicating that there are significant differences between the groups in the corresponding row and column.

	HW 1	HW 2	AI-O 1	AI-O 2	AI-R 1	AI-R 2
HW 1	-	244.5	94.5	37.5	238.5	231
HW 2	-	-	98.5	30	190	188
AI-O 1	-	-	-	202	74	73
AI-O 2	-	-	-	-	29	28
AI-R 1	-	-	-	-	-	276
AI-R 2	-	-	-	-	-	-

Table 17.2: U-values of the Mann-Whitney U-test on level of entertainment

Given the rejection of H_0 for both non-rephrased AI-generated sequences compared to those of the other categories, there can be stated that this category scores significantly lower on perceived level of entertainment.

Informativeness

The average scores on perceived level of informativeness are shown in result 17.D. Compared to the results of the level of entertainment, as discussed in the previous subsection, the scores are closer to each other. This is also noticed in the results of the significance test. While significant differences were obtained for the level of entertainment, the U-values for the perceived level of informativeness showed that there was not enough evidence to state that there were differences between all pairs - except for the second human-written sequence compared to the first non-rephrased AI-generated one, which exactly matched the critical value.

HW - 0.708, 0.500
 AI-O - 0.958, 0.667
 AI-R - 0.958, 0.708

i The ranks can be found in appendix section I.2, table I.2.

Result 17.D: Average score on level of informativeness per category

Clarity

When asked to rate the sequences on how clear they perceive the text, the scores are even closer to each other. Here, the Mann-Whitney U-test lead to the insight that there was not enough evidence to reject the null hypothesis for any of the score pairs: all exceeded the critical value by more than 75, with the average scores - as shown in result 17.E - being close among the groups.

HW - 0.667, 0.750
 AI-O - 0.542, 0.750
 AI-R - 0.667, 0.708

i The ranks can be found in appendix section I.2, table I.3.

Result 17.E: Average score on level of clarity per category

Comments

At the end of the survey, the respondents got the option to leave a comment. They made use of this option on a few occasions. As can be found in the comments shown

in result 17.F, the feedback mainly involved the style of writing (e.g., the number of words used and the choice of terms) and the quality and relatedness of the content.

Result 17.F: Comments provided by respondents in the evaluation survey

"would like to see bit more balance between information and entertainment; feel like the posts were either to 'clean' or used too much words/emotion"

"Some text used the exact same wording, that impacted my ratings."

"Some information related to f1 is inherently unclear or just not accurate e.g. lap times being in seconds and not minute seconds. So 1:33,35 is more clear/accurate opposed to 99,35 seconds. And some messages said there is a yellow flag in sector 9 and 10 but f1 tracks only have 3 sectors. Other than that very interesting concept, good luck!"

"Short and entertaining. Good luck! :)"

"I found the last one(s) with all the more funny terms (bolt of lightning, torpedo, sack of potatoes, cheetah, etc.) somewhat funny, but also subconsciously makes you take it less serious (in my case at least), so I would rather stick to it somewhat more formal than all those terms."



The raw results have lead to new insights in the differences in readers' perception between human-written and AI-generated posts, as well as the differences between the latter before and after deploying the rephrasing model. In the discussion, i.e., the next chapter, we will talk about how we can interpret the results and what impact they have on the study and potential future work in the domain.

In the current chapter, the topics that together form the overall research objective of the thesis, are deliberated upon individually. In each section, an interpretation of the findings, implications, suggestions for future work, and other related aspects, specific to a single topic, are discussed.

18.1 Data	75
18.2 System architecture	76

18.1 Data

The study has highlighted the importance of qualitative data, since missing or incorrect statistics in the race data can lead to the model not being able to generate posts that cover all noteworthy events, while blog examples that miss certain events or contain grammar mistakes lowers the performance of the LLM after fine-tuning. In below subsections, the strengths and limitations of the used data sets, as well as the insights and findings are discussed separately for the race and blog data.

18.1.1 Race data

Exploring and comparing the available data sources for race statistics pointed out that no publicly available API contains the desired information to cover all noteworthy events during a race. Using custom scripts and web scraping techniques, supported by data from various APIs, most of the data desired could be extracted, although some information - such as radio messages, records indicating certain events (e.g., a crash), and specific track locations (e.g., turn 4) - is absent in the composed data set.

This causes the generated blogs to be occasionally incomplete, such as in example 18.1, where no information is included about the cause of the retirement (e.g., a crash or an engine failure), or inaccurate, such as when a driver is mentioned to, e.g., pull a fantastic overtake while they in fact forced their rival off track.

"Russell has quit the race, giving up his Williams car in the middle of this lap."

The race data is, therefore, considered to be a major factor in the flaws of the generated posts, especially those related to the amount of detail and variety. Without background information (e.g., via radio messages or interviews), it is nearly impossible for the model to produce posts that are varying and entertaining and provide sufficient detail, without including any imaginary information. Therefore, it is suggested to, in future work, parse radio messages¹ with speech processing techniques to include such information as well.

Another limitation of the established race data set is that parsing the data in real time is feasible for only the *TFeed* data source [76], although still highly cumbersome. This is due to the ability to access *TFeed* data requiring complex scraping and downloads of large ZIP files. The data from the *Ergast* [5] and *live timing* [6] APIs, on the other hand, can only be accessed after a race has finished, leading to even less data available.

Therefore, it is desired that, for future implementation of the system in an actual real-time fashion, options for possible collaboration with *TFeed*² are explored to get direct access to their data and/or simplify the data collection process. There is no potential solution yet, however, for retrieving the race control (and possibly radio) messages from the *live timing* API in real-time. The paid subscription to the regarding platform might offer more real-time possibilities, although this remains unclear.

Example 18.1: Blog post with an incomplete event

1: The *live timing* API [6] provides access to MP3 files with radio messages, although they are not available until after the race.

2: No contact details of the platform could be found during the thesis, so this would require more research.

18.1.2 Blog data

The quality of the blog post data set was ensured by using the survey that pointed out the clear preference among readers for the posts of *Autosport.com* [33]. Considering that there are ‘only’ around 20 races per year, however, the amount of data is limited. This drawback could especially be noticed in the recognition of driver names for those that did not participate in more than half of the races in the included seasons. Although including more years (e.g., 2012-2018) helps the model in interpreting certain actions (e.g., pit stops or DRS actions) more easily, it is likely that even more driver name mistakes will be made, as more drivers would be included that are currently no longer in *Formula One*.

An alternative option to increase the amount of data, that does not involve the addition of more years, is to add blogs from other platforms, besides those from *Autosport.com*. As these are likely less preferred by potential readers in terms of writing style or events covered, however, it might have negative impact on the style of the generated blogs. This is, therefore, something to consider - and/or experiment with - in future work.

Another issue that was seen during evaluation, which can be linked to the blog data, is the ‘imaginary’ drivers being mentioned. The drivers that were randomly included on one or more occasions involved prominent ones, i.e., drivers that compete for big constructor names and/or won world championships. This can be seen as a *bias* of the model, since these names are simply more often mentioned in human-written posts and, therefore, more significantly present in the training data. A similar phenomena was seen for drivers that often ended up in one of the bottom places, such as Mick Schumacher: his name was incorrectly spelled or mistaken for another name in various generated posts. Such a bias problem is a complicated one to tackle. Manipulating the data to balance the number of mentions of drivers might reduce these biases, although that would decrease the amount of data even further, while certain events (e.g., overtakes) would be reduced excessively as well. To maintain (or even increase) the model’s performance, therefore, it is recommended to accept these biases, especially since they occurred in less than 5% of the generated posts.



Based on what we discussed in section 18.1, we know that both data sets have shown potential, but each come with limitations. Fortunately, by studying and implementing the formulated recommendations, i.e., including blog posts from other sources and exploring possibilities of collaborating with *TFeed*, there is still room for further improvements in the future.

18.2 System architecture

The study has made clear that a successful architecture for automatic text generation based on structured data ideally consists of multiple components that each focus on a specific (set of) task(s). The architecture was divided into three main components: the event identifier, the data-to-blog generator, and the blog rephraser. A discussion of each of these components can be found in the subsections hereafter.

18.2.1 Event identification

The rule-based method applied for identifying events based on the structured race data set resulted in filtered JSON objects, of which the majority of the events³ were also

3: Except for the events that could not be extracted from the input data.

mentioned in the *Autosport.com* blog posts. Furthermore, the EIM failed to cover only a small percentage (i.e., less than a third on average) of the events in those posts.

This implies that having full control of the identified events via the set rules is an appropriate approach for the desired task. The events that the model still failed to identify can easily be added in the future by simply defining additional rules in the EIM script. The model, therefore, still has great potential for future improvements. Below, an overview is given of events for which the model does not include rules yet, but are in theory possible with the current data set, and are seen as candidates - besides the messages discussed in subsection 18.1.1 - for increasing the number of noteworthy events being identified.

- ▶ Tyre (dis)advantage
Compare tyre compound and age between drivers running close.
- ▶ Strategy prediction
Check tyre compound and age to predict when a driver is likely to pit and how many stops they might make in the race.
- ▶ Team performance
Compare teammate performances and combined team performance.
- ▶ Championship prediction
Predict what the championship ranking would be if the drivers finish in the current order.
- ▶ Driver performance
Compare current ranking to previous race results.

Decreasing the number of events in the EIM output that are not necessarily noteworthy (i.e., they are not found in the *Autosport.com* posts), on the other hand, is a challenging task. While some events were seen as 'unnecessary' in some lap sequences, removing the rules to identify those may lead to the model missing out on events that *are* seen as noteworthy in other sequences, which is undesirable. The performance of the EIM in this specific area is, therefore, unlikely to achieve optimal results in the future. Including more events might even result in a lower score here. Having all noteworthy events in the data is, however, prioritized over excluding those that are not⁴. Therefore, this is not directly considered to be a problem in future work.

4: As mentioned in subsection 2.3.1, correct and appropriate content is preferred over nicely-styled text with inaccurate content.

18.2.2 Data-to-blog generation

The choice of a language model specifically pre-trained for a data-to-text generation task has led to blog posts that scored high on interpreting and mentioning the records in the filtered JSON objects. Due to the narrow task of the model, however, the posts lacked some creativity, as they tend to copy the exact terms used in the data, without being capable of understanding and making clear the similarity between, e.g., *making a pit stop* and *coming in for tyres*.

This can be explained by the initial model not being trained to produce such creative or varying content, as well as the fine-tuning data set, in which the terms used in the input-output pairs, especially those representing actions, were the same in many cases - with some manual exception rules for actions such as overtakes - as well. The results of the survey focusing on evaluating readers' perception have shown that this also impacts to what extent the posts are experienced as entertaining, as the generated posts scored significantly lower on level of entertainment.

Although the exact cause remains unclear, this relatively low score can likely be addressed to - among other factors - the lack of variety (in terms used), the 'clean' language, and the lack of emotion, as highlighted in the comments in result 17.F. For tackling this issue, the most obvious solution would be using an LLM that is strong in both data-to-text generation *and* creative text generation, although this turned out to

be a highly challenging task, as they are not (publicly) available and would, therefore, require extensive fine-tuning on a large amount of data. Therefore, the inclusion of an extra component, such as the RPM, has potential to be a solid alternative. A discussion on whether this is indeed a good solution can be found in the next subsection.

5: The rank exactly matched the critical value with a p-value of 0.05

In one specific case, the generated post sequence scored significantly better - although with slightly weak statistical evidence⁵ - on perceived level of informativeness compared to a human-written one. In the other cases, for both informativeness and clarity, no evidence was found that there are differences in the scores. This gives the impression that the generated posts score similar to human-written ones on these aspects, although further experiments would have to be conducted to get full insight in this matter. This could be achieved by, e.g., getting a larger number of respondents, to make the data come closer to a normal distribution, which creates opportunities for conducting other statistical tests.

The most common mistakes made during generation of the posts, i.e., mixing up who is behind or in front and incorrect recognition of driver names from their unique codes, also need to be tackled before the model can be considered as a truly *accurate* NLG component. A recommended option for solving the first mentioned issue, that does not necessarily require fine-tuning on a larger amount of data, involves the event identification process. By simply adding an extra event that indicates the order, such as in example 18.2, the model has the information it needs to interpret who drives behind the other. Unfortunately, such an approach is not ideal. In some cases, more than one of such actions are included in a single filtered JSON object, which might cause new issues with linking the right entities or correct interpretation of the actions.

Example 18.2: Additional object that clarifies the order

```
{'subject': [{'driver': 'BOT'}], 'action': 'catch', 'object': {'driver': ['ALO']}},
{'subject': [{'other': 'gap'}], 'action': 'be', 'object': {'main': ['only', '3.05'], 'driver':
['ALO'], 'timing': ['this lap']}},
{'subject': [{'driver': 'BOT'}], 'action': 'be', 'object': {'driver': ['behind ALO']}}
```

Therefore, the solution may preferably lie in the direction of *reinforcement learning*, where the model receives feedback on its output and learns from the mistakes it made. This same method, of which the use is seen more and more in the field of NLP nowadays [111], may also be beneficial for extracting structured data from the *Autosport.com* blog posts, to potentially replace the LFE approach with an ML-based one.

18.2.3 Blog rephrasing

The results involving the rephrasing component, as discussed in the previous chapter, have shown that adding this model to the system architecture significantly improves the perceived level of entertainment by (potential) readers, compared to the AI-generated posts prior to rephrasing, with strong statistical evidence⁶. In addition, both AI-R post sequences scored significantly higher on perceived entertainment level compared to one of the two human-written sequences, while there was insufficient evidence to state that there is any difference in entertainment score compared to the other human-written sequence. This indicates that the RPM is able to produce blog posts that are as, if not more, entertaining than the human-written ones the initial model was fine-tuned on.

6: The ranks were lower than the critical value for $p < 0.001$ by large margin.

Although it is now clear that the combination of the system's components are capable of coming close to human performance when it comes to writing entertaining text, it remains unclear what specific factors are the main cause of this success. The use of 'funny terms' was mentioned in the survey comments as a factor for increasing perceived entertainment, although this is also noted as a potential negative effect in other areas, as it can impact how serious the updates are taken. To continue on this

matter, it is important to note that the respondents were presented relatively short sequences of blog posts, which may impact how entertaining the text is perceived as well. Therefore, it is suggested to perform evaluation on more - or longer - sequences, potentially even full-race versions, to get a better understanding of the system's performance.

Reflecting on the scores on perceived level of informativeness and clarity, no statistically valid insights were obtained, as no evidence was found to confirm differences in the scores compared to either the HW or AI-O sequences. Similar to what was discussed in the previous subsection, gathering more respondents opens doors to other tests to get more qualitative insights in these factors.



Similar to the data-related discussion, the implementation of the overall system architecture has shown potential, although it also requires further optimization to achieve convincingly *accurate* and *entertaining* blog posts. We consider applying reinforcement learning to limit interpretation flaws and exploring the use of ML-based techniques to replace the LFE method as the top candidates for future improvements of the system.

During the course of the thesis, there was one central objective, i.e., the main research question. The study aimed to answer "How can accurate and entertaining live blog posts be generated based on structured, *Formula One* race data?", which was broken down into various sub-questions, each answered in the recaps at the end of the thesis parts.

The proposed system architecture showcased the potential for data-driven linguistic content in the car racing domain. The study showed that it is, in practice, feasible to automatically generate *Formula One* blog posts based on structured race data, in which the most noteworthy events are represented. This was achieved by a multi-model system architecture, with a rule-based model for event identification, a fine-tuned *Transformer*-based LLM - pre-trained for data-to-text generation - for generating the posts, and a second LLM for rephrasing the content in a more entertaining fashion.

The rule-based method proved to be a suitable approach with good results in identifying events. Furthermore, the first LLM showed good performance in interpreting the data and generating informative blog posts with limited occurrence of imaginary information, while the second one proved to improve the perceived level of entertainment of the posts.

Especially in terms of level of detail and factuality, however, the generated posts do not yet meet the quality of human-written *Formula One* live blogs. This part is mainly linked to the race data, in combination with the event identification component: certain highly noteworthy events can simply not be identified by the model based on the used data set.

These insights show that the proposed system architecture could, potentially, be an even better match for other domains. It could be a particularly good fit for those that have a larger amount of data available and/or less external factors impacting the events discussed. Examples of such are - among others - financial markets (updates on stock prices and trends) and transportation (real-time traffic updates). In these domains, external factors (such as incident causes in *Formula One*) have less impact and are, therefore, less crucial for readers to know, while the offer of publicly available data is large.

To better fit the current domain, i.e., car racing, or other sports disciplines, especially for generating engaging and entertaining content, various further steps need to be taken. Suggestions include addition of an extra component, i.e., a web/social media scraper to find what people are discussing about the race online (which might give background information about certain events) or a speech processing model to include radio messages, interviews, and/or commentary.

The study itself came with a few limitations. First of all, since the study was limited to publicly available data, the options for race data were limited, which restricted the possibility of applying the generation process in real time. Furthermore, only a limited number of LLMs were used during experimentation for structured data extraction. Considering this, a different (type of) LLM might still outperform the proposed LFE method for establishing the fine-tuning data set, which was not validated due to time restrictions.

Despite the fact that the overall system did not succeed in flawless live blog generation, the work can be seen as a great contribution to the field of NLG, especially to research in data-driven text generation. It has highlighted both the strengths and limitations of

state-of-the-art generative models, as well as decomposed the overall generation task in components that are each responsible for a specific sub-task. This decomposition makes it possible to improve such an architecture step by step, and easily identify the 'pain points'.



The multi-model architecture with a separate EIM, BGM, and RPM succeeded in generating informative blog posts based on the initial, structured input data. The RPM significantly improves the perceived level of entertainment of the posts.



Recommended future work steps include using reinforcement learning for improving the data-to-text generation step, using more blog data for fine-tuning, and further studying the option of using an LLM for extracting structured data from the example posts.



A framework for a data-driven generation system was proposed, which offers opportunities for adapting similar system architectures in the field.



No statistical evidence was found that the system succeeded in generating blog posts that have a similar level of entertainment compared to human-written ones. Furthermore, real-time implementation has not been achieved due to the limited availability of race data sources.



The current system architecture has great potential in domains where less details are needed and less external factors have impact on the events, such as in finance and transportation.



Research question recap

Part VI - Findings and Outcome

✓ RQ2. What is needed for the generated posts to be *accurate*?

✓ 2.3. To what extent can the accuracy of human-written blog posts be matched?

The generated posts cover approximately 70% of the events in human-written posts on average, while 70% of the events in the generated posts are also in the human-written ones. The model is able to correctly interpret and process approximately 95% of the subjects, actions, and objects in its generated posts.

✓ RQ3. What is needed for the generated posts to be *entertaining*?

✓ 3.5. To what extent can the level of entertainment of human-written blog posts be matched?

No significant evidence was found to state that there is a difference in perceived entertainment between the rephrased, AI-generated posts and the human-written ones. Further experiments are needed to draw a conclusion about this matter.

i The results of the entertainment evaluation survey are currently still in progress.

Bibliography

- [1] Hans Erik Næss and Simon Chadwick. *The Future of Motorsports: Business, Politics and Society*. Taylor & Francis, 2023 (cited on page 2).
- [2] Maury Brown. *Inside The Numbers That Show Formula 1's Popularity And Financial Growth*. Retrieved on 2023-07-10. 2023. URL: <https://www.forbes.com/sites/maurybrown/2023/03/29/inside-the-numbers-that-show-formula-1s-popularity-and-financial-growth/> (cited on page 2).
- [3] The New York Times. *Take a Look at Formula 1 Now*. Retrieved on 2023-07-10. 2022. URL: <https://www.nytimes.com/2022/04/22/sports/autoracing/formula-1-new-races-popularity.html> (cited on page 2).
- [4] Tom Bedford. *How to watch F1 online: stream every Formula 1 2023 race live from around the world*. Retrieved on 2023-07-10. 2023 (cited on page 2).
- [5] Ergast.com. *Ergast Developer API*. Retrieved on 2023-04-26. n.d. URL: <http://ergast.com/mrd/> (cited on pages 2, 13, 14, 21, 24, 27, 75).
- [6] Formula 1. *Formula 1 Live timing*. Retrieved on 2023-05-09. n.d. URL: <https://livetiming.formula1.com> (cited on pages 2, 21, 23, 27, 75).
- [7] Yair Galily. 'Artificial intelligence and sports journalism: Is it a sweeping change?' In: *Technology in society* 54 (2018), pp. 47–51 (cited on pages 2, 12).
- [8] Noam Lemelshtrich Latar. *Robot journalism: Can human journalism survive?* World Scientific, 2018 (cited on pages 2, 12).
- [9] Chenhe Dong et al. 'A survey of natural language generation'. In: *ACM Computing Surveys* 55.8 (2022), pp. 1–38 (cited on page 2).
- [10] Albert Gatt and Emiel Kraahmer. 'Survey of the state of the art in natural language generation: Core tasks, applications and evaluation'. In: *Journal of Artificial Intelligence Research* 61 (2018), pp. 65–170 (cited on pages 2, 121, 122).
- [11] Cécile L Paris, William R Swartout, and William C Mann. *Natural language generation in artificial intelligence and computational linguistics*. Vol. 119. Springer Science & Business Media, 2013 (cited on pages 2, 6).
- [12] Andreas Graefe et al. 'Readers' perception of computer-generated news: Credibility, expertise, and readability'. In: *Journalism* 19.5 (2018), pp. 595–610 (cited on pages 2, 8).
- [13] A van Dalen. 'The algorithms behind the headlines'. In: *Journalism Practice* 6.5-6 (2012), pp. 648–658 (cited on page 2).
- [14] S Levy. *Can an algorithm write a better news story than a human reporter?* 2012. URL: <http://www.wired.com/2012/04/can-an-algorithm-write-a-better-news-story-than-a-human-reporter/> (cited on page 2).
- [15] M Carlson. 'The robotic reporter'. In: *Digital Journalism* 3.3 (2015), pp. 416–431 (cited on page 2).
- [16] F1-History.org. *The History of Formula 1*. Retrieved on 2023-09-06. n.d. URL: <https://www.f1-history.org/> (cited on pages 4, 5).
- [17] Adrianna Morganelli. *Formula One*. Crabtree Publishing Company, 2006 (cited on page 4).
- [18] Formula1.com. *Formula 1 announces TV, race attendance and digital audience figures for 2021*. Retrieved on 2023-09-06. 2022. URL: <https://www.formula1.com/en/latest/article.formula-1-announces-tv-race-attendance-and-digital-audience-figures-for-2021.1YDpVJIOHGnuok907sWcKW.html> (cited on page 4).

- [19] Victor R. Lopez. *What is so special and good about Formula 1 nowadays?* Retrieved on 2023-09-06. 2022. URL: <https://www.f1-fansite.com/f1-column/what-is-so-special-about-formula-1/> (cited on page 4).
- [20] Formula1.com. *Stream F1 your way*. Retrieved on 2023-09-06. n.d. URL: <https://www.formula1.com/en-nl/subscribe-to-f1-tv> (cited on page 4).
- [21] Oliver Burkeman. *Forty years of the internet: how the world changed for ever*. Retrieved on 2023-09-06. 2009. URL: <https://www.theguardian.com/technology/2009/oct/23/internet-40-history-arpanet> (cited on page 5).
- [22] Shiona McCallum. *Elon Musk: Twitter rebrands as X and kills off blue bird logo*. Retrieved on 2023-11-19. 2023. URL: <https://www.bbc.com/news/business-66284304> (cited on page 5).
- [23] Akshay Java et al. 'Why we twitter: understanding microblogging usage and communities'. In: *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*. 2007, pp. 56–65 (cited on page 5).
- [24] DAC Group. *The Rise of Twitter: Its Success and Impact*. Retrieved on 2023-09-06. 2013. URL: <https://www.dacgroup.com/en-gb/blog/the-rise-of-twitter-a-look-at-its-success-and-impact/> (cited on page 5).
- [25] James Kelm and Jameel Gbajabiamila. *Introducing Twitter for Professionals*. Retrieved on 2023-09-06. n.d. URL: <https://business.twitter.com/en/blog/twitter-for-professionals.html> (cited on page 5).
- [26] Einar Thorsen and Daniel Jackson. 'Seven characteristics defining online news formats: Towards a typology of online news and live blogs'. In: *Digital Journalism* 6.7 (2018), pp. 847–868 (cited on page 5).
- [27] Jetske Bonenkamp. 'Automatic Generation of Live Update Posts During Formula One Races - Research Topics'. In: (2023) (cited on pages 5, 7, 12, 91, 121).
- [28] BBC Sport. *Sport Live Guide*. Retrieved on 2023-09-06. 2023. URL: <https://www.bbc.com/sport/live-guide> (cited on page 5).
- [29] EventingHQ. *Europees Kampioenschap Eventing Ponies 2021*. Retrieved on 2023-09-07. 2021. URL: <https://eventinghq.nl/liveblog-europees-kampioenschap-eventing-ponies/> (cited on page 5).
- [30] Motorsport.com. *F1 Liveblog*. Retrieved on 2023-05-18. n.d. URL: <https://nl.motorsport.com/live/> (cited on page 5).
- [31] RacingNews265. *Het laatste Formule 1 en Max Verstappen nieuws*. Retrieved on 2023-05-06. n.d. URL: <https://racingnews365.nl/> (cited on pages 5, 17).
- [32] Crash.net. *First for all the latest F1 news, views, results, pictures and more...* Retrieved on 2023-04-26. n.d. URL: <https://www.crash.net/f1> (cited on pages 5, 14).
- [33] Autosport.com. *F1 live text commentaries*. Retrieved on 2023-06-09. n.d. URL: <https://www.autosport.com/live/> (cited on pages 5, 17, 18, 27, 35, 45, 62, 64, 65, 70, 76).
- [34] RaceFans.net. *How to watch F1 live around the world*. Retrieved on 2023-11-19. n.d. URL: <https://www.racefans.net/f1-information/f1-faq/watch-f1-around-world/> (cited on page 5).
- [35] Karen Sparck Jones. 'Natural language processing: a historical review'. In: *Current issues in computational linguistics: in honour of Don Walker* (1994), pp. 3–16 (cited on page 6).
- [36] Robert Dale. 'Natural language generation: The commercial state of the art in 2020'. In: *Natural Language Engineering* 26.4 (2020), pp. 481–487 (cited on page 6).
- [37] Diksha Khurana et al. 'Natural language processing: State of the art, current trends and challenges'. In: *Multimedia tools and applications* 82.3 (2023), pp. 3713–3744 (cited on page 6).

- [38] Marco Fonseca et al. 'Innovative approach for engineering NLG systems: the content determination case study'. In: *Computational Linguistics and Intelligent Text Processing: 9th International Conference, CICLing 2008, Haifa, Israel, February 17-23, 2008. Proceedings 9*. Springer. 2008, pp. 478–487 (cited on pages 6, 7).
- [39] Somayajulu Sripada et al. 'A Two-Stage Model For Content Determination'. In: *Proceedings of the ACL 2001 Eighth European Workshop on Natural Language Generation (EWNLG)*. 2001 (cited on page 6).
- [40] Ehud Reiter and Robert Dale. 'Building applied natural language generation systems'. In: *Natural Language Engineering* 3.1 (1997), pp. 57–87 (cited on pages 6, 121).
- [41] James F Allen and C Raymond Perrault. 'Analyzing intention in utterances'. In: *Artificial intelligence* 15.3 (1980), pp. 143–178 (cited on page 6).
- [42] Farhood Farahnak et al. 'Surface realization using pretrained language models'. In: *Proceedings of the Third Workshop on Multilingual Surface Realisation*. 2020, pp. 57–63 (cited on page 7).
- [43] Dileep Pasumarthi. 'Natural Language Generation from Structured Data'. In: *Analytics Vidhya* (2020) (cited on page 7).
- [44] Anthony Gillioz et al. 'Overview of the Transformer-based Models for NLP Tasks'. In: *2020 15th Conference on Computer Science and Information Systems (FedCSIS)*. IEEE. 2020, pp. 179–183 (cited on pages 7, 122, 123).
- [45] Google Developers. *Introduction to Large Language Models*. Retrieved on 2023-11-13. 2023. URL: <https://developers.google.com/machine-learning/resources/intro-llms> (cited on page 7).
- [46] Muhammad Usman Hadi et al. 'A survey on large language models: Applications, challenges, limitations, and practical usage'. In: *TechRxiv* (2023) (cited on page 7).
- [47] Janna Lipenkova. 'Choosing the right language model for your NLP use case'. In: *Towards Data Science* (2023) (cited on page 7).
- [48] OpenAI. *GPT-4 is OpenAI's most advanced system, producing safer and more useful responses*. Retrieved on 2023-07-02. URL: <https://openai.com/gpt-4> (cited on page 7).
- [49] Tomas Vykruta. *Understanding Causal LLM's, Masked LLM's, and Seq2Seq*. Retrieved on 2023-11-19. 2023. URL: https://medium.com/@tom_21755/understanding-causal-llms-masked-llm-s-and-seq2seq-a-guide-to-language-model-training-d4457bbd07fa (cited on pages 7, 8).
- [50] Susan Zhang et al. 'Opt: Open pre-trained transformer language models'. In: *arXiv preprint arXiv:2205.01068* (2022) (cited on page 8).
- [51] Weizhen Qi et al. 'Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training'. In: *arXiv preprint arXiv:2001.04063* (2020) (cited on page 8).
- [52] Philip Resnik and Jimmy Lin. 'Evaluation of NLP systems'. In: *The handbook of computational linguistics and natural language processing* (2010), pp. 271–295 (cited on page 8).
- [53] C Clerwall. 'Enter the robot journalist'. In: *Journalism Practice* 8.5 (2014), pp. 519–531 (cited on page 8).
- [54] Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 'Evaluation of text generation: A survey'. In: *arXiv preprint arXiv:2006.14799* (2020) (cited on page 8).
- [55] Bethany Cadman. *What makes a good story?* Retrieved on 2023-09-12. n.d. URL: <https://writerslife.org/makes-good-story/> (cited on page 9).

- [56] Zubair Hamad Muhi. 'The Impact Of Blogging On Efl Students' Writing Development'. In: *English Learning Innovation (englie)* 4.2 (2023), pp. 135–149 (cited on page 9).
- [57] Ying Lin. *Blogging statistics you need to know*. Retrieved on 2023-09-15. 2022. URL: <https://www.oberlo.com/blog/blogging-statistics> (cited on page 9).
- [58] Lindsay Kramer. *Our 8-Step Guide for How to Write a Pro Blog Post*. Retrieved on 2023-09-15. 2021. URL: <https://www.grammarly.com/blog/how-to-write-a-blog/> (cited on page 9).
- [59] American Press Institute. *What makes a good story?* Retrieved on 2023-09-12. n.d. URL: <https://americanpressinstitute.org/journalism-essentials/makes-good-story/> (cited on page 9).
- [60] Angela M Lee and Hsiang Iris Chyi. 'When Newsworthy is Not Noteworthy: Examining the value of news from the audience's perspective'. In: *Journalism studies* 15.6 (2014), pp. 807–820 (cited on page 9).
- [61] Tony Rogers. 'What Makes a Story Newsworthy'. In: *ThoughtCo* (2019) (cited on page 9).
- [62] David Crystal. *Language and the Internet*. Cambridge University Press, 2001 (cited on page 10).
- [63] Marcin Lewandowski. 'The language of online sports commentary in a comparative perspective'. In: *Lingua posnaniensis* 54.1 (2012), p. 65 (cited on page 10).
- [64] Mohsen Ghadessy. *Registers of Written English. Situational Factors and Linguistic Features*. Pinter Publishers, 1988 (cited on page 10).
- [65] Charles A. Ferguson. 'Dialect, Register, and Genre: Working Assumptions About Conventionalization'. In: *Biber & Finnegan* (1983), pp. 15–30 (cited on page 10).
- [66] Janet Holmes. *An Introduction to Sociolinguistics*. Pearson Education, 2001 (cited on page 10).
- [67] Adrian Beard. *The Language of Sport*. Routledge, 1998 (cited on page 10).
- [68] Jim Albert, Jay Bennett, and James J Cochran. *Anthology of statistics in sports*. SIAM, 2005 (cited on page 12).
- [69] Henry Wang, Saman Sarraf, and Arbi Tamrazian. 'Sports narrative enhancement with natural language generation'. In: *Sports Analytics Conference* (2022) (cited on page 12).
- [70] Chris van der Lee, Emiel Kraemer, and Sander Wubben. 'PASS: A Dutch data-to-text system for soccer, targeted towards specific audiences'. In: *Proceedings of the 10th International Conference on Natural Language Generation*. 2017, pp. 95–104 (cited on page 12).
- [71] Junpeng Gong, Wen Ren, and Pengzhou Zhang. 'An automatic generation method of sports news based on knowledge rules'. In: *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*. IEEE. 2017, pp. 499–502 (cited on page 12).
- [72] João Pinto Barbosa Machado Aires. 'Automatic generation of sports news'. In: (2016) (cited on page 12).
- [73] Tatsuya Ishigaki et al. 'Generating Racing Game Commentary from Vision, Language, and Structured Data'. In: *Proceedings of the 14th International Conference on Natural Language Generation*. 2021, pp. 103–113 (cited on page 13).
- [74] TA Krijnen. 'Automatic generation of Formula 1 feports'. B.S. thesis. University of Twente, 2021 (cited on pages 13, 14).
- [75] Planet F1. *Planet F1 Live*. Retrieved on 2023-10-14. n.d. URL: <https://live.planetf1.com/> (cited on page 17).

- [76] F1 TFeed. *Formula 1 Live Timing and Circuit Information*. Retrieved on 2023-10-23. n.d. URL: <https://f1.tfeed.net/> (cited on pages 21–23, 27, 75).
- [77] Jetske Bonenkamp. *Automatic Generation of Live Blog Posts during Formula One races*. Retrieved on 2023-11-27. 2023. URL: <https://github.com/jetskebonenkamp/f1-blog-generation> (cited on pages 21, 22, 46, 53, 58, 60, 136).
- [78] Scikit Learn. *Novelty and Outlier Detection*. Retrieved on 2023-11-09. n.d. URL: https://scikit-learn.org/stable/modules/outlier_detection.html (cited on page 31).
- [79] Mahbub Alam. ‘Z-score for Anomaly Detection’. In: *Towards Data Science* (2020) (cited on page 31).
- [80] Mahbub Alam. ‘Support Vector Machine (SVM) for Anomaly Detection’. In: *Towards Data Science* (2020) (cited on page 31).
- [81] Yke Rusticus. *How to Extract Structured Data from Unstructured Text using LLMs*. Retrieved on 2023-10-25. 2023. URL: <https://xebia.com/blog/archetype-llm-batch-use-case/> (cited on pages 33, 34).
- [82] Pydantic Services Inc. *Pydantic*. Retrieved on 2023-10-25. n.d. URL: <https://pydantic.dev/> (cited on page 33).
- [83] Steven Pinker. ‘Rules of language’. In: *Science* 253.5019 (1991), pp. 530–535 (cited on page 34).
- [84] Abid All Awan. *What is Named Entity Recognition? Methods, use cases, and challenges*. Retrieved on 2023-10-21. 2023. URL: <https://www.datacamp.com/blog/what-is-named-entity-recognition-ner> (cited on page 34).
- [85] Prashant Sharma. ‘Dependency Parsing in Natural Language Processing’. In: (2023) (cited on page 35).
- [86] Ruslan Mitkov et al. ‘Coreference resolution: To what extent does it help NLP applications?’ In: *Text, Speech and Dialogue: 15th International Conference, TSD 2012, Brno, Czech Republic, September 3-7, 2012. Proceedings 15*. Springer. 2012, pp. 16–27 (cited on page 36).
- [87] Pawel Mielniczuk. ‘Intro to coreference resolution in NLP’. In: *Towards Data Science* (2021) (cited on page 36).
- [88] Marta Maslankowska. ‘Most popular coreference resolution frameworks’. In: *Towards Data Science* (2021) (cited on page 36).
- [89] spaCy. *Linguistic Features*. Retrieved on 2023-10-26. n.d. URL: <https://spacy.io/usage/linguistic-features> (cited on pages 36, 58).
- [90] Arvind Padmanabhan. *Semantic Role Labelling*. Retrieved on 2023-11-05. 2020. URL: <https://devopedia.org/semantic-role-labelling> (cited on page 36).
- [91] AllenNLP. *Semantic Role Labeling - Demo*. Retrieved on 2023-11-05. n.d. URL: <https://demo.allennlp.org/semantic-role-labeling> (cited on pages 36, 50, 51).
- [92] Tianyi Tang et al. ‘Mvp: Multi-task supervised pre-training for natural language generation’. In: (2022) (cited on page 37).
- [93] RUCAIBox. *MVP-data-to-text*. Retrieved on 2023-11-20. 2022 (cited on page 37).
- [94] Yilun Zhao et al. ‘Large Language Models are Effective Table-to-Text Generators, Evaluators, and Feedback Providers’. In: *arXiv preprint arXiv:2305.14987* (2023) (cited on page 38).
- [95] Hugging Face. *The AI community building the future*. Retrieved on 2023-10-24. n.d. URL: <https://huggingface.co/> (cited on pages 39, 52, 55, 60).
- [96] Guan Wang et al. ‘Openchat: Advancing open-source language models with mixed-quality data’. In: *arXiv preprint arXiv:2309.11235* (2023) (cited on page 39).

- [97] Lianmin Zheng et al. 'Judging LLM-as-a-judge with MT-Bench and Chatbot Arena'. In: *arXiv preprint arXiv:2306.05685* (2023) (cited on page 39).
- [98] Guan Wang et al. *OpenChat*. Retrieved on 2023-12-14. 2023. URL: https://huggingface.co/openchat/openchat_3.5 (cited on pages 39, 60).
- [99] Benjamin Marle. *Introduction to the Open LLM Falcon-40B: Performance, Training Data, and Architecture*. Retrieved on 2023-09-15. 2023. URL: <https://towardsdatascience.com/introduction-to-the-open-llm-falcon-40b-performance-training-data-and-architecture-98388fa40226> (cited on pages 48, 130).
- [100] University of Twente. *JupyterLab*. Retrieved on 2023-12-09. n.d. URL: <https://www.utwente.nl/en/service-portal/research-support/it-facilities-for-research/jupyterlab> (cited on pages 52, 60).
- [101] Yufan Liu. 'Explore parameter efficient fine-tuning methods on Large Language Model'. In: (2023) (cited on page 52).
- [102] Edward J Hu et al. 'Lora: Low-rank adaptation of large language models'. In: *arXiv preprint arXiv:2106.09685* (2021) (cited on page 52).
- [103] F. Pansham. *What rank (r) and alpha to use in LoRA in LLM fine-tuning?* Retrieved on 2023-11-29. 2023. URL: <https://medium.com/@fartypantsham/what-rank-r-and-alpha-to-use-in-lora-in-llm-1b4f025fd133> (cited on page 60).
- [104] Jetske Bonenkamp. *F1 Blog Generation - Google Drive*. Access via University of Twente only. 2023. URL: <https://drive.google.com/drive/folders/1M5r9tp92fAVhVKfiYqQdgx-P3H-3P0Ts?usp=sharing> (cited on pages 69, 72).
- [105] Steve Kirk and Harry Kirk. *Formula 1 Fans Group*. Retrieved on 2024-01-02. 2020. URL: <https://www.facebook.com/groups/formulaonefans> (cited on page 71).
- [106] SurveySwap. *Find survey respondents now*. Retrieved on 2024-01-03. n.d. URL: <https://surveyswap.io/> (cited on page 71).
- [107] Instagram. *Stories*. Retrieved on 2023-01-03. n.d. URL: <https://help.instagram.com/1660923094227526> (cited on page 71).
- [108] Zach. *How to Perform a Normality Test in Google Sheets*. Retrieved on 2024-01-02. 2022. URL: <https://www.statology.org/normality-test-google-sheets/> (cited on page 71).
- [109] Zach. *Mann-Whitney U-Test*. Retrieved on 2024-01-02. 2018. URL: <https://www.statology.org/mann-whitney-u-test/> (cited on page 72).
- [110] Anatol Stefanowitsch. 'Critical values for the Mann-Whitney-text'. In: *Freie Universitat Berlin* (). Retrieved on 2024-01-02 (cited on page 72).
- [111] Victor Uc-Cetina et al. 'Survey on reinforcement learning for language processing'. In: *Artificial Intelligence Review* 56.2 (2023), pp. 1543–1575 (cited on page 78).
- [112] Ehud Reiter. 'NLG vs. templates'. In: *arXiv preprint cmp-lg/9504013* (1995) (cited on page 121).
- [113] Ratish Puduppully, Li Dong, and Mirella Lapata. 'Data-to-text generation with content selection and planning'. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 6908–6915 (cited on page 121).
- [114] Kees Van Deemter, Mariët Theune, and Emiel Kraahmer. 'Real versus template-based natural language generation: A false opposition?' In: *Computational linguistics* 31.1 (2005), pp. 15–24 (cited on pages 121, 122).
- [115] Abelardo Vieira Mota, Ticiana Linhares Coelho da Silva, and José Antônio Fernandes De Macêdo. 'Template-Based Multi-solution Approach for Data-to-Text Generation'. In: *Advances in Databases and Information Systems*. Ed. by Jérôme Darmont, Boris Novikov, and Robert Wrembel. Springer International Publishing, 2020 (cited on pages 121, 122).

- [116] Richard E Fikes and Nils J Nilsson. 'STRIPS: A new approach to the application of theorem proving to problem solving'. In: *Artificial intelligence 2.3-4* (1971), pp. 189–208 (cited on page 122).
- [117] Erkut Erdem et al. 'Neural natural language generation: A survey on multi-linguality, multimodality, controllability and learning'. In: *Journal of Artificial Intelligence Research* 73 (2022), pp. 1131–1207 (cited on page 122).
- [118] Karen Kukich. 'Where do phrases come from: Some preliminary experiments in connectionist phrase generation'. In: *Natural language generation: New results in artificial intelligence, psychology and linguistics* (1987), pp. 405–421 (cited on page 122).
- [119] Touseef Iqbal and Shaima Qureshi. 'The survey: Text generation models in deep learning'. In: *Journal of King Saud University-Computer and Information Sciences* 34.6 (2022), pp. 2515–2528 (cited on page 122).
- [120] Hanqing Zhang et al. 'A survey of controllable text generation using transformer-based pre-trained language models'. In: *arXiv preprint arXiv:2201.05337* (2022) (cited on page 123).
- [121] Zhengbao Jiang et al. 'How can we know what language models know?' In: *Transactions of the Association for Computational Linguistics* 8 (2020), pp. 423–438 (cited on page 123).
- [122] Enkelejda Kasneci et al. 'ChatGPT for good? On opportunities and challenges of large language models for education'. In: *Learning and Individual Differences* 103 (2023), p. 102274 (cited on page 123).
- [123] Hugging Face. *Llama 7B Chat HF*. Retrieved on 2023-09-22. 2023 (cited on page 127).
- [124] Arjun Sha. *12 Best Large Language Models (LLMs) in 2023*. Retrieved on 2023-09-15. 2023. URL: <https://beebom.com/best-large-language-models-llms/> (cited on page 130).
- [125] Arjun Sha. *How to Try Out Google's PaLM 2 AI Model Right Now*. Retrieved on 2023-09-15. 2023. URL: <https://beebom.com/how-use-google-palm-2-ai-model/> (cited on page 130).
- [126] Arjun Sha. *How to Use ChatGPT Rival "Claude" Right Now*. Retrieved on 2023-09-15. 2023. URL: <https://beebom.com/how-use-anthropic-claude-chatgpt-rival/> (cited on page 130).
- [127] Stanford HELM. *Core scenarios*. Retrieved on 2023-09-15. 2023. URL: https://crfm.stanford.edu/helm/latest/?group=core_scenarios (cited on page 130).
- [128] Bhathiya Bandara. *Falcon 40B: Redefining the Limits of Open-Source LLMs*. Retrieved on 2023-09-15. 2023. URL: <https://medium.com/aimonks/falcon-40b-redefining-the-limits-of-open-source-llms-e96faa907192> (cited on page 130).

APPENDIX


A

Reader preference survey

A.1 Introduction

To get insight in what potential readers want to read, what information they find relevant, what writing style, what text length, and other factors they prefer, an online survey is used. The survey has been spread on various platforms, such as via racing forums and social media. It is composed in a way that the results are anonymous, but the respondents are asked for some personal information that is potentially relevant, such as preferred team(s) or driver(s) and country of living. The respondents have been asked to answer questions regarding how often they watch *Formula One* races, whether they follow news or blogs about the sport, what platforms they use, and more.

The survey is split up into different sections, each section representing a new subject. In below subsections, a global description of each section and the corresponding purpose are discussed.

 The survey introduction is reused material from the *Research Topics* report [27].

A.1.1 Personal information

As mentioned, the survey responses are kept fully anonymous. Therefore, the respondent is not asked to fill in their name and/or specific details such as address. However, they are asked for their country of living and their age group (in ranges of 5 years), as these are factors potentially that are correlated with certain answers. Despite this information, the results are considered anonymous as these two factors solely cannot lead back to a specific individual. Other personal information such as gender is left out as this is not regarded as relevant information here.

A.1.2 Teams, drivers, and engagement

Within this section, the questions aim at getting a better understanding of the factors impacting the respondent's choices. By asking them about their team(s) and driver(s) preference, insight is gained in what might bias the answers to further questions. In addition, retrieving information about the time they spent watching races and/or reading news, blogs, or statistics via online platforms gives insight into how engaged the respondent is into the sport, which can also have influence on answers later on in the survey. Overall, the section globally sketches the profile of the respondent.

A.1.3 Platforms

As a follow-up on the previous section, information is collected that involves the platform(s) the respondent currently uses and what type of content they read or watch there. This can indirectly give insight in what type of content (i.e. statistics, news, or blogs), writing style, and overall layout they prefer, which can also be used to validate the answers to questions later in the survey.

A.1.4 Examples

The respondent will be presented some examples of blog posts and will be asked to rate - either a single post or a sequence of posts - those on a scale from one to ten on (1) the information it contains, (2) the writing style, and (3) the likelihood that they will read such posts. The intention is that the presented posts are varying in content and style.

A.2 Setup

Due to the length of the survey, in combination with the fact that the questions and potential answers are already included in the overview of the results (to be found on pages 94 to 120), no separate overview of the setup of the survey is included in this document.

A.3 Results

On pages 94 to 120, the results of the survey are included. A total of 24 individuals filled out the questionnaire. As can be obtained from the results, the majority of the respondents have the Dutch nationality. Furthermore, all age groups below 70 are represented, although most respondents are between the age of 20 and 25. The participants rarely attend *Formula One* races, but they watch (nearly) every race live, and can therefore all be considered to meet the target group requirement of being a 'regular race follower'.

The answers to introductory questions involving the reading consumption of *Formula One* news and blog items varies among the respondents: only a single participant follows live blogs during every race, while all others do so every now and a quarter of the respondents claim to never follow such blogs. These live blogs are, however, visited *afterwards* regularly by more than half of the respondents. Compared to live blogs, pages with online statistics or news articles are more commonly visited by the respondents, as the majority reads those on a regular basis.

The most preferred driver and team is likely biased by the fact that the Dutch nationality is highly represented in the survey results. These results can, however, still be of use in linking certain patterns in blog preferences to driver preferences.

Looking at the comments the respondents included about their choice of platform(s) for following the races visually, pricing and the *language* of the commentary seems to be the main drive factor. The style and content of the audio commentary is not explicitly mentioned.

Besides the platforms used for watching the actual races, the participants were asked to provide information about their current use of platforms for textual updates and statistics. The two most popular platforms mentioned are the platform of *Formula One* itself and *RacingNews365*. The latter might be slightly biased due to this platform being available in Dutch, while some of the others only spread English content. The reasons behind the choice of news platform(s) vary: while some note the quality of the articles, familiarity, or accessibility, others prioritize platforms that are up-to-date, reliable, and/or contain technical (background) information. A platform that was not among the default options but is mentioned by multiple participants is the platform X (previously *Twitter*), of which they mention that it is "the fastest platform to share news during a live race".

Although the example blogs were presented to the participants in groups of three blog sequences of three different platforms during the same race, they are included in the results per platform for clarity.

RacingNews365

Based on the blog sequences of all three races, the respondents indicate that the platform should give more frequent updates and cover more events - especially certain details such as positions and/or interval times, while it in some cases uses too much expression and has slight bias in the posts. The use of short titles that directly indicate what is going on, however, is seen as a good quality some respondents. Although it is not the most popular platform out of the three, approximately half of the respondents would be open to using this platform to read their live blogs during a race. The individuals that would explicitly not choose this platform mention that the focus is too much on the front field as one of the reasons.

PlanetF1

Among the three different races represented by the blog sequences, there are relatively large differences in ratings. While the sequence of the Bahrain Grand Prix is seen as too limited in length, detail, and events covered, the other two sequences contained, according to the respondents, too long posts that were not given frequently enough. The writing style is also mentioned in the comments, as this is considered to be 'hard to read easy and quick'. Here, details about drivers and their positions are also missed by the participants. Only a quarter of the participants indicate that they would use this platform for textual updates.

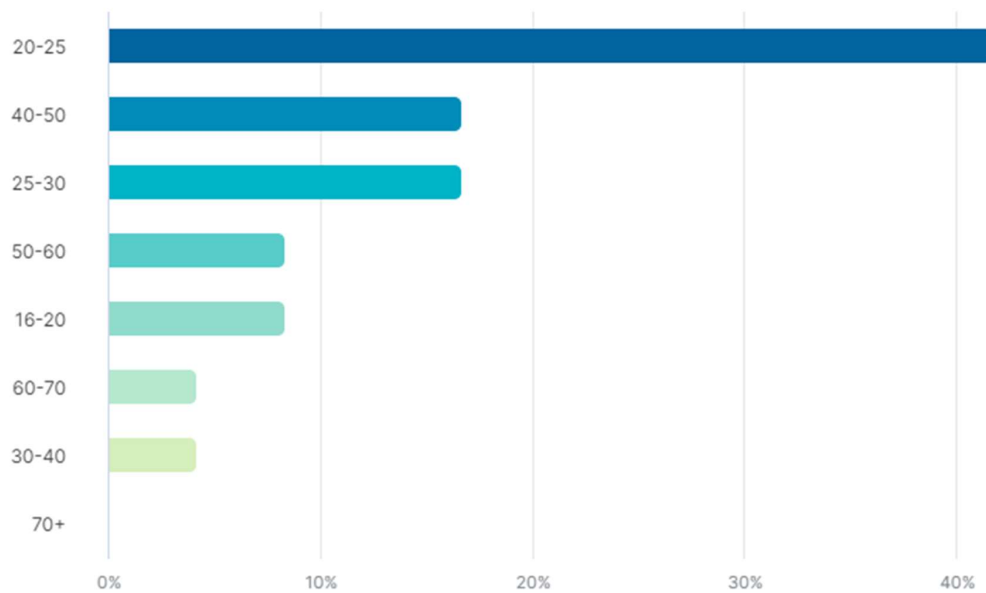
Autosport

Among the three platforms, Autosport turns out to be the most popular one. Although some participants find the posts slightly too long and include too much events and details, the respondents like the balance and the view and understanding they get by reading the content. It does, however, miss some of the important events that were included in the other blog sequences. Based on all three sequences, the majority of the survey respondents indicate that they would see themselves use this particular platform to stay updated during a race. The main reasons mentioned for this choice include the readability of the text, the writing style, and the amount of information included.

1. In what country do you live?

ANSWER	AMOUNT	RATIO
Netherlands	9	37.5%
The Netherlands	4	16.7%
Netherlands	4	16.7%
The Netherlands	3	12.5%
United Kingdom	1	4.2%
Nederland	1	4.2%
Nederland	1	4.2%
Australia	1	4.2%

2. How old are you?



3. How often do you attend Formula One races?

	-3	-2	-1	0	1	2	3	
Never	14	3	1		4	1	1	Every race

4. How often do you watch Formula One races live?

	-3	-2	-1	0	1	2	3	
Never			1		2	8	13	Every race

5. How often do you watch Formula One races back later?

	-3	-2	-1	0	1	2	3	
Never	5	2	3		9	2	3	Every race

6. How often do you follow Formula One races live via radio?

	-3	-2	-1	0	1	2	3	
Never	19	1	1		3			Every race

7. How often do you follow Formula One races live via textual updates/blogs?

	-3	-2	-1	0	1	2	3	
Never	6	2	4	2	5	4	1	Every race

8. How often do you check Formula One races afterwards via statistics?

	-3	-2	-1	0	1	2	3	
Never	2	2	2	1	7	4	6	Every race

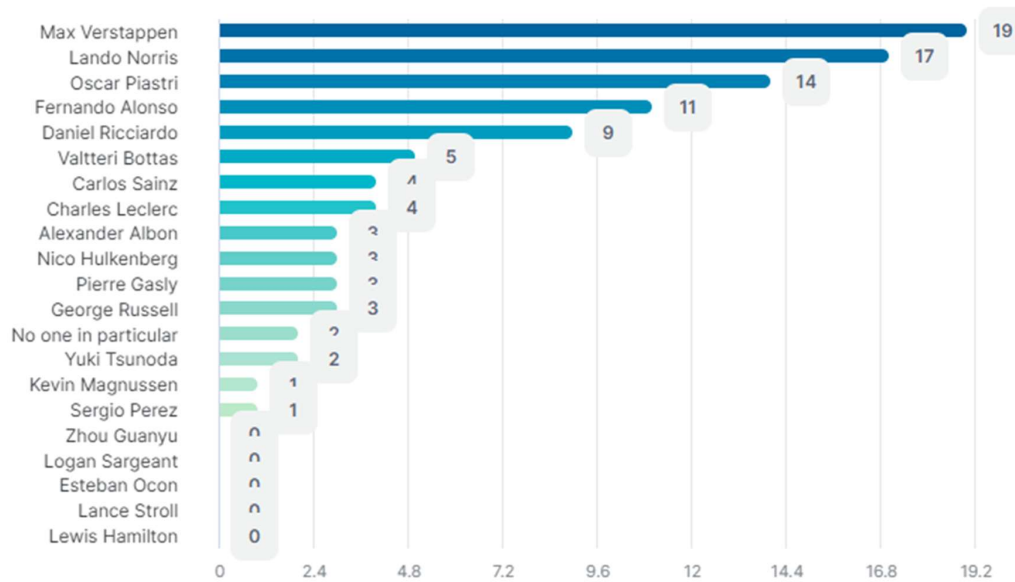
9. How often do you check Formula One races afterwards via news articles?

	-3	-2	-1	0	1	2	3	
Never	1	4	2	1	7	4	5	Every race

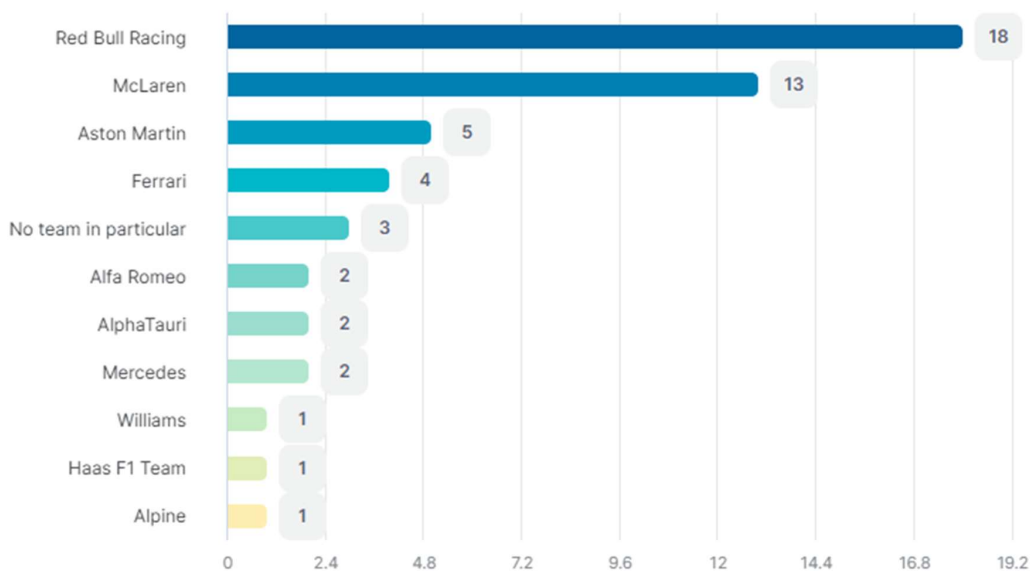
10. How often do you check Formula One races afterwards via textual updates/blogs?

	-3	-2	-1	0	1	2	3	
Never	7	1	3		9	3	1	Every race

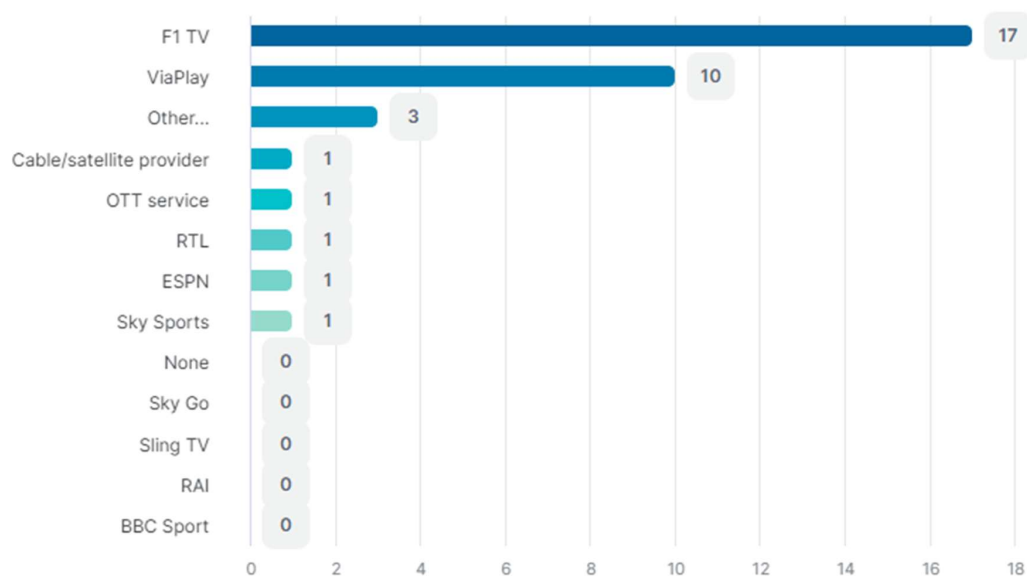
11. What driver(s) do you support?



12. What team(s) do you support?



13. What platform(s) do you use to watch Formula One races?



14. Why do you use this platform/these platforms?

- Viaplay to watch the race and the footage before and after the race. Sometimes I use F1TV to listen to the drivers radio communication with the engineer.
- Viaplay. Because it was the only option without creditcard. F1 pro needs a creditcard
- Variaty in screens. I can watch the race and have a live view in the cocpit.
- To be able to watch the race. It was the cheapest
- The alternative is Viaplay, which don't provide enough insight like the live timing in the F1 app
- Relatively easy and everything is available and customizable
- Pricing
- Only ones available
- Not to expensive life formula 1

Nice people who give comments. Platform works really nice. Choosing different languages is also a positive.

Most viewchannels, like driver, track info etc. And in combination with multiviewer, you can see it all at once

its the cheapest option for me, since I can share a subscription with my dad, who also is also very invested in F1. Sky is way overpriced in my opinion

It is possible to watch it in Dutch.

I like the way in which they display the information of the races.

F1tv cause it has the British commentators. Via play sometimes when I am at other places.

Enigste provider in Nederland en in het Nederlands.

English commentary and datachannels.

Dutch oriented and for analyses before and after F1 events such as free practices, qualifying and races.

Cause at the moment they are the only one providing f1 in The Netherlands. Few years back we watch with ziggosport and rtl 7.

Betrouwbaar en goede verbinding

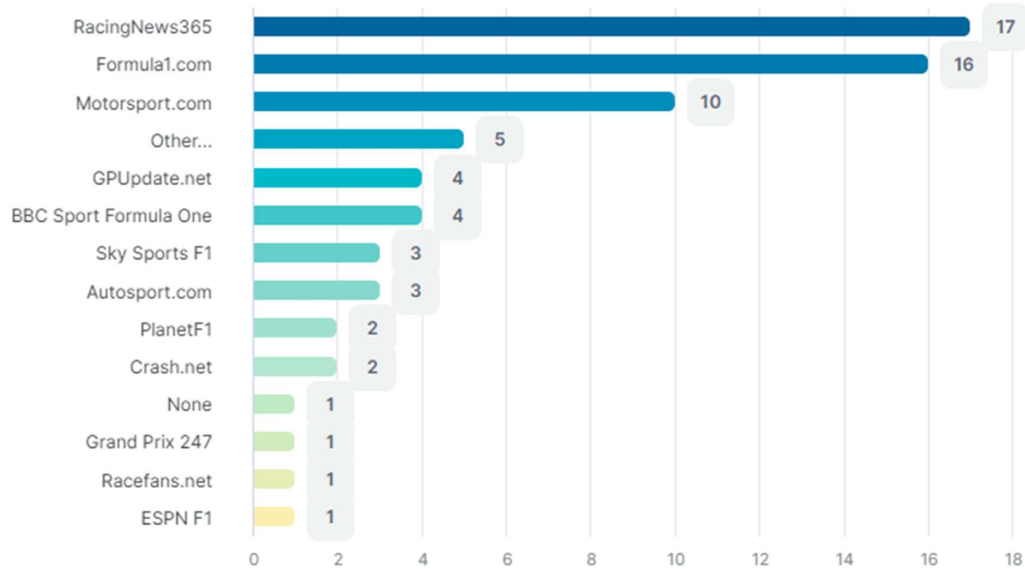
Because this one is free

Because they're convenient, offer different types of commentary, have interviews and offer analysis before and after races

Because there is live coverage and it was suggested in the media and (hypothetically) I could watch other Viaplay programs as well

because it is the easiest to access in Australia

15. What platform(s) do you use to read Formula One articles?



16. Why do you use this platform/these platforms?

To use text based when video is not available

To scroll through articles etc.

To read some news and get to now some stats

To follow the teams and changes in regulations

Once a week to check the times and schedule of racing

Nrws, insights, fun facts

Not all are honest. So i try multiple platforms to see the truth. And still is fragile

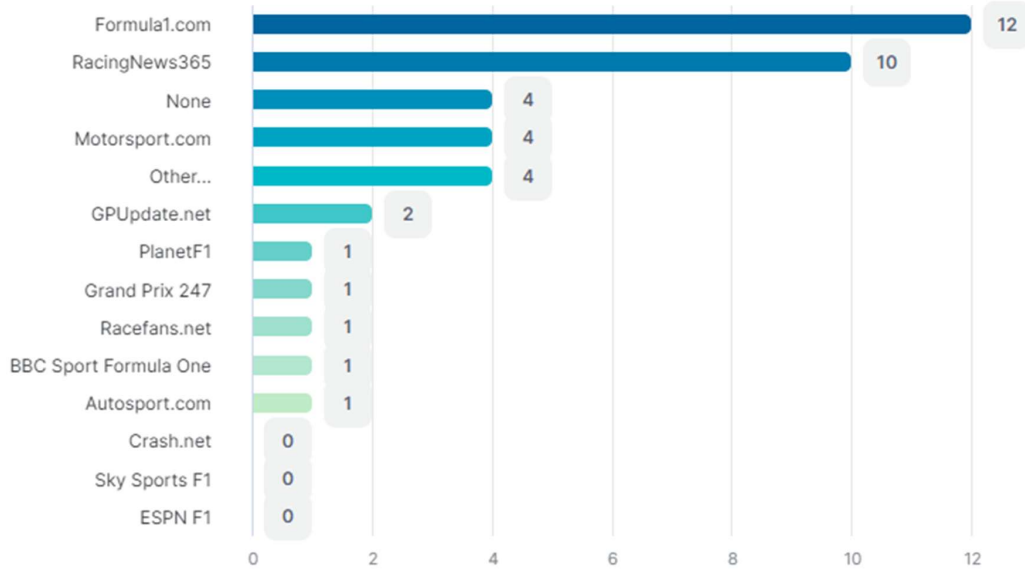
No particular reason

Nieuwsartikelen over de F1

Most up to date and trusted source

Lees ze niet
It shows up on my social media feeds
It is very up-to-date and in Dutch
I think they are reliable and update content on a regular basis.
I think it has the most up-to-date information about formula 1.
I mostly use X (twitter before) to follow F1 news. On that platform they share all the articles. I read the ones that are interesting to me.
I like the way the news is presented and I don't have time to check all of them.
I have the F1 app installed and get updates/notifications from time to time
Good articles
F1.com for the official articles and technical background and Racingnews for more general stuff.
For a variety of types of information and for memes
Cause I usually see these coming in my social media feed.
because I am familiar with them

17. What platform(s) do you use to read Formula One live textual updates/blogs?



18. Why do you use this platform/these platforms?

To read f1news
To check out updates they don't broadcast
To get relevant information about the races and drivers
This is usually the first one that pops up when I search google. But I almost never do cause I want to watch the race back when I miss it live.
The same as my previouce answer
same as before
No specific reason
no real reason
No particular reason

Must up to date and trusted source
Most accurate in my belief.
Live comment
It's the official channel for the blog, so it's unbiased
It is in Dutch and interesting
I think it has the most up-to-date information about formula 1.
I can access it easily on my mobile phone
Geen
Fun facts
Ease of use.
Don't use it enough
Daar voorspel ik de uitslag van de races
Because X is the fastest platform to share news, during a live race.

RacingNews365

RacingNews365, Bahrain GP 2023

16:05, Lap 1

Lights out and away we go

"Aggressive at the start from the Ferrari and Sergio Perez.

Verstappen leads, Leclerc second, Perez third and Sainz fourth as contact between the two Aston Martins.

Everyone is still going."

16:07

What was Stroll doing?

"Lance Stroll broke way too late for Turn 4 of the opening lap, and whacked into Fernando Alonso. Incredibly lucky both Aston's are still going - as Max Verstappen is 2s clear at the start of the third lap."

16:09

Sketchy at the start

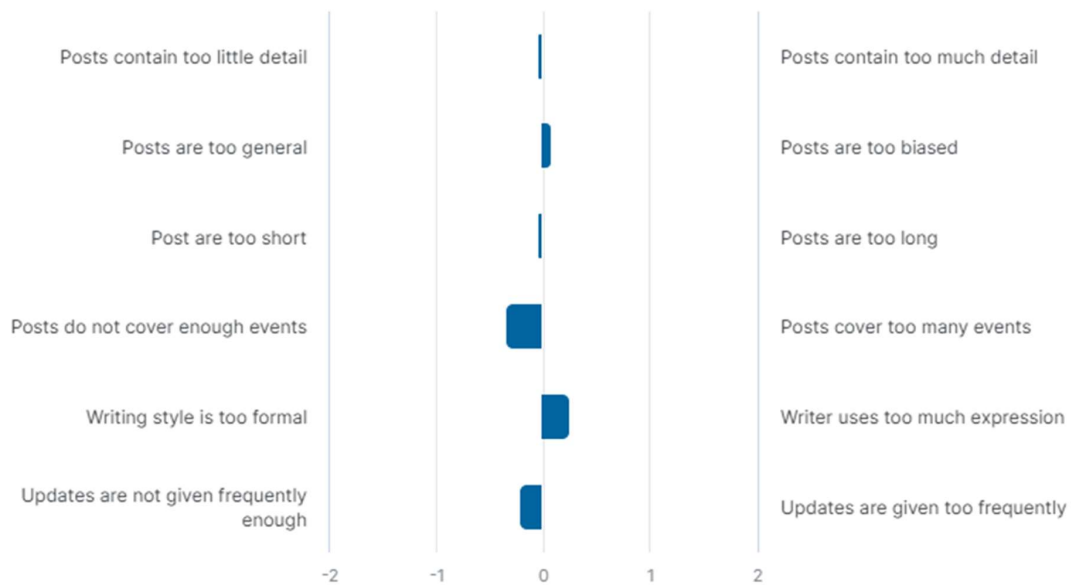
"A poor start from Sergio Perez allowed Charles Leclerc to pull alongside and grab P2 at the start. Very impressive everyone is still going."

16:11, Lap 5

Verstappen had his porridge this morning

"Verstappen is 3.5 seconds ahead of Leclerc at the start of the fifth lap, with Leclerc losing more time after a big lock-up at Turn 1.

The gap is now four seconds."



RacingNews365, Monaco GP 2023

15:50, Lap 36

"Sainz is furious with Ferrari's strategy team, who focused on covering Hamilton.

"I don't care about Hamilton," he said in an expletive-laden radio rant."

15:52, Lap 37

"It's worth noting that whilst Ocon and Hamilton switched from Mediums to Hards, Sainz has gone the other way.

He is about to have the same issues his rivals fought in the first stint.

Stroll makes more contact, this time with Magnussen into Anthony Nogh  s - that costs him more front wing parts."

15:53, Lap 39

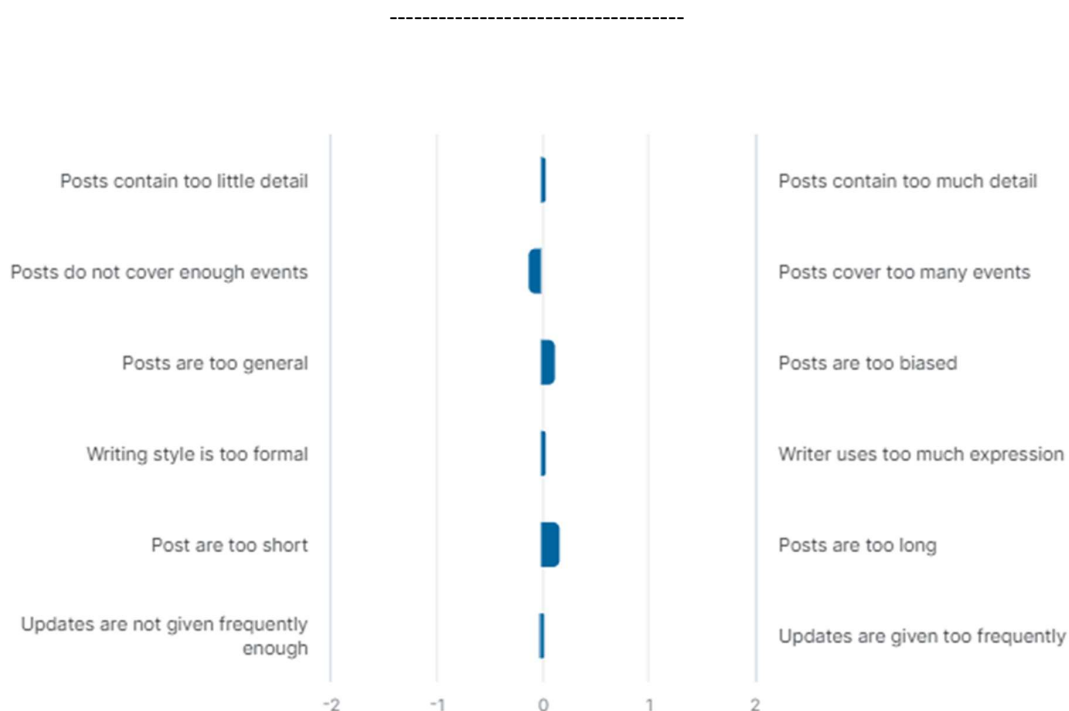
"Halfway into the race and Verstappen, on worn and grained Mediums, is stretching the gap back out over Alonso.

This is some performance from the Dutchman."

15:57, Lap 41

"The track temperature has dropped nine degrees since the start of the race and the air temperature has plummeted as well.

Rain is still in the area but still no sign yet."



RacingNews365, British GP 2023

16:20

Light rain on the way

"Red Bull reporting "light-ish" rain that will last around 4 minutes that will hit the track immentantly."

16:25

Wind creating problems

"Verstappen reporting that the wind is making it "difficult to drive" for him up front.

Well, we had to make it a challenge Max after all these 'easy' races!"

16:27

Russell close to Leclerc

"Russell is gaining on Leclerc now, around 0.5s behind the Ferrari. Seems to be losing out in the middle sector."

16:28

Ocon retirement

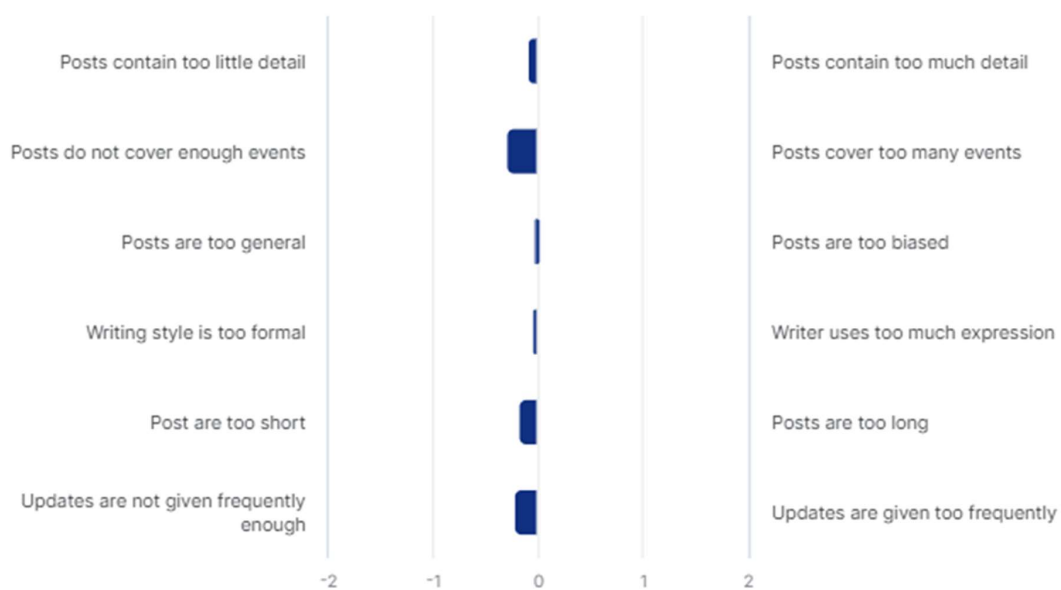
"Ocon had a hydraulic leak on his car which caused his retirement according to the team."

16:31

Perez makes progress

"Perez up to P10 after passing Albon into Stowe, but he's not easily moving through the field.

The wind clearly affecting Red Bull, it's 7.9km which was similar to FP2."



Timewise it should be the other way round
the descriptions were a bit short/seemed cut off, but that seems to be normal for live commentary
Only becomes clear a few minutes later why Leclerc went to second position, could be already updated earlier
I think the updates are Almost perfect. I would only add a bit more updates in which they explain the information of more drivers.
I really liked it actually with a bit of an edge and frequent updates
I need facts when I am on my way. Need: Verstappen overtakes Piastei (p2 to p1)
I like to watch the interval times cause I would like to know who is in drs and who isn't.
I liked the way they are written, it was very easy to read. They have the right amount of information in 1 post.
If you put them all together you get the perfect blog
I miss positions
Like how it has a short title to each post so you directly know what is going on
I like this one!
Feel like out of the 3 this contains the least details.

PlanetF1

PlanetF1, Bahrain GP 2023

Lap 1

"Lights out! Go go go!"

Lap 1

"Sergio Perez has lost a place to Charles Leclerc and Fernando Alonso has a suspected puncture.
Contact with Lance Stroll!"

Lap 2

"Panic over. There was contact between Stroll and Alonso but both cars are fine to continue racing.
Alonso has lost two places though."

Lap 3

"Max Verstappen has opened up a gap of 1.5s-2s to second-placed Charles Leclerc."

Lap 4

"Make that three seconds. Verstappen is flying!"

Lap 6

"Sergio Perez is within DRS range of Charles Leclerc in the battle for P2."



PlanetF1, Monaco GP 2023

Lap 36

"Medium tyres for Perez and a front wing change, his awful weekend gets worse. The stewards say no further action as he gave the place back to Stroll. Well, kind of, not by choice and not without leaving the Aston Martin damaged. Sainz is furious meanwhile with Ferrari's strategy indecision. He felt there was a chance to pass Ocon."

Lap 38

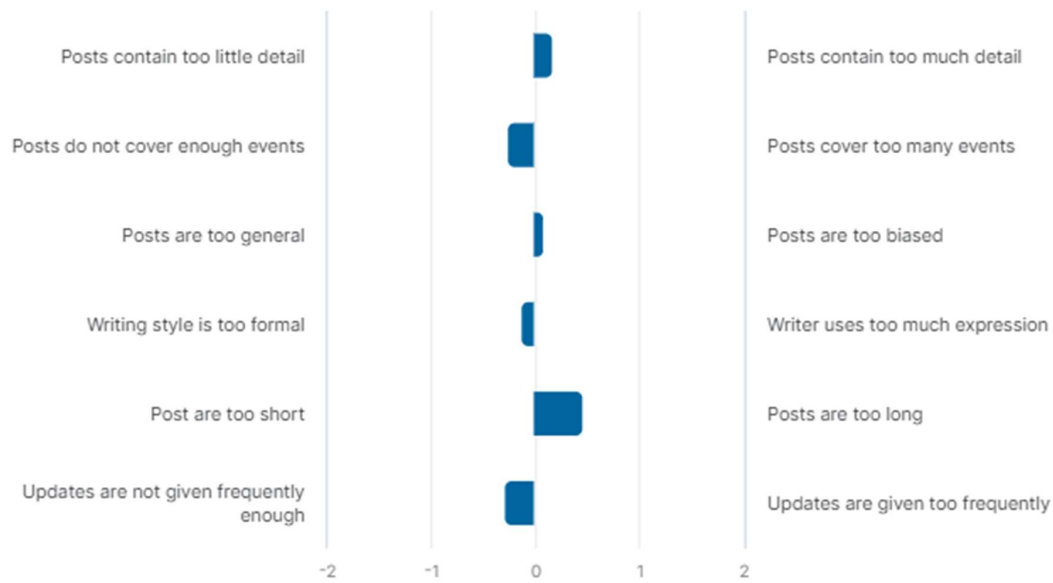
""Potential risk of rain with 20 laps to go," Williams tell Albon. This rain so far has been elusive. Stroll meanwhile has given Magnussen a whack to further wound his Aston Martin."

Lap 40

"The Magnussen-Stroll contact at Anthony Noghes has been noted."

Lap 41

"Relying on the naked eye rather than the radars, there is a big black cloud, clearly dumping rain, moving closer from the hills towards the track."



PlanetF1, British GP 2023

Lap 11

"Red Bull have informed Max Verstappen they expect some "light-ish" rain in the next few minutes... how that manifests itself and whether or not that'll have a big impact, we do not know."

Lap 12

"Elsewhere, Mercedes claim to George Russell that it'll miss the track. Proof if ever it were needed that in Britain, you can never get away from talking about the weather."

Lap 13

"Sergio Perez continues his recovery work with a move down the inside of Stowe on Lance Stroll in the Aston Martin, he's now up to P11 but he's a long way from where he wants to be still."

Lap 15

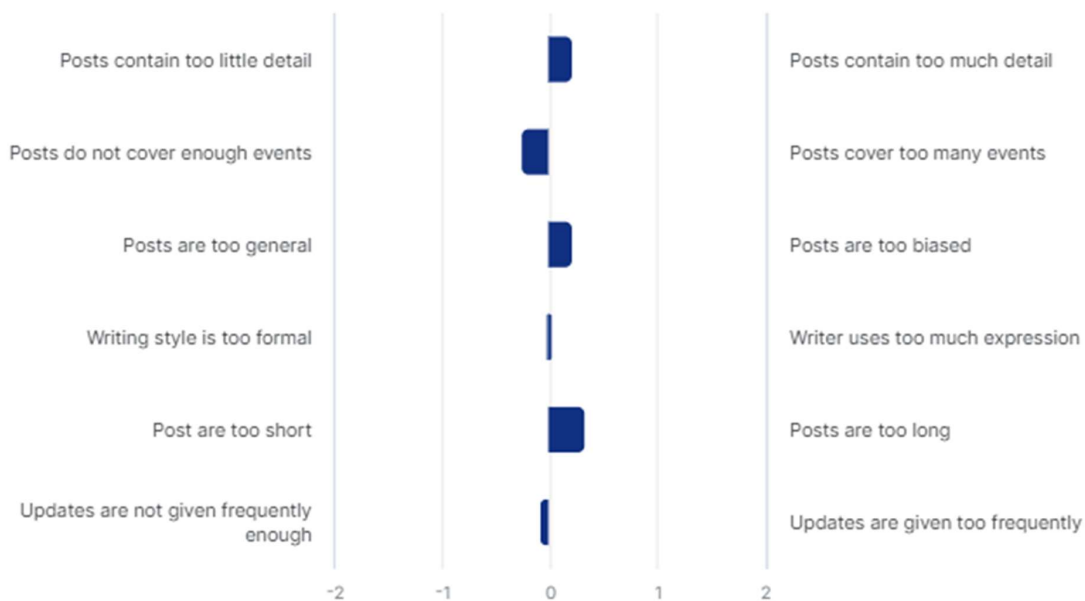
"Light drizzle" is the message from Max Verstappen, while Lando Norris has a lap time deleted for track limits at Copse. Please, Lando, let's leave track limits in Austria..."

Lap 16

"George Russell is still all over the back of Charles Leclerc as the Ferrari driver has to defend into Stowe once again. He's wanting P4 here."

Lap 17

"Meanwhile, Alpine have confirmed to us that a hydraulic leak was the cause of Esteban Ocon's retirement, and Sergio Perez is now up into the points after passing Alex Albon."



- Very little information, doesn't talk about many drivers.
- This one combined with the previous one would have my preference thus far
- This commentary seemed to be a bit more like what you would typically hear during a race start
- Same as the last drs interval
- Indicating the lap is more graphic to the comment where time os indicated
- I miss positions from the drivers
- I like this type of lifeblood! Sweet and short sentences

I didn't really like the style they were written in which made it a bit harder for me to read them easy and quick.

Again. All together is the best. It is what you like personal

Can't read this on my way

Too long

Rather watch it later

Quite long but okay

Just good

Autosport.com

Autosport, Bahrain GP 2023

16:05

"And the 2023 Bahrain Grand Prix is go! Verstappen starts well, as Perez gets crowded out and Leclerc gets the move done into Turn 1!"

16:05

"The two Aston Martins have come so close to a major clash at Turn 4 on the opening lap! It looks like they've got away with it but Alonso is down to seventh and Stroll is ninth."

16:06

"Both Mercedes have got into the top six after that Alonso/Stroll near miss, Hamilton ahead of Russell at the end of the first lap."

16:07

"Bottas has made a mega start gaining four places on the opening lap to move up to eighth, while team-mate Zhou has had the opposite fortune and lost four spots. Hulkenberg also a big loser at the start, dropping four spots and is down to 14th."

16:08

"Verstappen's got a 1m37.974s on the last lap to break the DRS gap to Leclerc, it's 1.9s as it stands at the start of lap 3."

16:08

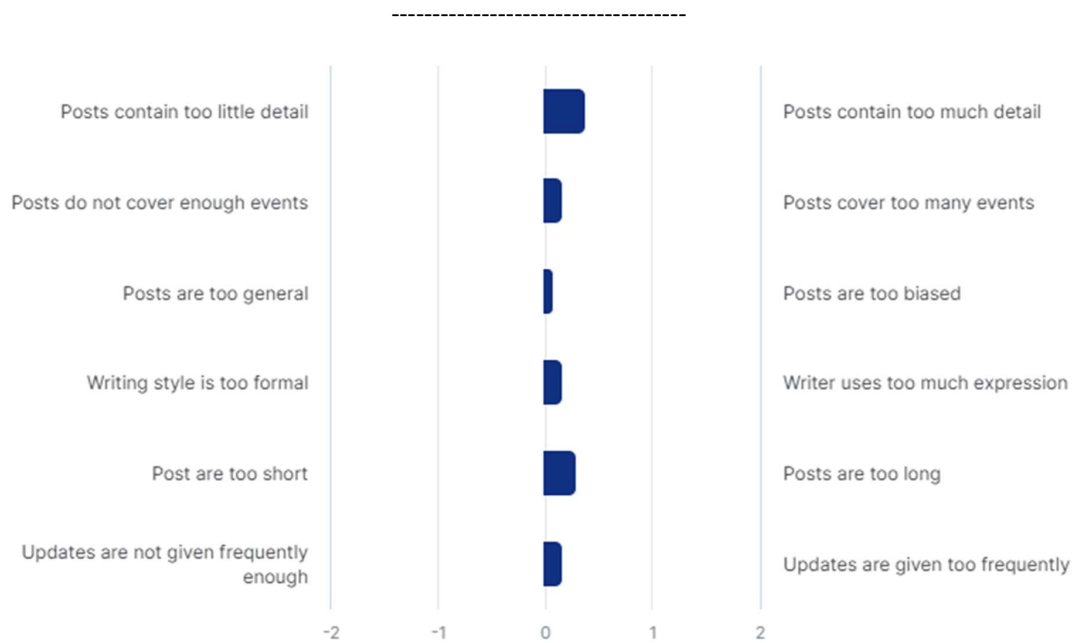
"Replays confirm Stroll definitely gave Alonso a bang from behind under braking into Turn 4, but somehow they've both avoided any serious damage."

16:09

"I've been hit at Turn 4, they cannot do that," Alonso said over team radio. He probably didn't know that *they* was his new team-mate! Welcome to Aston Martin!"

16:11

"Stroll gets by Bottas into Turns 1-2 to take eighth place, while further back Sargeant is battling with Norris for 12th."



Autosport, Monaco GP 2023

15:49

"Sainz shouts "I don't care about Hamilton" behind him as he feels his strategy has ruined his race. He's seventh still."

15:50

"Perez has gotten back into 19th ahead of Sargeant."

15:50

"Sainz asked what Leclerc's pace was and says he was faster. Not a happy camper in that Ferrari right now."

15:51

"Albon in 18th told there's a risk of rain with 20 laps to go."

15:51

"Now clear of traffic at last, Verstappen's gap to Alonso is 8.0s starting lap 38. Rain continues to be a concern on the pitwall."

15:51

"Stroll touched Magnussen at the last corner as he tried an overtake and reports he's got damage. He's still in 15th."

15:52

"Through all of the drama over the last few laps, Hamilton in eighth has posted the fastest lap of the race with a 1m15.650s."

15:53

"The gaps between Leclerc, Gasly and Russell remain fairly stable. Ocon behind is biding his time knowing he's in a net third once they pit (assuming rain doesn't bring everyone back to the pits again)."

15:54

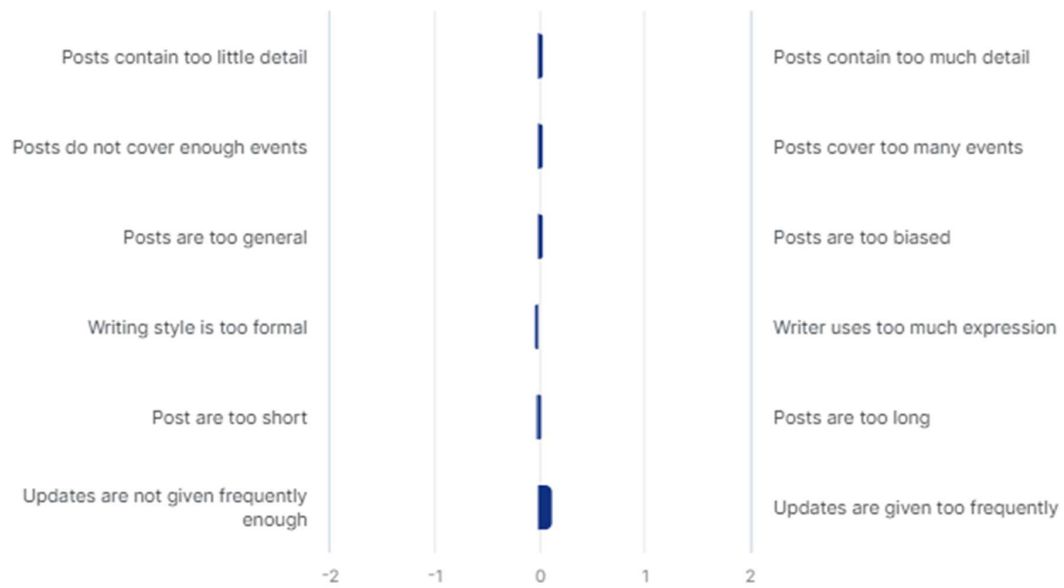
"The Stroll/Magnussen touch has been noted by the stewards."

15:54

"Graining or not, Verstappen is still able to lap faster than Alonso and has extended his lead to 8.8s by going eight tenths quicker last time around."

15:55

"By the end of lap 40 that gap has grown to 9.2s. Verstappen is really on it here trying to keep Alonso at arms length."



Autosport, British GP 2023

16:19

"Rain is in the air... lots of teams are reporting it will hit the circuit in a couple of minutes. Spicy."

16:22

"Russell is reporting drops of rain on his visor... so, if his Spanish GP experience is to be followed, there's a 100% chance of sweat in that Mercedes cockpit."

16:23

"Stroll's afternoon and season takes another turn for the worse, as Perez gets past for P11.
Just one more place for Perez before he hits the points-paying positions."

16:23

"Norris confirms over McLaren team radio he is going for Plan A, which is expected to be a one-stop race. So he'll be looking after his tyres to push out this stint."

16:25

"Tsunoda is the first driver to make a planned pitstop from P13."

 16:25

"Verstappen has pushed out his lead to 3s over Norris, his last lap half a second quicker than the McLaren driver."

 16:26

"Medium tyres off and soft tyres on for Tsunoda."

 16:27

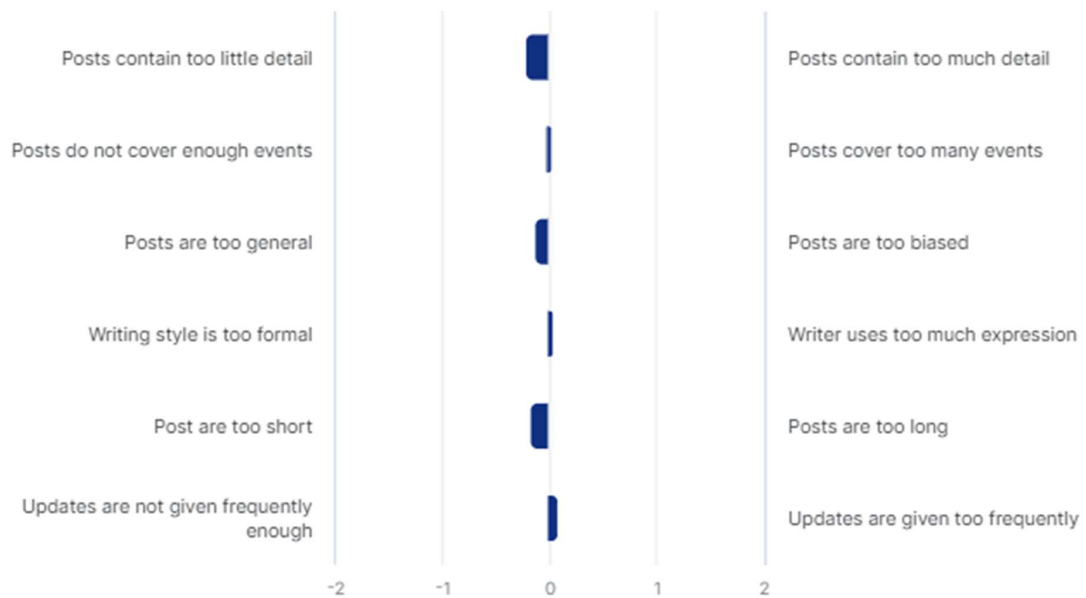
"Light drizzle," Verstappen reports on his radio."

 16:29

"Lots of tyre management going on at the front, while Russell still cannot find a way past Leclerc in fourth place despite running half a second back."

 16:29

"Into the points for Perez. He moves past Albon and into the points using DRS along the Hanger Straight."



Wow, that was quite a lot of information about the race in a short period of time. For me it was a bit too much and the posts were too long.

Too wordy, missed the important information

this one is my favourite

They are good

There's a good balance in events described, length and frequency.

That was perfect

Positions are missing

Out of the 3 I liked this one the most by far. Because more regular updates are given and not only the main big events of the top 3 and clash between the Aston Martins is talked about but also other events in the rest of the field (Bottas, Mercedeses, Norris, etc.).

I would not follow the race this way because I don't get enough of the information which makes the races fun for me to watch.

I think these updates would be a great way to follow the race without watching. You know what's going on by many drivers and you get a better feeling for the atmosphere of the race.

Again I like this one the most because more updates about more drivers are given which gives a better view and understanding of what's happening across all drivers.

Rather watch it later

Nice one

Many short posts but don't indicate where in the race you are or what's it about

Feel like compared to the previous I missed some things in the Autosport one, such as about the retirement of Ocon.

Why would you skip the platform(s) not chosen?

Too short and fast updates

Too much words in one post

Too much detail

Too long, too focused of to frequent

Too little detail

they are good but i'd want to read both for more context

There was to much text in one post which made it quite hard to read.

The notifications are very long and only talk about a small proportion of the race

Same as the first two

Posts too long or lacking information /events.

Planet F1 has a nice readable live report

One is enough

not that interesting

Not my favorit

Not applicable

N/A

Minder aansprekend.

I don't like the writing style

Gives not relevant information, and is focused on only the big names and front field

Geen voorkeur

Don't know if it is possible in Dutch

Don't know

Again too general and not enough info given.

B

Research Topics

In this chapter, content of the *Research Topics* report that is relevant as support to certain parts of the thesis is placed. Therefore, it involves reused material of the original report [27].

B.1 Data-to-text generation

There are various methods that can be used for the automatic generation of natural language, of which some were introduced in section ???. A commonly used approach is the template-based approach, which was already used in the previous century [40] [112]. Over the years, more advanced approaches were introduced, such as the planning-based approach or data-driven methods that involve the use of probabilistic or statistical modeling techniques, as is discussed in the survey paper of Gatt and Krahmer [10]. State-of-the-art approaches mainly seem to involve the use of large data sets and DL models [113]. Below, a variety of approaches for the generation of text are discussed individually.

Template-based

When talking about template-based text generation systems, one means systems that can convert non-linguistic input directly, without the need of an in-between step such as training a model [114]. The templates can be simply described as texts that have 'gaps' that need to be filled with the desired data [115] [114]. Often, triples are used to present the data entities and their relationships to eventually form accurate sentences with multiple input entities [115]. An example of such a template for a car race comment could look as follows:

[DRIVER 1] *has just breezed past the* [CAR BRAND 2] *of* [DRIVER 2].


Here, between the brackets, the names and car involved can be inserted so that it forms a sentence representing what is actually happening in the race. The triples that would be needed to form this sentence would look as follows:

<[DRIVER 1], *overtake*, [DRIVER 2]>

<[DRIVER 2], *team*, [TEAM 2]>

<[TEAM 2], *car brand*, [CAR BRAND 2]>

Within these triples, the left and right units represent the entities, while the middle one represents the relationship between the two [115]. This template approach is a relatively simple method of data-to-text generation, although it comes with several disadvantages. For instance, many templates are needed for such an approach to result

 The content of section B.1 is reused material from section 2.2 (*related work - language generation*) of the *Research Topics* report [27].

in original, non-repetitive sentences. A wide range of cases needs to be covered, such as an event occurring in the past or present, for which different template sentences are needed. Furthermore, template-based systems are often more difficult to maintain and improve compared to more modern and/or advanced approaches [114] and the result can be poor when the constraints of the templates do not cover the constraints of the output text [115]. However, due to its simplicity and performance (especially when it involves shorter text where repetitiveness is less of an issue), it is still a commonly used method in the field of data-to-text generation.

Planning-based

The planning-based approach is, similar to the template-based one just discussed, one of the oldest approaches to NLG [116]. Within this approach, the text to generate is seen as a sequence of actions, each with preconditions and effects [10]. A sequence of actions that meets the communicative purpose is retrieved from the input using a planner, which is then converted into natural language [10]. In the survey paper of Gatt and Krahmer [10], two ways of applying a planning-based approach are discussed: either through the grammar or using reinforcement learning. Within the first, a modern approach is called Lexicalised Tree Adjoining Grammar (LTAG) [10]. LTAG involves the use of elementary trees, which can be considered to be linguistic building blocks, that are combined with conceptual information to form text [10]. Whereas planning-based text generation through grammar is in principle a rule-based method where the relationship between an action and the corresponding consequence(s) is seen as a fixed phenomena, the approach involving reinforcement learning considers this relationship as *uncertain* [10]. Here, the text generation is represented as a Markov decision process, where each state is linked to a potential action, and each combination of a state and an action has a probability of transitioning from one state to a new state with the corresponding action [10]. These transitions are connected to a reinforcement signal [10].

Deep learning

Previously, generation of text relied on pre-defined linguistic structures [117]. In recent years, a shift has been noticed, in which neural network (NN) approaches, also known as deep learning (DL) techniques, are implemented in the field of NLG more and more [117], which leads to these pre-defined structures no longer being needed. All the way back to the 1980s, the first preliminary experiments with NN applications in the field of NLG were conducted [118]. The decade that followed thereafter, the use of NN architectures in the domain gained interest and more research was done on the subject [10]. The results of using DL techniques within the field has led to an increase in performance compared to any other AI and/or statistical methods used previously [119]. Overall, DL based approaches in NLG applications make use of generative language *learning* methods and combine these with NN based frameworks [117].

While recurrent NNs based on Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) structures used to be the go-to approach for text generation, countering some of their flaws arising from its inherent recurrent structure have led to a shift towards a new architecture [44]. This type of NN structure, which is most commonly used nowadays, is the Transformer architecture, an architecture that forms the base of popular language models like GPT and BERT, which have outperformed prior approaches with large margins [44]. This approach leaves out the recurrency within the NN and only relies on attention mechanisms, which aim to understand the underlying construction of sentences by extracting the relations between words within the sentence [44].

Due to the success of the Transformer-based architectures, the majority of the state-of-the-art text generation approaches involve using this type of architecture [44]. In 2018, large-scale and pre-trained language models began to take over the field of NLP (and NLG) [120]. Such models use a large amount of data and unsupervised learning techniques, based on the Transformer architecture, resulting in them being able to perform NLG tasks with just "a bit" of finetuning [120]. Over the years, the use of such language models for various NLP-related tasks has expanded [121]. By inputting data into such models to finetune them, they can extract knowledge by determining the relationships between certain entities and apply this knowledge to, among other capabilities, generate text based on new inputted data [121]. The recent development in which language models are first pre-trained before being finetuned to a more narrowed down task has led to a large step made in the performance of such models [122]. Examples of such models are the Generative Pre-trained Transformer (GPT-3), which is the language model behind the popular chatbot ChatGPT, and Bidirectional Encoder Representations for Transformers (BERT) [122].



Live timing data

Below, the data that can be publicly accessed via the API's of *Formula One* itself is listed. For each item, the URL to the data for the Grand Prix in Miami in 2023 is used as an example. To get an understanding of what data can be extracted from each entry point, the keys extracted using a Python function are also listed.

RaceControlMessages

Content: Messages from race control per lap

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/RaceControlMessages.json

Keys: 'Utc', 'Lap', 'Category', 'Flag', 'Scope', 'Message', 'Sector', 'Status', 'RacingNumber'

SessionInfo

Content: Information about circuit

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/SessionInfo.json

Keys: 'Meeting' ('Key', 'Name', 'OfficialName', 'Location', 'Country', 'Circuit'), 'ArchiveStatus', 'Key', 'Type', 'Name', 'StartDate', 'EndDate', 'GmtOffset', 'Path'

SessionStatus

Content: Current status of the session

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/SessionStatus.json

TeamRadio

Content: MP3 files of team radio messages including time

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/TeamRadio.json

Keys: 'Utc', 'RacingNumber', 'Path'

TimingAppData

Content: Information about tyres, starting position, current position, and more

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/TimingAppData.json

Keys: 'RacingNumber', 'Line', 'GridPos', 'Stints' ('LapTime', 'LapNumber', 'LapFlags', 'Compound', 'New', 'TyresNotChanged', 'TotalLaps', 'StartLaps')

TimingStats

Content: Best lap times, speeds, and more

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/TimingStats.json

Keys: 'Line', 'RacingNumber', 'PersonalBestLapTime', 'BestSectors', 'BestSpeeds'

TrackStatus

Content: Info about whether the track is clear

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/TrackStatus.json

WeatherData

Content: Weather info, including track temperature

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/WeatherData.json

Keys: 'AirTemp', 'Humidity', 'Pressure', 'Rainfall', 'TrackTemp', 'WindDirection', 'WindSpeed'

ContentStreams

Content: Live coverage, audio, commentary

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/ContentStreams.json

Keys: 'Type', 'Name', 'Language', 'Uri', 'Utc', 'Path'

SessionData

Content: Track status, session status

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/SessionData.json

TimingData

Content: Gaps between drivers, nr of laps, pit out/in, and more

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/TimingData.json

Keys: 'GapToLeader', 'IntervalToPositionAhead', 'Line', 'Position', 'ShowPosition', 'RacingNumber', 'Retired', 'InPit', 'PitOut', 'Stopped', 'Status', 'NumberOfLaps', 'NumberOfPitStops', 'Sectors', 'Speeds', 'BestLapTime', 'LastLapTime'

LapCount

Content: Current lap

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/LapCount.json

ChampionshipPrediction

Content: Predicted points as it stands now

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/ChampionshipPrediction.json

Keys: 'Drivers' ('RacingNumber', 'CurrentPosition', 'PredictedPosition', 'CurrentPoints', 'PredictedPoints'), 'Teams' ('TeamName', 'CurrentPosition', 'PredictedPosition', 'CurrentPoints', 'PredictedPoints')

AudioStreams

Content: Live coverage

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/AudioStreams.json

Keys: 'Name', 'Language', 'Uri', 'Path', 'Utc'

DriverList

Content: Including team, picture, and more

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/DriverList.json

Keys: 'RacingNumber', 'BroadcastName', 'FullName', 'Tla', 'Line', 'TeamName', 'TeamColour', 'FirstName', 'LastName', 'Reference', 'HeadshotUrl', 'CountryCode'

ExtrapolatedClock

Content: Maximum time left

URL: https://livetiming.formula1.com/static/2023/2023-05-07_Miami_Grand_Prix/2023-05-07_Race/ExtrapolatedClock.json

Keys: 'Utc', 'Remaining', 'Extrapolating'

D

Experiments

D.1 LLaMA

Below subsection headers show the input given to the model. The text displayed in each subsection shows what the model gave back as output. The LLaMA model used in this experiment is the 7B finetuned model, trained in the first half of 2023 and optimized for dialogue use cases [123].

D.1.1 What drivers are there in *Formula One*?

Formula 1 drivers are some of the most skilled and experienced racers in the world. They must have a combination of speed, agility, and endurance to compete at the highest level. Here are some of the most notable Formula 1 drivers:

1. Lewis Hamilton - Lewis Hamilton is widely regarded as one of the greatest Formula 1 drivers of all time. He has won 7 world championships and 103 races, including a record 6 constructors' championships with Mercedes.
2. Michael Schumacher - Michael Schumacher is a legendary Formula 1 driver who has won 7 world championships and 91 races. He holds numerous records, including most race wins, fastest laps, and most championships won in a row.
3. Sebastian Vettel - Sebastian Vettel is a four-time Formula 1 world champion and has won 53 races. He is known for his consistency and ability to perform under pressure.
4. Fernando Alonso - Fernando Alonso is a two-time Formula 1 world champion and has won 32 races. He is known for his exceptional driving skills and ability to adapt to different cars and tracks.
5. Kimi Räikkönen - Kimi Räikkönen is a former Formula 1 world champion and has won 21 races. He is known for his calm and focused driving style, and his ability to perform well in difficult conditions.
6. Max Verstappen - Max Verstappen is a young and talented Formula 1 driver who has won 10 races. He is known for his aggressive driving style and ability to push the limits of the car.
7. Daniel Ricciardo - Daniel Ricciardo is a popular Formula 1 driver who has won 7 races. He is known for his speed and ability to perform well in qualifying.
8. Charles Leclerc - Charles Leclerc is a rising star in Formula 1 and has won 3 races. He is known for his exceptional driving skills and ability to adapt to different cars and tracks.
9. Esteban Ocon - Esteban Ocon is a talented Formula 1 driver who has won 1 race. He is known for his consistent performance and ability to perform well in difficult conditions.
10. Pierre Gasly - Pierre Gasly is a young and talented Formula 1 driver who has won 1 race. He is known for his speed and ability to perform well in qualifying.

D.1.2 What drivers are there in *Formula One* currently?

There are currently 20 drivers in Formula 1, and here are the names of the drivers and their teams:

1. Lewis Hamilton - Mercedes
2. Valtteri Bottas - Mercedes
3. Max Verstappen - Red Bull Racing
4. Charles Leclerc - Ferrari
5. Sebastian Vettel - Ferrari
6. Pierre Gasly - AlphaTauri
7. Daniel Ricciardo - Renault
8. Esteban Ocon - Renault
9. Lando Norris - McLaren
10. Carlos Sainz Jr. - McLaren
11. Alexander Albon - Williams
12. Robert Kubica - Williams
13. Daniil Kvyat - Toro Rosso
14. George Russell - Williams
15. Antonio Giovinazzi - Alfa Romeo
16. Kimi Räikkönen - Alfa Romeo
17. Romain Grosjean - Haas
18. Kevin Magnussen - Haas
19. Sergio Perez - Racing Point
20. Lance Stroll - Racing Point

Note: This list only includes the drivers who are currently competing in the Formula 1 World Championship and does not include any reserve drivers or drivers who are not competing full-time in the championship.

D.1.3 What Grands Prix are organised in *Formula One* and where?

There are 21 Grands Prix organised in Formula One, held in different countries around the world. Here is a list of the current Formula One Grands Prix, along with their location and circuit:

1. Australian Grand Prix - Melbourne Grand Prix Circuit, Melbourne, Australia
2. Bahrain Grand Prix - Bahrain International Circuit, Sakhir, Bahrain
3. Chinese Grand Prix - Shanghai International Circuit, Shanghai, China
4. Azerbaijan Grand Prix - Baku City Circuit, Baku, Azerbaijan
5. Spanish Grand Prix - Circuit de Barcelona-Catalunya, Barcelona, Spain
6. Monaco Grand Prix - Circuit de Monaco, Monte Carlo, Monaco
7. Canadian Grand Prix - Circuit Gilles Villeneuve, Montreal, Canada
8. British Grand Prix - Silverstone Circuit, Northamptonshire, England
9. German Grand Prix - Hockenheimring, Hockenheim, Germany
10. Hungarian Grand Prix - Hungaroring, Budapest, Hungary
11. Belgian Grand Prix - Circuit de Spa-Francorchamps, Stavelot, Belgium
12. Italian Grand Prix - Autodromo Nazionale Monza, Monza, Italy
13. Singapore Grand Prix - Marina Bay Street Circuit, Singapore
14. Japanese Grand Prix - Suzuka Circuit, Suzuka, Japan
15. Russian Grand Prix - Sochi Autodrom, Sochi, Russia
16. Mexican Grand Prix - Autódromo Hermanos Rodríguez, Mexico City, Mexico
17. United States Grand Prix - Circuit of The Americas, Austin, Texas, USA
18. Brazilian Grand Prix - Autódromo José Carlos Pace, Interlagos, São Paulo, Brazil
19. Abu Dhabi Grand Prix - Yas Marina Circuit, Yas Island, Abu Dhabi, United Arab Emirates

20. Sao Paulo Grand Prix - Interlagos, São Paulo, Brazil
21. Australian Grand Prix - Melbourne Grand Prix Circuit, Melbourne, Australia

D.1.4 What do you know about *Formula One*

Formula One, also known as F1, is the highest class of single-seater auto racing sanctioned by the Fédération Internationale de l'Automobile (FIA). The F1 season consists of a series of races, known as Grands Prix, held on purpose-built circuits and on public roads around the world. The results of each race are combined to determine two annual World Championships, one for drivers and one for constructors.

Orientation on Large Language Models

According to a comparison of LLMs presented by Sha [124], the two best-performing models are the latest two GPT versions. There are, however, some less commonly used LLMs available that each have their own advantages and limitations. In table E.1, an overview of a selection of language models that are potentially suitable for the live update generation task is given, together with their strengths and limitations briefly presented.

Table E.1: Overview of Large Language Models [124]

Creator	Model	Strength(s)	Limitation(s)
OpenAI	GPT-3	General purpose, fast processing time (response within seconds) [124]	Frequent spread of inaccurate information, low level of expertise in more narrow domains [124]
	GPT-4	Multimodal input, complex capabilities, mostly accurate information [124]	Slow response and inference time [124]
Google	PaLM 2 (Bison)	Multiple responses at once, low-cost, regularly updated [124]	Refuses answers to fictional questions [125]
Anthropic	Claude v1	Large context window [124]	High level of hallucination, spread of inaccurate information [126]
Cohere	Cohere Command	High accuracy and robustness [127]	Relatively expensive [124]
TII	Falcon	Open source, can be used for commercial purposes [124], ranked first on OpenLLM leaderboard [99]	Relatively low number of parameters [128]
Meta	LLaMA	Open source, wide potential when fine-tuned [124]	Released for research only [124]

Whereas some of the models presented in the overview perform exceptionally well, this level of advancement can in some cases have negative impact on the response time. By a well-prepared phase of fine-tuning to the specific domain of *Formula One*, a smaller model that would be unsuitable for more general tasks could be a better fit here, as the speed of processing is likely to benefit and the risk of including information not related to the racing field is likely to decrease. Aspects such as the number of parameters the model is trained on, however, can be important for the variety in used language structure, as a high number of parameters means the model can learn complex language patterns in a more easy manner.

Considering the limited computational resources and the requirement for an open-source model, Meta's *LLaMA* and TII's *Falcon* are, based on purely the theoretical study, considered as the best options among the CLM based models for the blog generation task. However, these models are only available with a relatively large number of parameters, starting at 7 billion, which is likely to be infeasible due to computational limitations. Preference goes, therefore, to working with smaller models that are pre-trained for a task similar to blog generation.

Analysis of blog structure

SA.1 "Catching both Mercedes drivers is Sainz in P7. The Ferrari man is just over one second behind the pair."

Live data:

- ▶ interval to leader in current and previous few laps (all three drivers)
- ▶ current position (all three drivers)

Knowledge:

- ▶ team name (all three drivers)
- ▶ driver name from driver id (Sainz)

SA.2 "Perez is back into second after his slow pit stop. 10 laps to go and this is all up for grabs!"

Live data:

- ▶ current position (Perez)
- ▶ pit stop duration (Perez)
- ▶ start position (Perez)
- ▶ current lap

Knowledge:

- ▶ total number of laps in race
- ▶ average pit stop duration
- ▶ driver name from driver id (Perez)

SA.3 "Norris has cleared out of DRS range of Hamilton as his hard tyres come into their own against the softs. Some excellent defending has paid off for the McLaren driver to push on in second place."

Live data:

- ▶ interval to leader in current and previous few laps (both drivers)
- ▶ current tyre compound (both drivers)
- ▶ current position (both drivers)

Knowledge:

- ▶ team name (both drivers)
- ▶ driver name from driver id (both drivers)

SA.4 "Red flag! The race has been stopped by this rain!"

Live data:

- ▶ race control message (red flag)
- ▶ weather information (rain)

SA.5 "Verstappen pits from the lead again to swap inters for full wets as the virtual safety car is deployed."

Live data:

- ▶ current position (Verstappen)
- ▶ pit status (Verstappen)
- ▶ previous tyre compound (Verstappen)
- ▶ current tyre compound (Verstappen)
- ▶ race control message (virtual safety car)

Knowledge:

- ▶ driver name from driver id (Verstappen)

SA.6 "Perez pits from second place, needing one hell of an undercut to get back into the lead given the gap had grown to over eight seconds."

Live data:

- ▶ current position (Perez)
- ▶ pit status (Perez)
- ▶ gap to leader in current and previous few laps (Perez)

Knowledge:

- ▶ driver name from driver id (Perez)

SA.7 "Zhou drops back again, as he moves out of the way and concedes P11 to Piastri."

Live data:

- ▶ current position (both drivers)
- ▶ previous position (both drivers)

Knowledge:

- ▶ driver name from driver id (both drivers)

SA.8 "With a strong tow and DRS in action, Hamilton moves past Zhou and into the top 10."

Live data:

- ▶ current position (both drivers)
- ▶ previous position (both drivers)
- ▶ interval before position swap

Knowledge:

- ▶ driver name from driver id (both drivers)

SA.9 "Gasly is now stuck behind Alonso with the Alpine driver swarming all over the Aston Martin's gearbox."

Live data:

- ▶ interval to leader (both drivers)

Knowledge:

- ▶ team name (both drivers)
- ▶ driver name from driver id (both drivers)

SA.10 "The trio at the front have pulled a small gap of 3.6s on Leclerc in fourth, who still has Russell tucked up behind him."

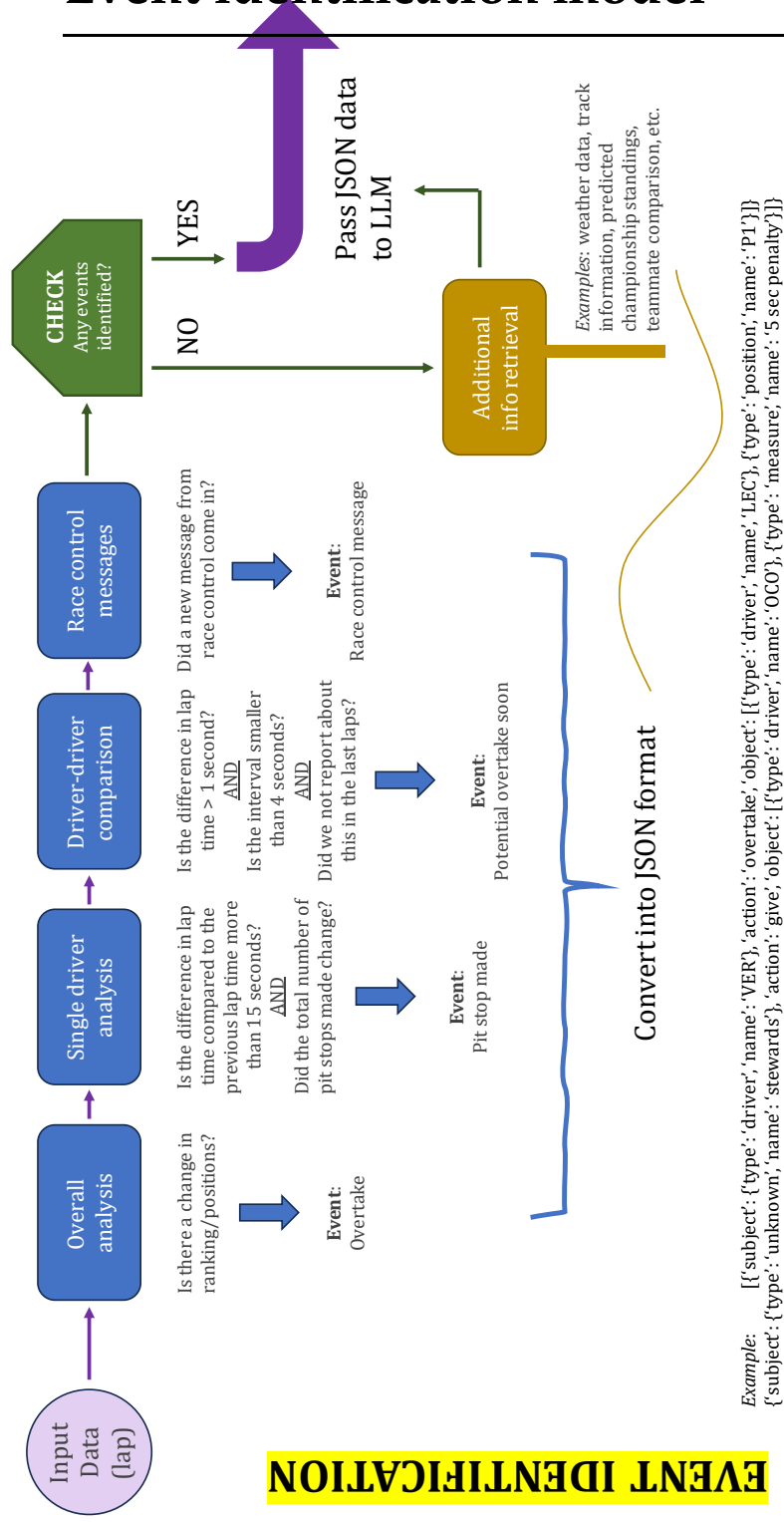
Live data:

- ▶ interval to leader (top five drivers)

Knowledge:

- ▶ driver name from driver id (Leclerc and Russell)

Event identification model



H

Experimentation

H.1 Structured data extraction with an LLM

Blog post	Result
"Max Verstappen just overtook Leclerc after the Safety Car restart"	<pre>{"driver": {"name": "Max Verstappen", "country": "Red Bull", "team": "Red Bull Racing"}, "car": {"name": "Ferrari", "country": "Italy", "team": "Scuderia Ferrari"}, "event": {"type": "race", "date": "2021-07-25", "start": "7:30pm", "duration": "90 minutes"}, "race": {"name": "Monaco Grand Prix", "country": "Monaco", "track": "Circuit de Monaco"}}</pre>
"Sergio Perez already gained 6 positions in 20 laps after overtaking Albon"	<pre>{"position": 6, "driver": "Sergio Perez", "team": "Red Bull", "event": "Australian Grand Prix", "year": 2020}</pre>
"Ferrari just pitted Sainz for the second time this race"	<pre>{"pitting": {"driver1": {"driverName": "Sainz"}, "driver2": {"driverName": "Sainz"}}</pre>
"After an extremely fast pitstop Hamilton managed to end up in front of his team mate in P5"	<pre>{"event": "Pitstop", "driver": "Lewis Hamilton", "team_mate": "Vettel", "position": 4}</pre>

Table H.1: Results of data extraction experiment with Falcon LLM (1)

Table H.2: Results of data extraction experiment with Falcon LLM (2)

Blog post	Result
"Max Verstappen just overtook Leclerc after the Safety Car restart"	<code>{"typeOfAction": "Overtake", "Drivers": [{"driver": {"name": "Max Verstappen", "team": "Red Bull Racing"}}, {"driver": {"name": "Charles Leclerc", "team": "Scuderia Ferrari"}}], "Teams": [{"name": "Red Bull Racing", "team": {"name": "Max Verstappen"}}, {"name": "Scuderia Ferrari", "team": {"name": "Charles Leclerc"}}]}</code>
"Sergio Perez already gained 6 positions in 20 laps after overtaking Albon"	<code>{"action": "Sergio Perez overtaking Albin", "drivers": {"Sergio Perez": [{"team": "Red Bull Racing", "driver": "Sergio Perez"}], {"team": "McLaren", "driver": "Lando Norris"}}, "Albin": [{"team": "Haas F1 Team", "driver": "Kevin Magnussen"}]}, "drivers": {"Sergio Perez": [{"position": 6, "team": "Red Bull Racing", "driver": "Sergio Perez"}], "Albin": [{"position": -2, "team": "Haas F1 Team", "driver": "Albin"}]}</code>
"Ferrari just pitted Sainz for the second time this race"	<code>{type: 'pits', driver1: 'Sainz', driver2: 'Vettel', team1: [1], team2: [2], advantage: [1, 2]}</code>
"After an extremely fast pitstop Hamilton managed to end up in front of his team mate in P5"	<code>{"event": "Pitstop", "type": "P1", "drivers": [{"driver": "Lewis Hamilton", "team": "Mercedes"}, {"driver": "Fernando Alonso", "team": "Ferrari"}, {"driver": "Sebastian Vettel", "team": "Red Bull"}], "teams": [{"team": "Mercedes", "drivers": [{"id": "LH", "driver": "Lewis Hamilton"}]}, {"team": "Ferrari", "drivers": [{"id": "FA", "driver": "Fernando Alonso"}]}, {"team": "Red Bull", "drivers": [{"id": "RB", "driver": "Sebastian Vettel"}]}</code>

H.2 Named entity recognition

1: Driver, driver in advantage, driver in disadvantage, team, team in advantage, team in disadvantage, timing (e.g., *last lap*), location on track, interval, overtake, near overtake, pitstop, incident, message, retirement, tyre compound, and weather.

In the first attempt, a NER model was trained to recognize 17 additional entity types¹. Below example involves one of the blogs with corresponding entity assignments that were used for training the NER model. 15 blogs with manually set named entities were used during this training process.

"**Hulkenberg** [DRIVER] has **stopped** [PITSTOP] for a new front wing after **contact** [INCIDENT] with **Perez** [DRIVER] as the **Red Bull** [TEAM] driver continued his recovery. **Perez** [DRIVER_ADV] is up to **P13** [POSITION] and is **chasing** [NEAR_OVERTAKE] **Ocon** [DRIVER_DISADV]."

As can be seen in the results of two test blogs shown in figure H.1, the trained NER model accurately recognizes some of the entities, especially in case of the shorter blog. Position *P11* is correctly identified here, while drivers *Gasly* and *Zhou* are also labelled along with the right (dis)advantage label. The model comes with some flaws, however. Timing entity *10 laps* is not recognized, while in the second example even more 'mistakes' are made. *Ferrari* should have been labelled as team rather than driver, while *middle sector* should have been recognized as location on track, *Leclerc* and *Verstappen* should have a (dis)advantage label in the first sentence, and for entity *Verstappen* in the second sentence the advantage label should be left out.

The probable main issue with blogs' NER performance is the limited size of training data and wide range of labels. Experimentation has been conducted with a narrower set of labels, including driver, team, position, timing, gap, tyre, and location, and repeated with the same training data. As can be seen in the new results (figure H.2), the labels seem more accurate compared to those in figure H.1. However, performance is still far from flawless. For example, *10*, in what should have been location entity *turn 10*, is recognized as a position, while team entity *Aston Martin* is mistakenly seen as driver. Furthermore, in all three examples of figure H.2, the turns (representing a location on

i The Jupyter notebook with the script for creating the NER training data can be found on GitHub [77].

After being stuck in P11 POSITION for 10 laps, Gasly DRIVER_ADV has finally managed to get past Zhou DRIVER_ADV to get into the point position.

Leclerc DRIVER has taken his advantage of Verstappen DRIVER back up to 2.5s now, as the Ferrari DRIVER_DISADV looks strong in the middle sector around here.

Verstappen DRIVER_ADV was told he's free to push, but only within the bounds of looking after his tyres for 20 laps.

Figure H.1: Results of the named entities detected in a subset of the blog examples (1)

track) are not labelled. This indicates that, although less labels are present, the amount of training data used is still not sufficient for achieving the desired results.

Hamilton DRIVER 's coming back at Alonso DRIVER though, but the Spanish veteran kept his former team-mate at arms' length into Turn 4. He'll try to stabilise now and then go after Sainz DRIVER .

Wow! Alonso DRIVER tracks Hamilton DRIVER through the opening sector, goes very wide into Turn 10 POSITION , and cuts through on the tight left-hander to grab fifth POSITION ! Lovely battle.

Alonso DRIVER 's all over Hamilton DRIVER like a rash, and gets past at Turn 4 - but Alonso DRIVER gets a wiggle on the exit and Hamilton DRIVER nips back past! The Aston Martin DRIVER driver looks like he'll get another go soonish...

Figure H.2: Results of the named entities detected in a subset of the blog examples (2)

After doubling the training examples, as shown in figure H.3, both drivers, the tyre compound, the position, and the team are all correctly labelled. Although most of the blog posts the retrained model was tested on were accurate, some inconsistencies could still be noticed. When, e.g., a *five-second penalty* is mentioned in a sentence, the entity *five-second* gets labelled as gap. Deploying more linguistic feature extraction techniques, however, can help in connecting these gap entities to their corresponding noun to catch such errors.

Stroll DRIVER 's in from third POSITION now, and collects the hard TYRE tyre. He rejoins behind Russell DRIVER , albeit four GAP seconds behind the Mercedes TEAM .

Figure H.3: Results of the named entities detected by a further trained NER model

H.3 Dependency parsing

Real examples from the blog post data set were used to test the dependency parsing concept introduced on page 35. The method extracting the root, subject, and attribute worked fine for simple sentences, but resulted in incomplete information for more complex sentences, as shown in below examples.

"De Vries, who didn't pit with the midfield runners during that brief virtual safety car period, is having his race ruined as he drops behind Tsunoda, Zhou and Sargeant on his aging hard tyres. The Dutchman is now 14th."

- ▶ {'root': having, 'subject': Vries}
- ▶ {'root': is, 'subject': Dutchman, 'attribute': 14th}

"But it is also race over for Ocon who retires from the race in the pits. He joins Leclerc and Norris out of the action early."

- ▶ {'root': is, 'subject': it, 'attribute': race}
- ▶ {'root': joins, 'subject': He, 'object': Leclerc}

In the first example, there is no object nor attribute present that is connected to the root (i.e., *having*). Here, the root verb (and its wider context, *having his race ruined*, is not necessarily the data that would be retrieved from the structured input data, as it is simply the *result* of the driver dropping behind, which would be a conclusion that can be drawn by the LLM itself. Therefore, there can be concluded that finding the root of the sentence does not directly mean that it helps to explain the event or action discussed.

To get to a more appropriate representation of the data, inclusion of more dependencies and other actions than solely the root verb is needed. Considering that this approach requires some of the other LFE methods as well (e.g., PoS tagging for finding the actions). Therefore, further experimentation with dependency parsing is discussed in the subsection hereafter.

I

Evaluation survey

I.1 Content

The following subsections present the content of the survey used for evaluating the level of entertainment of the generated blog posts in comparison to human-written ones.

Introduction

In this survey, you will be asked to rate some sequences of Formula One blog posts to help me get results for my Master thesis. Filling out the questionnaire will take approximately 5 minutes and your response is fully anonymous.

For each of the blog sequences, imagine that you are not watching the race and you want to read the text to stay updated about what is going on during the event.

Please rate the level of entertainment, information, and clarity of the posts purely on what you are reading, without taking into account knowledge you potentially already have about the race.

Sequence 1

This sequence contains automatically generated posts (non-rephrased). Sainz wants to overtake Verstappen, but the gap is 0.85s. Magnussen takes DRS and Ricciardo is 12th.

Ricciardo has come in to pit with a lap-time of 99.926s.

Albon has lost 16 positions since the start of the race. His current position is 18 on lap 34.

Ricciardo gets the blue flag on lap 35.

Schumacher passes Magnussen for ninth, taking 10th place. Latifi moves past Stroll for 11th.

Ricciardo pits for tyres, while Tsunoda also makes a pitstop.

Stroll has slowed down massively, with a lap-time of 126.48s.

The stewards have waved a double yellow flag for sector 9 and sector 10. The safety car is deployed on lap 38.

Sequence 2

This sequence contains automatically generated posts (rephrased by the RPM). Gasly, in a desperate bid to gain an advantage, dives into the pits for a set of hard tyres, but this decision backfires as he plummets down the order to 11th place. Leclerc, sensing an opportunity, also makes a pit stop for hard tyres, but his gamble doesn't pay off either as he slips to 6th position. Meanwhile, Hamilton, confident in his strategy, stays out.

Stroll's Aston Martin car has come to a grinding halt at the beginning of the lap, causing the race to be temporarily halted.

Carlos Sainz, the Spanish racing driver, made a daring move to overtake Frenchman Esteban Ocon, resulting in Sainz securing the sixth position on the track.

Stroll is set to stroll all over Lawson, who could potentially lose his spot on the grid to the Canadian.

Russell pulls into the pitlane for a set of hard tyres, but the stop costs him a few places and he emerges in seventh.

Hamilton, like a stealthy ninja, silently swoops in and snatches the position from Ocon, leaving the Frenchman in the dust.

Alonso decides to switch to hard tyres during the pit stop, causing him to drop down to 12th place. Meanwhile, Lawrence makes a strategic move by pitting from 12th position to undercut Tsunoda, who is still out on the track.

Sequence 3

This sequence contains human-written posts. A five second penalty for Perez for track limits. He is the first driver to be caught out tonight!

Norris, in second place for a lap, is next in and ditches his mediums for another set of fresh mediums. He comes out in seventh, just behind Zhou but ahead of Alonso!

Verstappen still leads, with his advantage up to 23s, with Russell now the lucky recipient of second place.

The Norris-Alonso battle has reached boiling point yet again, with the pair in close quarters after their latest stops.

Gasly now gets the black and white warning flag for track limits. He, Albon and Lawson are the three drivers pushing their luck right now.

Sequence 4

This sequence contains automatically generated posts (non-rephrased). Perez makes his pitstop.

Alonso moves Leclerc past to take seventh, as Albon stops for tyres. Zhou also changes his tyres, with a lap of 94.117s.

Perez has moved past Sainz into 15th.

Albon might overtake Hulkenberg soon.

Magnussen pits on lap 20, while Lawson and AlphaTauri also stop for tyres.

Russell is now in the DRS range. Perez is 8th, with a gap of 0.54s.

Sequence 5

This sequence contains human-written posts. Perez has moved up to tenth at the expense of Albon.

This is incredibly tense, as Sainz still leads with three Brits directly behind him.

Russell looks for a way past Norris but is shown no way through. Three laps to go.

Albon had tumbled back to 14th, in an incident with Perez that is being investigated by the stewards. He regains one spot at the expense of Zhou.

That Russell vs Norris squabble has allowed Sainz to sneak off ahead, momentarily, but Norris is still in DRS range so it is short lived.

The incident between Perez and Albon will be investigated after the race. Meanwhile a soft-shod Magnussen has passed team-mate Hulkenberg for 11th, Albon following him through.

Sequence 6

This sequence contains automatically generated posts (rephrased by the RPM). The stewards have thrown down the caution flag in sector 2 due to the dust-up between Russell and Verstappen.

Perez blazes through the track like a bolt of lightning, setting a scorching new fastest lap of 105.311 seconds!

Bottas, starting from 19th place, decides to make a pit stop in order to undercut Sainz, who is yet to come in.

Tsunoda, starting from the 10th spot, manages to pull off a slick undercut move on Russell, who is still stuck in the pits. Piastri, on the other hand, decides to switch to hard tyres and finds himself dropping down to the 16th position. Sainz, not to be outdone, makes a pit stop of his own.

Norris, like a cheetah on the prowl, pounces on Hulkenberg, sending him tumbling down the rankings like a sack of potatoes. Norris then leaps into 10th place, leaving Hulkenberg in the dust.

Leclerc blazes through the track like a lightning bolt, setting a scorching new fastest lap of 104.67 seconds!

Bottas dives into the pits like a torpedo, emerging with medium tyres and a new lease on life. He re-enters the race in 19th position, ready to make his move up the grid.

I.2 Results

Table I.1 shows the results of the Jarque-Bera test, which was performed on each of the samples to assess whether the data is normally distributed. The degrees of freedom used in the formula is 2, and a p-value of 0.05 is taken. As can be obtained in the far-right column, this lead to the rejection of the null-hypothesis, i.e., that the data is normally distributed, for only a few of the samples.

In tables I.2 and I.3, the resulting ranks of the Mann-Whitney U-test, on level of informativeness and clarity respectively, can be found. The corresponding critical value (with $p < 0.05$) is 192.

Table I.1: Results of the Jarque-Bera test on the evaluation survey results

	Sample size	Skewness	Kurtosis	JB	p-value
<i>Entertaining</i>					
HW	48	-0.615	1.902	10.265	0.006
AI-O	48	0.229	-0.693	1.380	0.501
AI-R	48	-1.123	0.744	11.192	0.004
<i>Informative</i>					
HW	48	-1.118	1.202	12.885	0.002
AI-O	48	-1.386	2.294	25.896	0.000
AI-R	48	-1.013	1.521	12.838	0.002
<i>Clear</i>					
HW	48	-0.976	0.236	7.740	0.021
AI-O	48	-0.538	-0.675	3.225	0.199
AI-R	48	-0.577	-0.396	2.972	0.226

Table I.2: U-values of the Mann-Whitney U-test on level of informativeness

	HW 1	HW 2	AI-O 1	AI-O 2	AI-R 1	AI-R 2
HW 1	-	248	233	285	241.5	287
HW 2	-	-	192	252	200	250
AI-O 1	-	-	-	232	280.5	235
AI-O 2	-	-	-	-	239.5	285.5
AI-R 1	-	-	-	-	-	287
AI-R 2	-	-	-	-	-	-

Table I.3: U-values of the Mann-Whitney U-test on level of clarity

	HW 1	HW 2	AI-O 1	AI-O 2	AI-R 1	AI-R 2
HW 1	-	270	280	270	281	279
HW 2	-	-	266	285	280.5	281.5
AI-O 1	-	-	-	263	272.5	270.5
AI-O 2	-	-	-	-	279	280
AI-R 1	-	-	-	-	-	286.5
AI-R 2	-	-	-	-	-	-