Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 3 (2023)

A Review on Text Summarization to Develop a Live Sports Match Journalism System Using NLP

[1]Kaustav Sanyal, [2]Vinay Tiwari, [3]Mohammad Saad, [4]Shankar Nag

[1]Assistant Professor, Department of Computer Science & Engineering, Durgapur Institute of Advanced Technology & Management, West Bengal.
[2][3][4]Student, B.Tech, Department of Computer Science & Engineering, Durgapur Institute of Advanced Technology & Management, West Bengal.

Abstract: The review paper details the process of converting live speech commentary into text and subsequently generating a text summary from this transcription. Natural Language Processing (NLP) serves as the most suitable technique for this task. Sporting events attract widespread global attention, with many individuals preferring news articles over live experiences. Consequently, there's a pressing need for rapid news production following the conclusion of any sporting event. Due to constraints such as time and budget, comprehensive reporting by commentators or journalists becomes impractical. Therefore, this project holds significant promise and effectiveness. The primary aim is to automate the creation of text summaries from live sports commentary. To achieve this, a system has been devised to generate summaries from live game commentary, an area that has seen minimal focus in current research. An audio recognition system will process provided audio data, transforming it into a lengthy text format. Subsequently, a text-summarising technique will condense this transformed text into a summary.
Keywords: Natural language processing, Natural language toolkit, Speechrecognition, Text summarisation.

1. Introduction :

In the rapidly evolving landscape of technological advancements, the exponential growth in data collection encompasses diverse types of information worldwide. Among these diverse data types lies the collection of news summaries related to sports matches. Sporting events present a captivating experience best felt in real-time—whether by witnessing the action firsthand in the stadium, watching it unfold on television screens, or immersing oneself in the live commentary that vividly captures the essence of the event. These mediums convey not just the game's events but also the emotions, thoughts, and raw expressions that accompany them, significantly enriching the spectator's experience. However, despite the immersive nature of live experiences, a considerable majority of enthusiasts opt for alternatives such as televised broadcasts or live commentary over physically attending the match. These mediums, through skilled communication, vividly articulate the ongoing events, painting a mental picture for the audience, albeit in a remote fashion. Yet, intriguingly, some individuals prefer the convenience of post-match news summaries over these live experiences, often due to constraints such as time or the inability to navigate through lengthy textual commentaries. This preference for concise summaries over detailed textual accounts of the game highlights a notable gap that our project aims to bridge.

The central objective of our project revolves around automating the process of generating news content from live sports commentary, thereby mitigating the need for labor-intensive and time-consuming manual journalistic work. By leveraging innovative technology and methodologies, we intend to streamline this process while significantly reducing associated costs. To achieve this goal, our project is structured into two principal phases: Speech-to-Text and Text Summarisation. Both phases necessitate a deep understanding and application of Natural Language Processing (NLP), a field crucial in handling linguistic complexities and nuances.

The Speech-to-Text phase involves the efficient conversion of live broadcast commentary into a textual format through advanced speech recognition techniques. However, this transformation isn't devoid of challenges; factors like interruptions, speech disfluencies, and abrupt transitions necessitate the implementation of de-noising techniques to refine and effectively utilise the transcribed speech data. Once the speech data is transformed into textual form, the subsequent phase involves the intricate process of text summarisation. Text Summarization represents the core essence of our project, aiming to distill the expansive textual commentary into concise and digestible news summaries. This phase ensures that while condensing the content, no vital information is lost,

filtering out redundant or inconsequential details. The text summarization process bifurcates into two primary methodologies: Extractive Text Summarization and Abstractive Text Summarization.

Extractive Text Summarization involves the identification and selection of crucial sentences or phrases from the textual commentary, amalgamating them to construct a coherent and compact summary. This method serves diverse purposes, including time efficiency, alleviating information overload, and highlighting pivotal aspects of the commentary.

On the other hand, Abstractive Text Summarization represents a more intricate approach. It involves generating brief summaries by synthesizing essential words or phrases without introducing extraneous content. This method poses a greater challenge and complexity compared to its extractive counterpart, requiring a deeper comprehension of context and linguistic nuances.

In essence, our project embodies a convergence of technological innovation and linguistic prowess, aiming to cater to the preferences of individuals seeking concise yet comprehensive insights into sports events without compromising on essential information. Through the seamless integration of Speech-to-Text and Text Summarization, we endeavor to deliver efficient and accessible news summaries derived from live sports commentary.[1]

2. Literature Review:

This chapter is composed of two sections. The first section has thoroughly defined the technologies on which the proposed system is built, while the second section contains a survey of the related system for automated journalism.

Natural Language Generation:

Natural Language Generation or NLG is a branch of artificial intelligence or computational linguistics. It focuses on generating natural language text or speech in any language from structured data or input. In the 1970s, one of the main focuses of Natural Language Processing research was trying to identify users' opinions, objectives, and plans in order to achieve a proactive and extended interaction between users and expert systems for consultation and command, where the system's responses should be collaborative. The NLG field has been in development ever since, and it is not surprising that the first NLG systems that translated data into very simple texts with little or no variation started to appear, such as advice-giving systems or synthesized weather forecasts. NLG can be regarded as an emerging technology since many real-world applications have been built based on this technology. From a research standpoint, NLG is a subset of Natural Language Processing.

According to Input of the system NLG can be classified into two types. Those classified types are D2T (Data to Text) and T2T (Text to Text) [2]

Data to text: The normal input for D2T is structured data, which would come from a database, knowledge base, labeled corpus, etc. Robot journalism is one good example of a D2T system. They have a considerable impact in the fields of journalism and media studies.

Text to text: The input for the T2T system will be texts or isolated sentences, and it includes learning how to express a piece of data in different or creative ways. Abstracted summarization systems are one of the best examples of T2T. [2]

According to the communicative goals of a system NLG can be classified as:

Informative Text: The purpose of the system is to generate informative texts from factual data. Fog and SunTime create weather forecasts, taking as input numerical information from simulation systems that represent parameters like temperature, precipitation level, and wind speed from different places and different hours of the day. Another example is SkillSum [3], a system that generates basic skills reports to help people with poor basic numeracy and literacy skills.[2]

Textual summaries: This type of system produces textual summaries from one or more data sources. This summary can be associated with different fields: engineering [4], financial summaries [5] and sports [6], among others.

Simplified text: Systems that aim to help people with oral or writing problems, derived from cognitive difficulties or language barriers. Some examples are systems that produce text to help aphasic people or systems that allow visually impaired people to examine graphics.

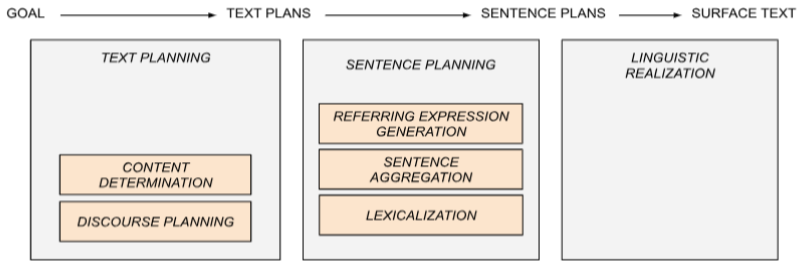

Fig 1: NLG system architecture proposed by Reiter and Dale [12]

Persuasive Text: Systems that try to persuade or take advantage of the user's emotional state. Examples of these systems are: STOP, a system that generates tailored smoking cessation letters; and systems that aim to decrease anxiety in cancer patients by giving them personalized health information.[2]

Dialogue system: The dialogue system's main purpose is to improve human-machine communication. Users interact directly with the system, which creates sentences conditioned by the previous context. There are lots of applications, such as: automatic question generator system for academic writing support or the adaptable tutorial dialogue system to improve knowledge on certain subjects.[2]

Explanations: The output of this system is an explanation of the steps that the system went through to execute an algorithm, process a transaction, or solve a mathematical problem. P. Rex is an example of a natural language proof-explanation system.

***Recommendations*:** Systems that create recommendations by processing information related to users' preferences and opinions. Shed is an online diet counseling system that provides personalized diets based on the user profile.[7]

Design of an NLG System:

We can describe an NLG system as a group of tasks that transmit information to a certain audience to achieve a specific goal in natural language. Therefore, characterizing the input, the tasks, and the output of the system is as important as specifying its context and communicative goal. In order for the system to successfully achieve its purpose, each task should cover those aspects. The design of NLG systems is an open field where a broad consensus does not exist. There is a diversity of architectures and implementations that depend on the problem for which the NLG system was created. It's hard to identify common elements and provide a complete abstraction that is applicable to most NLG systems. However, a lot of effort has been put into trying to define a general architecture for the context of NLG. As our interest is to have a starting point that helps to us determine what tasks compose an NLG system in order to build a system that generates summaries of football matches, after reviewing the literature, we came to the conclusion that the most consensual architecture is the one proposed by Ehud Reiter and Robert Dale. According to them, an NLG system performs seven tasks, and the interaction between them can be represented by a three-module architecture.[2][8] (Figure .1)

NLG Task:

While converting the input to a human-readable form of text,six NLG tasks have been pinpointed in the majority of NLG systems.They are listed as follows:

Content Determination: In this phase, it determines the details to include in the text. It screens out the most crucial information to be presented in the final output text. This includes options. In Cricket, we might not have to mention everything about each single delivery. Content determination is available in most NLG systems, and the approaches related to this task are closely related to the domain of the application.

Text Structuring: This activity will decide the sequence of information in the text. The application domain sets limitations on ordering preferences. In the field of cricket, it's advisable to commence with general details like the match's date, location, toss, and opening batsmen before delving into specifics like special overs, hits, deliveries, and balls in the match.

Sentence Aggregation: There is no need to show every message in separate sentences. By combining multiple messages into a single sentence, it would be more fluid and readable. At the same time, there are instances where aggregation should be avoided. For example, in the cricket domain, when a player gets a hat-trick, it is better to express it in one sentence than to express it in three separate sentences for each out.

Lexicalization: Lexicalization involves discovering the appropriate words and phrases to convey the information. After the aggregation is completed, the sentence's content can be transformed into natural language. The complication of lexicalization is that the same event can be expressed in several ways. In the cricket domain, if an event like a player took a catch, many journalists would report this in different ways while giving the same meaning to the reader.[3]

Referring to expression generation: This action would choose the words and phrases to recognize domain entities. This would avoid ambiguity. According to Reiter and Dale, the difference between lexicalization and REG is that REG will contain the function of a "discrimination task", where the system needs to communicate sufficient information to distinguish one domain entity from other domain entities.[3][8]

Linguistic Realization: This task encompasses structuring sentence elements and creating the correct morphological forms. Here, it would form a well-formed sentence. One of the challenges here is that it would contain various linguistic components that may not be present in the input. This would convert the sentence to the right morphological form.[3]

There are different approaches to linguistic realization. i.e., human-crafted templates, a human-crafted grammar-based system, and a statistical approach.

The above-mentioned steps can further be categorized into three modules. They are document planning, microplanning, and realization.

NLG TOOLS:

There are various technical tools that can be applied in the NLG process, but there are tools used mainly in the part where the computer generates language. This happens after it understands the information. Different programming languages support different tools for this. Here are some examples:

NLTK (Natural Language Toolkit):

NLTK is a free and open program that includes modules and pre-built, ready-to-use computational linguistic course materials. It covers a wide range of symbolic and statistical natural language processing. This toolkit is mainly designed to be used with the Python language. It provides easy-to-use interfaces for tasks like tokenization, stemming, tagging, parsing, and more. The user-friendly learning experience of this toolkit enlightens one of the primary purposes of this toolkit. [10]

PyNLPI:

PyNLPI is a Python library for NLP and offers tools for tasks like tokenization, stemming, and part-of-speech tagging. This has several packages and modules and is licensed under the General Public License.[9]

Natural Owl:

Natural Owl is a free and open multilingual natural language generator that produces language-annotated OWL knowledge structures. In the semantic web, OWL serves as a standard language for defining ontologies. The structure of Natural OWL involves a pipeline, where the generation process unfolds across two stages referred to as the document planning stage and the microplanning stage. In the document planning stage, the selection process for Natural OWL involves identifying all ontologies that have a direct mapping to the current instance. In the microplanning stage, each language is allocated to one or more unfilled template-structured micro-plans.[3]

Simple NLG:

Simple NLG is an NLG tool used to generate natural language text. It helps in forming sentences with the correct structure and organizes them in a sequential manner. It is mainly created for large-scale data-to-text NLG systems for summarizing those large volumes. Simple NLG is a Java library that specifies a range of lexical and phrasal types.[6]

Automated Journalism to sports domain:

Sports represent an optimal domain for automated journalism due to abundant data availability and the established format and style in news reporting. Numerous sports, including baseball, football, and cricket, are

subjects of ongoing research in automated sports journalism. The game's recap of Little League Baseball was provided by the media coverage of Associated Press (AP). Utilizing an artificial software from Automated Insights with the data supplied by MLBAdvanced Media (MLBAM), the official statistics provider, results in the creation of this robotic reporter. There are similar applications and researches and some of them are listed below.[9]

GoalGetter:

GoalGetter is a system that converts data into spoken Dutch summaries for football matches. To produce summaries, it processes data presented in a tabular format, similar to STREAK [7], a system that generates basketball summaries rather than football.

STREAK creates summaries through a revision-based approach to summarization. In the initial pass, it generates a draft with fixed information such as winners and losers. In the second pass, it opportunistically incorporates additional information, as permitted by the structure of the existing text. GoalGetter differs from STREAK in that STREAK creates a written output, whereas GoalGetter generates a spoken output. GoalGetter obtains information from Teletext, which broadcasts text via the television signal and decodes it at the receiver end. A key distinction from other NLG systems is its deviation from the pipeline architecture. The generation process and prosody involve three additional sources apart from domain data, including syntactic templates, knowledge state, and context state.[9]

The generation module incorporates the fundamental generation algorithm of LGM. Teletext data will be parsed to a machine-readable format and will be input to the generation module. Additionally, to this input it needs domain data which contains fixed background data on the relevant domain. For example, in GoalGetter those are the data about the football teams and the players. At the same time generation modules need syntactic templates which are syntactic tree structures containing open slots for variable information. Each template is employed under specific conditions, and the interaction between these conditions dictates the produced text. Two records are kept during generation. The knowledge state indicates which section of the input data structure is conveyed by the system and which part is not. This is achieved by labeling all fields in the input data structure. The alternative record is the Context State, which documents diverse aspects of the linguistic context. [9]

GameRecapper The GameRecapper system is a template-based system that generates Portuguese summaries of football matches from structure input data such as game sheets taken from www.zerozero.pt website. This data is converted into a JSON tree, and the

implementation of GameRecapper is also based on the GoalGetter system.[9]

Previous research based on this system:

Haojie Zhuang et al. [16] proposed a model. The model involves three main elements: a generator for encoding unprocessed text into a shorter format, a discriminator to train the generator in generating readable summaries, and the use of CNN/DailyMail details collection for concentration on summarizing specific official papers. The model incorporates abstractive techniques based on deep learning, including encoding data with CNN, generating summaries with a feed-forward neural network, and using RNN to combine extractive and abstractive techniques. The GAN architecture includes a generator creating data examples and a discriminator distinguishing between real data and generated examples. The aim is for the generator to produce samples most similar to real data. Future studies plan to combine this GAN with LSTM-CNN based deep learning. Additionally, there's a proposal to substitute the generator with a pre-trained BERT model, processing input phrase-by-phrase to enhance accuracy.

Surbhi Bhatia et al. [4] is providing relevant details on abstract methodologies and extractive techniques. The abstractive summarization technique utilizes a graph-based method for handling duplicate sentences. This involves tasks like constructing graphs, ensuring sentence accuracy through applied constraints, and independently scoring each sentence. Sentiments are merged using SentiWordNet during the scoring process. In extractive summarization, the principal component analysis (PCA) method is employed. This method reduces data proportions and creates a synopsis based on ranking the most pertinent statements.

The details regarding opioid use on Garmin Nuvi 255W GPS. The present abstractive approach enhanced performance by 13%. Through ROUGE score, the outcomes of both methods are assessed.[1]

Research Gap:

In the related search from speech to text conversion, a notable gap exists in addressing the issue of background noise, specially generated by crowds in a stadium. Existing research does not include challenges posed by such noise which can affect the accuracy of speech recognition systems. This challenge imposes complexity while the conversion of spoken words into text. To tackle this problem, the models need more development in robust algorithms that are capable of handling the crowds noise, especially in sports commentary scenarios.

Another crucial area for improvement in this domain is the handling of advertisements during a live commentary. Commentators often involve sponsors in their commentary, or there are some advertisements inserted during the ongoing match.This imposes a challenge in speech to text summarization as the system needs to exclude those unnecessary parts.There is no research done that will detect those unnecessary parts and remove them. Developing these models is crucial to improve the accuracy while transforming the speech into text form.

Overall, these gaps highlight opportunities in enhancing speech-to-text conversion in real-world, noisy settings and improve their adaptability in handling diverse content structures.

3. Conclusion:

In this research work, we implemented a challenging task, which is the news generation of a sports match using live commentary. This project helps us reduce the work of journalists and also cuts the budget. Journalists have to do in-depth research work to get the summarized version of a sports match. Automatic news generation will help ease this. It will help reduce the reading time by providing a summarized textual version of an extensive text commentary. The purpose of this project is to recognize the speech and represent it in lengthy text form. Then create a summarized version of the text using that lengthy text. The work is implemented using some techniques from Natural Language Processing. Some major challenges will be faced while doing this project, including background noise removal and the recognition of each word with a different accent during speech recognition. We have to overcome the denoising phase in this project as it poses a critical challenge. In light of the accomplishments, it is crucial to recognize the persistent challenges and opportunities for enhancement. Refinements and updates to the models will overcome the issues and increase the overall performance. The automatic news generation can also be used on other occasions.

4. Future Work:

There are upcoming initiatives in this project that we plan to undertake in the future. Some of the live sports commentary features commercials interspersed between them. Our main goal is to get a summary of a sports match without compromising crucial information. This includes removing unnecessary dialogue, background noise, cross-talks, and excessive advertisement segments. Additionally, the noise reduction aspect of this project requires further refinement to achieve more precise results. These are the developments that are needed in the project to obtain the news synopsis with greater precision and quality.

References

1. ”speech to text conversion and summarization”international journal of emerging tech- nologies and innovative research (www.jetir.org), issn:2349-5162, vol.10, issue 4, page no.c463-c470.
2. João Pinto Barbosa Machado Aires. Automatic generation of sports news. 2016.
3. Ion Androutsopoulos, Gerasimos Lampouras, and Dimitrios Galanis. Generating nat- ural language descriptions from owl ontologies: the naturalowl system. *Journal of Artificial Intelligence Research*, 48:671–715, 2013.
4. Surbhi Bhatia. A comparative study of opinion summarization techniques. *IEEE Transactions on Computational Social Systems*, 8(1):110–117, 2020.
5. Albert Gatt and Emiel Krahmer. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170, 2018.
6. Albert Gatt and Ehud Reiter. Simplenlg: A realisation engine for practical applications.In *Proceedings of the 12th European workshop on natural language generation (ENLG2009)*, pages 90–93, 2009.
7. Eli Goldberg, Norbert Driedger, and Richard I Kittredge. Using natural-language processing to produce

weather forecasts. *IEEE Expert*, 9(2):45–53, 1994.

8. Andreas Graefe. Guide to automated journalism. 2016.
9. MHDY Gunasiri. *Automated cricket news generation in Sri Lankan style using natural language generation*. PhD thesis, 2021.
10. Haojie Zhuang and Weibin Zhang. Generating semantically similar and human- readable summaries with generative adversarial networks. *IEEE Access*, 7:169426– 169433, 2019.
11. Alejandro Ramos-Soto, Alberto Bugarín, and Senén Barro. On the role of linguistic descriptions of data in the building of natural language generation systems. *Fuzzy Sets and Systems*, 285:31–51, 2016.
12. Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.
13. Jacques Robin and Kathleen McKeown. Empirically designing and evaluating a new revision-based model for summary generation. *Artificial Intelligence*, 85(1-2):135–179, 1996.
14. Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. Plan- then-generate: Controlled data-to-text generation via planning. *arXiv preprint arXiv:2108.13740*, 2021.
15. Jin Yu, Ehud Reiter, Jim Hunter, and Chris Mellish. Choosing the content of textual summaries of large time-series data sets. *Natural Language Engineering*, 13(1):25–49,2007.
16. Edward Loper and Steven Bird. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.