Automating Sports Commentary

36118 Applied Natural Language Processing – Assignment 2A

Naveen Muralidharan, Nathan Collins, Aman Dalal, Yasaman Mohammadi

Team: Skynet

Assessment 2B:	Applying NLP Techniques
Type:	Group, Individual Assessment
Deliverables:	Working Model
	Consolidated Project Report
	Proof of Collaboration
Length:	10-15 pages
Weight:	100 pts, 40%
Due:	Monday, 15 May, 23:59

Assessment Criteria:

- *Synthesis of rational, NLP approaches used and outcomes for the project*
- *Communication of an engaging narrative with insights, value-added, and ethical considerations using appropriate language and visualise*
- *Demonstration of creative and innovative thinking in the choice of data, techniques and problem statements for real-world applications*
- *Demonstration of critical thinking, teamwork, and deep reflection with appropriate evidence for contributions made to the project*

Section 1: Project Planning

[1.1] Project Objective & Scope

The project objective is to develop a Python script that generates a real-time and informative sports commentary for televised cricket matches. The script will offer the functionality to select between an “emotive narration” of each significant cricket event or a simplified alternative. This will be carried out by analysing visual data from the field and the onboard statistics. The commentary will also provide a text-to-speech conversion, with additional translation access for conversion into common Indo-European languages.

This undertaking was chosen as it presented a genuine challenge. In achieving these goals, the final product may aid in communicating a cricket match to an audience unable to observe a live feed, where the translation feature could help expand the popularity of Cricket to countries where existing commentary is not provided. The final product may also aid in mitigating common barriers experienced by individuals with communication disabilities and serve as a proof of concept for expansion into other sporting industries.

[1.2] Initial Project Strategy

Data Collection:
Videography data that detailed player actions, umpire decisions and match statistics would be sourced, sorted and fed as input through applying openCV. A ball-by-ball commentary for each cricket event that transpired was to also be obtained via archived transcripts sourced from Cricbuzz & ESPN Cricinfo. By comparing the two data sources, an emotive language structure could be replicated.

Data Cleaning and Pre-processing:
Data would be cleaned for suitability prior to analysis and modelling by removing duplicates, NaN values and performing any necessary feature engineering. This includes concatenating the “Commentary” feature into a single **11,100-line** corpus, tokenising the corpus with the existing Large Language Model (LLM) GPT2 tokeniser and configuring an SVC classification model.

Feature Selection:
The feature of significance was determined to be the Commentary column.

Model Selection and Instantiation:
A pre-trained “TFGPT2LMHead” Model is to be primarily instantiated as the model of interest. By adjusting parameters such as “batch_size”, “epochs”, “learning_rate” and “warmup_step”, the linear model could be optimised.

< While the initial approach was to create a model that generated commentary based on the umpire decisions and player actions, this was adjusted to later scraping match statistics instead. Rational for this choice is explored in this report.>

Model Evaluation:
Grade the effectiveness and performance of the output text with either “Perplexity”, “BLEU” (Bilingual Evaluation Understudy) or “ROUGE” (Recall-Oriented Understudy for Gisting Evaluation). A BLEU test for example can be conducted through bleu_score within the NLTK library (Radford et.al, 2019).

Section 2: Understanding the Data

[2.1] Data Description

[2.1a] Text Data:

Raw commentary data was acquired from Kaggle, a scraped dataset derived from cricket match archives. The data frame consists of 11574 separate entries across 6 columns:

"**ID**", "**Match_id**", "**Team**", "**Over_num**", "**Commentary**", "**batsman**", and "**score**".

Feature Description:

ID:
Represents the cricket event entry identification as a unique value.

Match_id:
Denotes the corresponding 7-digit match identification number that the event and associated commentary describe.

Team:
A 2 to 3-letter acronym representing the name of the IPL team that the event and associated commentary describes. A total of 10 unique names were identified.

Over_num:
The "ball moment" in which the 9 unique events took place. A group of six deliveries is referred to as an "Over" and is defined as a float. The left side indicates the over number, and the right indicates the ball number for that over. For the feature Over_num, each set of six deliveries will be categorised by the occurrence of the event. For example, if the event resides in the first over, it is marked as "1st".

Commentary:
The primary feature of interest. This column will host the application of the majority of NLP techniques for analysis for model production to emulate. This feature lists the batsman feature, followed by the score and a short description of pitch and player descriptions intertwined with emotive expression and critique. For example: "*Chahar to Prithvi Shaw, FOUR, smashed! This is the length that Shaw enjoys a lot. Shorter from Chahar, and it's been dispatched with a nonchalant pull through square leg*".

batsman:
Two names separated by "to". The first name is described in full and represents the batsman, the second is the bowler who delivered the ball, denoted with the individual's last name.

score:
The final verdict of each event. This is categorised by first, the quantity of runs achieved by the batsman, such as "**no runs**" or "**2 runs**", or the final position of the ball, such as "**OUT**", "**FOUR** or "**SIX**".

Data Cleaning

The data frame was investigated prior to use and cleaned where appropriate. No duplicates or missing variables were identified. As the columns “Team” and “Batsman” were not compatible in their original form, annotation and feature engineering were performed, so to incorporate their use in the final commentary. This was achieved by converting the “Team” feature into the actual team’s name and storing the altered string in a new column “Team_new”. As the names of batsman and bowler were coupled, these were split to yield the first and last names. This facilitated the assigning of placeholders to these individuals, enabling cleaner model production.

[2.1b] Image Training Data:

Image data | “Umpire-verdict” models:

Assorted image data (**.jpg** files) consisting of diversified screenshots from 20 hours of televised cricket matches. The data consists of 390 “**umpire**” and 385 “**non_umpire**” stills applied to train the model to distinguish which figure onscreen represents the umpire.

Figure 1,2: Example of image data identifying an “Umpire” or “Non-Umpire” to determine outcome of play or event. (Left: non_umpire_236.jpg, Right: umpire_66.jpg).

The second model differentiates the umpire evaluations based on their signal or gesture. This includes “**no_action**” (78), “**no_ball**” (98), “**oneshort**” (2), “**out**” (102), “**penaltyrun**” (4), “**sixes**” (86), “**wide**” (99), “**bye**” (4), “**cancel**” (2), “**dead**” (4), “**four**” (9), and “**legbye**” (4).

Figure 3 Example of image data representing an “Out” verdict, determined by the umpire (out_61.jpg).

[2.1c] Image data | “Score-scraping” models:

Scraping the screen for details OCR (Py-tesseract and Open-CV)

Recordings (**.mp4** files) were also gathered from televised cricket broadcasts for scrapping the score, names, and statistics from the match. Each clip was acquired from “KAYO Sports” through Open Broadcaster Software (OBS).

> *All broadcast recordings will not be used outside of model training, ethicality in this method of collection must be considered, especially with refining the model* and proceeding with deployment.

To ensure result reliability, both the quality and framerate of each stream were maintained at maximum (**1080p** at **50fps**, extracting 50x 2.0-megapixel images every second), exercising further caution to ensure extraction quality paralleled the stream quality. As GitHub only permitted limited storage accessibility for uploads, video datasets were distributed with the team across Google Drive.

To further maximise efficiency in data retrieval, a reverse approach to this data collection method was applied. Score data was tallied from *CricBuzz* archives to acquire the sequence of significant events more efficiently than processing matches that often exceeded 4 hours in duration.

Scrapping was subsequently performed on these streams by collecting additional stills from recording through OpenCV. Analysis of these frames was processed through pytesseract, a Python toolset that facilitates optical character recognition (OCR) to ultimately interpret the scoring and associated player names displayed consistently throughout a match. This data was the final output allocated to the commentary model for content generation.

Rational for Model Preference:

The positioning and size of the rectangular scoreboard remained on screen throughout the majority of a match, ensuring consistency and reliability. With a contrasting background between the characters and a clean layout, this further reinforced the decision to apply this score-scraping model over the umpire-verdict model, as recognition of each character displayed would be clearer for interpretation.

Figure 4 Highlighted target area for OCR for data collection.

[2.2] Application of NLP & non-NLP methods

A variety of NLP (orange) and Non-NLP (green) methods and techniques were implemented throughout this project to extract information from the commentary dataset and construct models. These are explored below.

Dataset Exploration

Word cloud, **sentiment analysis** and **count** were applied to the commentary, verdicts, and name features of our data set – each revealing details that contributed value to each model. For example, 'FOUR', 'SIX' and 'WIDE' were the most frequent verdicts, insights into the commentator's choice of vocabulary and sentiment, and who were the most discussed batsmen and bowlers. This is explored in greater detail within the pre-modelling insights section of this report.

Umpire-verdict Model

From a series of images containing umpire signalling, information about each action was recorded and extracted by implementing **Keras inception v3**. An image classification model was subsequently built using **linear SVC** to recognise and discern posture. With the help of **openCV**, frames from match videos were acquired, where these actions and gestures were then classified using the model.

The model, however, failed due to a poor prediction rate, as televised content tended to place greater emphasis on the players and audience rather than the umpire – resulting in a lack of moments where the umpire was visible to capture and no data input.

Score-scraping Model

A subsequent approach involving recognising events and statistics through an on-screen scoreboard was adopted next. The primary input data stemmed from footage of televised IPL cricket matches. **OpenCV** was applied to extract stills from each recording, which were subsequently analysed through **OCR** to gather information about the score of runs, wickets, and player names. The package **Pytesseract** was then employed to extract this information and map events to determine a connection between each and the commentary data. The performance of this model improved significantly from the previous attempt, enabling the project to proceed to the final step of implementing the automated commentary portion.

Automated Commentary Model

Construction of this model began by implementing Google's **Bidirectional Encoder Representations from Transformers (BERT)** as a means for generating commentary. It was built through the transformers package "**BertForMaskedLM**", a package selected for its renowned capability and ubiquitous applications in NLP undertakings. Despite its reputation, the generated commentary wasn't coherent and did not meet the standards an audience would expect.

Another pre-trained and highly regarded model, **OpenAI**'s **Generative Pre-Trained Transformer 2 (GPT-2**, sourced from **Hugging Face**), was applied next in generating commentary from the data frame. The model was selected as it remains one the most diverse open-source large language models publicly available, bearing high precision, scalability, a vast embedded corpus, and multi-disciplinary applicability; with the subsequent output proving much clearer than the previous. In addition to GPT-2, applying the "stochastic gradient descent optimisation" technique, "**AdamW**" was explored for deep learning in computer vision and natural language processing. This approach was taken as it is an adaptive optimisation method that estimates first and second-order moments with added methods to decay weights. Because of this, it was hoped to aid in the training process, for improved outcomes.

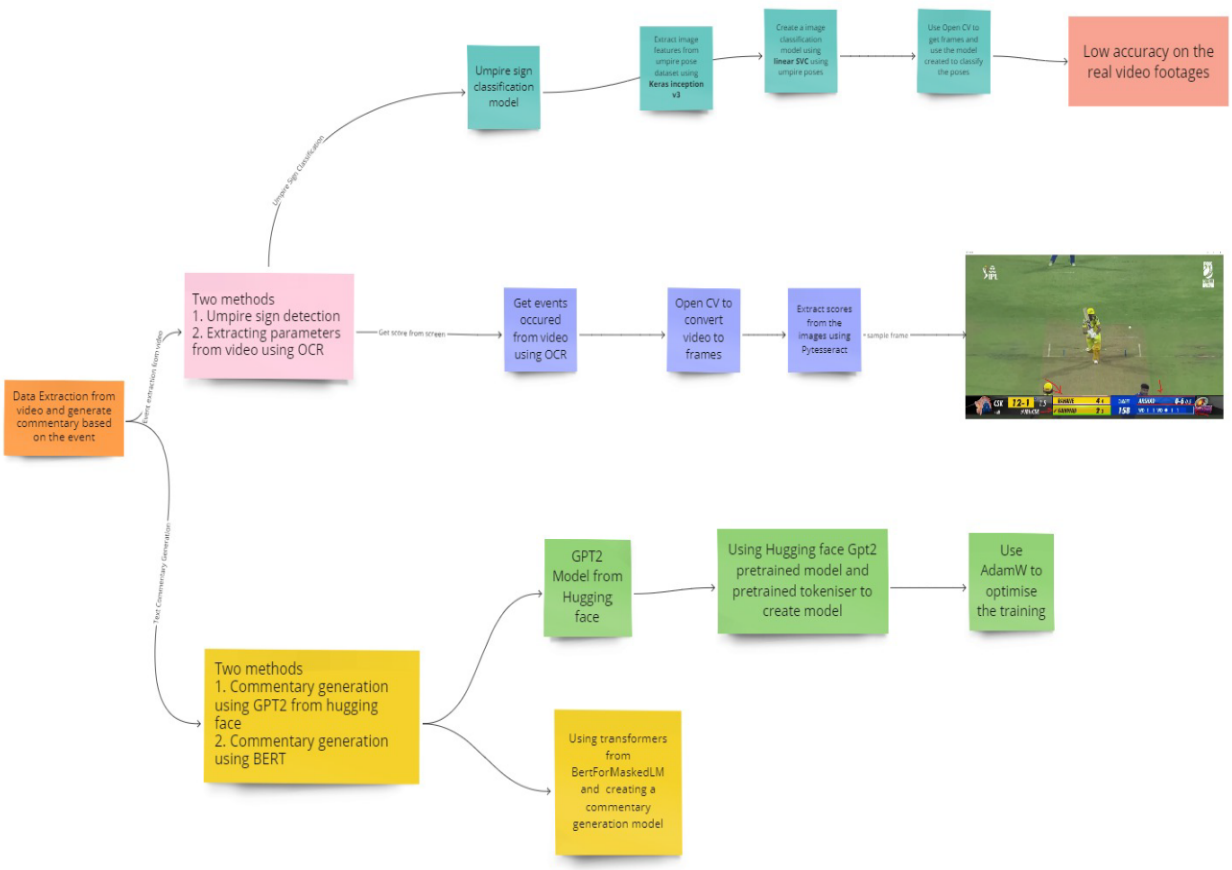


Figure 5 A "miro" flowchart, illustrating the thought process and journey behind successful and unsuccessful models.

[2.3] Trials Conducted

[2.3a] Commentary Generation Model

Transformer based architectures are typically applied for text generation (Vaswani et al., 2017). Experiments were carried out on each transformer-based technique applied in the project against the output as a model for commentary generation.

The three transformer-based experimentation techniques (Jay Alammar, 2019):

- BERT (Selected for Experimentation)
- GPT-2 (Selected for Experimentation)
- Transformer XL

The dataset for commentary generation is sourced from Kaggle, collected by: (Narendra Prasath, 2020).

The first trial instantiated a model using “pre-trained BERT” from transformers. The resulting text was evaluated and deemed insufficient and inaccurate.

Examples:

Prompt:

“Axar hits four”

Generated Text:

“axar hits four and four more.. and four more..... for four. four more..... and four more..... and.....”

Experimentation then proceeded with GPT-2 using the same dataset. The Model was trained with a “pre-trained GPT-2” model from hugging face and the base model was evaluated with basic prompts.

Prompt:

“Dhoni hits a six”

Generated Text:

“0: Dhoni hits a six-foot-3, 250-pounder in the back of the head. He has a short, thin body and a short, long nose. The first two games of the season, the Pacers have allowed the second-fewest points in the league. The Pacers have allowed the second-fewest points in the league. "We're not a team that's going to play a lot of basketball," said coach Frank Vogel, who has

1: Dhoni hits a six-footer from the left corner, and the Heat go on to win.
3. Miami Heat (5-6) Last week, the Heat were ranked as the No. 2 team in the league in scoring. This week, they're No. 2 in scoring. Miami is in the midst of a rebuilding process, and they've been able to get back on track. The Heat are a team that has had a lot of success.

2: Dhoni hits a six-foot-long ball with his right hand. The ball bounces off the rim and lands on the rim.

The ball lands on the rim and lands on the rim. The ball lands on the rim and lands on the rim. The ball lands on the rim and lands on the rim. The ball lands on the rim and lands on the rim. The ball lands on the rim and lands on the rim. The ball lands on the rim and lands on the”

Result:

Inaccurate commentary.

Model training was initially conducted with personal computers, requiring upwards of 10 hours to train (due to the large nature of the dataset). This task was delegated to Google Colab Plus, providing the project with a High RAM runtime and GPU for more advanced computation. After testing this approach with prompts, it yielded:

Prompt:

“Dhoni hits a six”

Generated Text:

“Dhoni hits a six over the bowler. Dhoni is the man running the show here. Ish Sodhi to Pant, SIX, the ball flies down, Pant is back in the crease, and he punches it through point, no chance for the fielder in the deep Shakib to Hardik Pandya, out Caught by Rahul!! And the rest. Shakib takes a step back and takes some more air. And it goes flying. Shakib has been off”.

Prompt:

“Out! Caught by de Kock!”

Generated Text:

“Out! Caught by de Kock! Just the sort of thing a fielder should do. Just a simple call for him. The full ball does the trick, just a straight ball over mid-wicket, de Kock was waiting on it and got a thick edge past the keeper. Southee c de Kock b Bhuvneshwar 25(27) [4s-2 6s-2] Prasidh to Parthiv Patel, SIX,”

The output was correctly described in the first few sentences, deeming the commentary accurate, though with limitations. The second sentence, however, is inaccurate as it describes a completely different and unrelated event with randomly inserted player names. To overcome this, the dataset was split into specific events and trained to generate commentary for the correct event. The results yielded promising potential, as indicated below:

Prompt:

"Dhoni hits a six"

Generated Text:

"Dhoni hits a six with a slower delivery Siraj to Rohit, SIX, the timing is impeccable. The ball is sailing over his leg. A long shot on the back foot Gopal Pandya to SIX, full toss from the front leg, and SIX! The bowler doesn't mind. Goes to the ground and lofts it over deep mid-wicket. Uthappa to Maxwell, SIX, a fine delivery. Maxwell can see"

Prompt:

"Nate hits a six"

Generated Text:

"Nate hits a six-run ball over mid-wicket and the fielder is pleased with it, so deep into the crease that he clinches it over long-on for a maximum Suryakumar Yadav to Chris Lynn, SIX, not the best pitch of the night. This is the one where he's a little more determined to come up with his ideas. Lynn has given the delivery to him all along, and the delivery's sailing over deep mid-wicket "

To overcome the insertion of random player names, the names of Batsmen and bowlers were replaced with {{BATSMAN}} and {{BOWLER}} to act as corresponding placeholders:

Prompt:

"Nate hits a six"

Generated Text:

"Nate hits a six that makes it six. He has done this before - he picks up a ball and clobbers it back into the stands, but there's no hurry {{BATSMAN}} to {{BOWLER}}, SIX, that was a lot of fun and it's been a good innings from {{BOWLER}} {{BATSMAN}} to {{BOWLER}}, SIX, no ball from {{BOWLER}}. That's good timing"

As an experimental approach, batsman name (GAIKWAD), bowler name (ARSHAD) and the corresponding score (indicated below) were inserted in place of the placeholders.

Figure 6 A still frame utilised in model training.

Final generated commentary:

Prompt:

"*Gaikwad hits a six"*

Generated Response :

"Gaikwad hits a six and it sails over the long-off boundary {{BOWLER}} TO {{BATSMAN}} , SIX, {{BATSMAN}} has picked himself up on the short ball and the length-wicket {{BOWLER}}, to {{BATSMAN}}, SIX, so much that that it's gone to waste {{BOWLER}}, to {{BATSMAN}} SIX, wow. SIX in the early going"

Final Commentary after replacing names:

" GAIKWAD hits a six and then picks it up. Full power and a big length delivery that goes in the stands. Another six over long-on ARSHAD to GAIKWAD, SIX, GAIKWAD! Goes inside out and does what he does best - swings over deep mid-wicket and pummels it over deep mid-wicket for a maximum ARSHAD to GAIKWAD SIX "

Prompt:

" Gaikwad hits a four "

Final Commentary:

" GAIKWAD hits a four, GAIKWAD stays on the off-side, gets a strong pull to the left of the diving keeper, misses the cut, it's a cracking shot. Not a lot of luck for the man in the deep
ARSHAD to GAIKWAD, FOUR, not all that easy. GAIKWAD leans forward and beats short fine leg for four, just past the rope "

This experimental approach proved that the models were effective. The finalised product was exported from Google Colab and placed in Google Drive for further testing and evaluation (see the link provided at the beginning of the report).

[2.3b] Video Event Detection

The first model created could identify umpire poses and gestures, and provide a corresponding commentary to the event that occurred through an SVM classification of deep features (A. Ravi, et.al., 2018). The dataset contains umpire poses that represented four key events: SIX, NO BALL, OUT and WIDE (A. Ravi, et.al., 2018). Images of umpires were acquired from varying sources, where features of the image were identified using keras inception v3 networks, and classified using the Linear SVC. Though the trials of the model performed well for specific events, there were many frames classified incorrectly.

Py-tesseract was instead utilised to extract the name of batsmen, bowler and scores from the specific on-screen score frames to generate commentary. Initially, the OCR didn't perform well due to image formatting issues (RGB). Thus OpenCV was applied to convert the these score frames to extract the player names accurately. Issues were still encountered in trials when extracting data for the score. This often was the result of OCR, occasionally representing the digit '1' as 'I', or 'WD' was wrongly identified as 'wO'. This however can be rectified with future tweaking.

[2.4] Ethical Implications

Privacy

Applying televised matches for commentary generation may raise concerns towards the privacy of both players and members of the audience. As the Umpire-verdict model (which relies on collected identification data - even if it's to distinguish umpires from players or the audience) and the Score-scraping model both utilise player names within the batsman feature of the commentary data frame, it is imperative that personal information about each individual is not collected or used without consent or falls in the boundaries of analysis and distribution.

Bias and Discrimination

A model's output has the capacity to perpetuate and propagate biases of input data. Should the data reflect any discriminatory views (for example, an existing commentator praising a specific batsman over another consistently), it may perpetuate this bias in the automated commentary it generates, subsequently not recognising the gravity of certain cricket events over others. It is important to ensure that the training data is diverse and covers multiple and impartial commentaries while remaining a representation of events that reflect the perspectives shared by the entire cricket community.

Ownership / Intellectual Property and Commercialisation

As broadcasted commentary is a form of live media and becomes the intellectual property of the owner, the distribution rights of commentator data used for training are to be respected and have approval prior to implementation. The use of commentary without permission or acknowledgment of its source has the capacity to lead to legal disagreements. Moreover, as the final automated commentary is data, should the project be expanded and applied internationally across other cricket leagues or sports, international distribution laws may vary and thus need to be addressed and absolved prior.

Inaccuracy

Due to the rule and uniform distinctions of cricket, a risk of inaccuracy in the automated commentary is always a factor. Incorrect predictions and insights could mislead and influence the viewing experience. As nuance is also captured and replicated in the descriptions provided by commentators, such as "Oooo", "Oh!" or "Wowow!", accurately applying these reactionary remarks where appropriate to correlate with the circumstance is also just as important for viewer immersion.

Job Displacement

The application of automated commentary may result in skill obsolescence, leading to disinterest in pursuing the commentary profession or job displacement for existing commentators. As this project intends to augment and not replace the existing commentary service in an effort to expand the industry, it is still important to consider the potential impact on employment in order to mitigate any negative effects. This includes only offering the model as a transitional instalment or as a means to communicate to physically or environmentally impaired viewers.

Section 3: Insights and Evaluation

[3.1] Pre-modelling Insights

Following cleaning and pre-processing through tokenisation, analysis of the commentary text was performed with **NLTK** libraries, providing visualisations to better interpret the data prior to application in models. The analysis yielded insights about the frequency of words, individuals, verdicts, and the overall expression of the commentaries. As the commentary model would be trained on this data, the frequencies of verdicts may play a role in influencing the model's output.

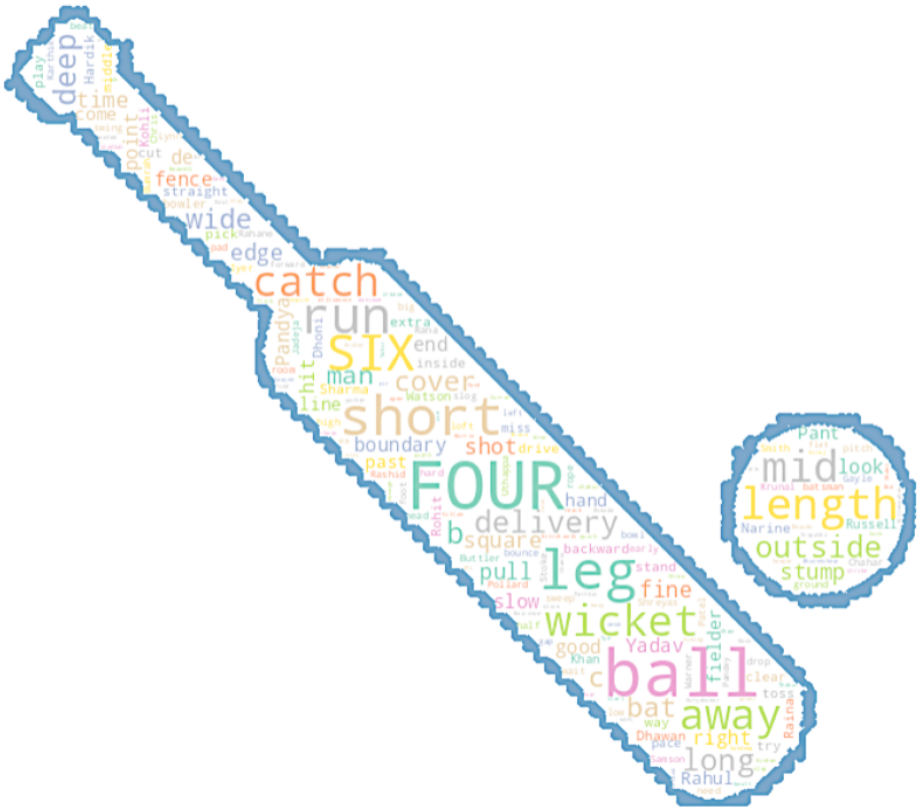


Figure 7 A WordCloud displaying the frequency of words in the dataset, by size.

By applying a simple **WordCloud**, the largest words indicate that '**FOUR**' would be among the more frequently mentioned words, followed by '**SIX**'. This chart helps distinguish the frequency of outcomes preferentially mentioned by commentators. 'Ball', 'leg', 'short', 'run', 'wicket', 'length', 'mid', 'catch', 'deep', 'cover', 'away', 'outside', 'fence', 'boundary' and 'long', are some words which are frequently used by the commentators and form a big part of their vocabulary.

By further engaging with sentiment analysis, it's observed to be mostly neutral terminology (~85%) with a higher occurrence rate of positivity (~10%) and over negativity (~5%). This indicated that commentators of this data frame typically refrain from making pessimistic and cynical statements, which could be interpreted as unengaging. Positivity retains excitement to retain viewership, thus sentiment analysis has indicated the finalised commentary behind the model generated must possess a slant towards a positive output. This makes sense, as the motive of a commentator is to promote an entertaining experience.

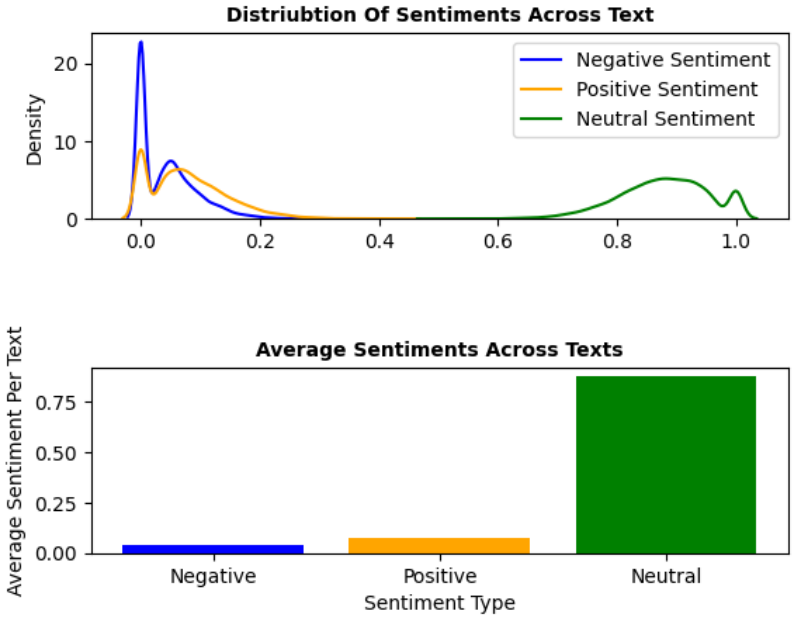


Figure 8 Graphical representations of performing sentiment analysis on the text data.

A final noteworthy visualisation came from tallying player names, revealing the most popular batsmen and fielders. These bar charts indicate two possibilities, the batsman and fielders have been on the pitch for longer than other players and thus are talked about for longer periods of time; or the players with the most mentions are more popularly discussed due to favourability.

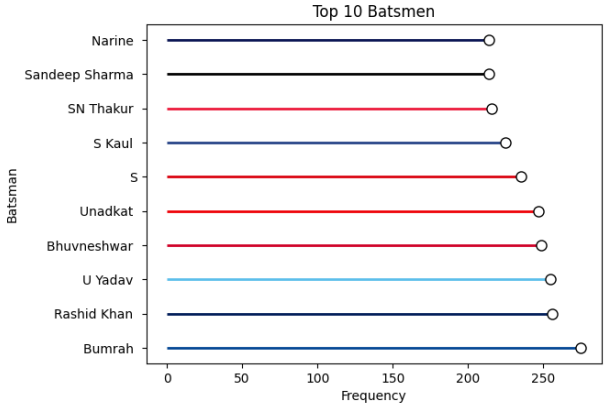


Figure 9 A "Lollipop chart" indicating the top 10 most frequently mentioned batsmen.

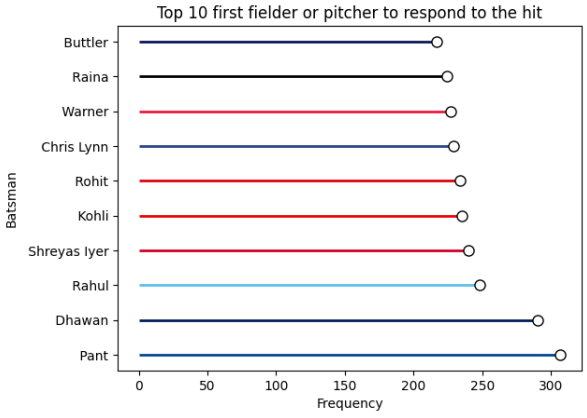


Figure 10 A "Lollipop chart" indicating the top 10 most frequently mentioned responder to a hit ball.

[3.2] Insights from the Data

1. IPL commentary more often than not, includes descriptions of positive sentiment in their remarks and exclamations, as indicated by the sentiment analysis and most frequently used interjections (see appendix).

2. Commentators frequently analyse the performance of specific players and provide insights into their strengths, weaknesses and performance both during and external to the cricket event, as indicated by the frequency of batsmen and fielders in their discussions. This discussion typically follows a significant event, such as hitting a multiple run (Four, or Six), or being caught out as indicated through feature engineering the score towards the end of the description. Commentary additionally includes references to the history of the game, including notable past matches, when describing these players.

3. Commentary tends to highlight key moments in the match, such as significant wickets or runs scored, as indicated by the length of the commentary feature, where this discussion is longer following an “Out”, “Six” or “Four” verdict. This is likely to update entertain the audience between interchanging players.

4. Commentators often engage in repeated use of reactionary words, figurative language and metaphorical expression, such as “Wow” and “Oh” to describe the actions and magnitude of events on field, as indicated by the frequently used interjections graph (see appendix).

5. Commentators tend to integrate light-hearted humour between pitches, indicated by the string length of the commentary feature and the higher positive sentiment analysis.

[3.2] Insights from the Model

Umpire-verdict Model

Insights: *unanticipated complexity.*

Umpire signals are specific to the rules of cricket, and understanding each gesture is vital to accurately determine a verdict on field. This is becomes especially important if someone is tasked with keeping track of umpire decisions internally, or perhaps is problematic as a hearing impaired member of the audience and can’t view a live score. These signals are held in either a stagnant position (no ball), or a continuous motion (similar to no ball, but repeatedly pointing to one side and back). Given the Umpire-verdict model relied on still frames to decide between these outcomes, it often confused the two decisions. Umpire verdicts can likewise be difficult to interpret when the camera is positioned at a different angle – leading to height irregularity between arms, or as the uniforms, heights, skin colour and additional clothing articles worn (bucket hats or gloves) shift between games.

GPT-2 Model:

Insights: *accurate if events are isolated.*

The **pre-trained** model yielded text that was considered fundamentally irrelevant. Following **training the model on the entire dataset** with all the events and evaluating the model, the resulting commentary generated became more appropriate for application but carried a mix of events instead of the prompted event. **Splitting the dataset** and training separate models to generate mutually exclusive commentary for appropriate events facilitated a strong performance.

The models performed exceptionally well for Model_Six, Model_Four and Model_Out, and the quality of commentary generated is appropriate, whereas the Models_One and Model_NoRun are generating commentary with a lower quality. The key insight here is because of the discrepancy in the corpus size the model is ultimately trained on. Thus, to improve the quality of a poor-performing model, more data would be required for training. While creating a separate model for each event solved the event mix-up problem, the model began to generate player names not in the prompt. This was remedied by replacing these names with placeholders. These placeholders are replaced by the player names extracted from the video using packages OpenCV and py-tesseract OCR; resulting in a commentary that is accurate and based on the player playing in the video.

BERT Model

Insights: *inaccurate.*

The pre-trained BERT model from hugging face transformers was applied, as well as training using the project's dataset. No insights were attained and the resulting commentary was of poor quality. No further application was experimented with BERT, though future projects would encourage tuning the model further.

Section 4: Outcomes

[4.1] Delivery Outcomes and Values Added

5.1 Project Delivery Outcomes

The project set out to recognise the gestures and actions of an umpire through a pre-trained model using Keras and then mapping these events to the automated commentary generation model. But the lack of similar real-world data in live stream/broadcast tainted its performance. While this limitation prevented the development of a fully functional model, it pointed the project towards changes that could be made in the sports broadcasting world. While the project did not meet this delivery outcome, uncovering this issue added value for the broadcasting industry – highlighting the need to provide continuous tracking of the umpire's actions. If applied, this model's method could set the stage for a highly successful method of automatic commentary generation.

Refining the approach to accommodate observed limitations by targeting the on-screen statistics with OCR proved to be a consistent and accurate approach to characterise the various events. Pytesseract's application performed better on the detailed score information (bottom right) rather than the main scorecard (bottom left). While this model achieved its intended outcome with satisfactory results, it was presumed the value added to the industry could be expanded further.

Two further methods were explored - the first, involving BERT, generated text not rich enough and failed to meet the delivery outcome in achieving a natural response. This encouraged further exploration with GPT-2, edging the project closer to a fully functional model. This came with the limitation that the comment generated would not sync with the corresponding event. For example, the event 'SIX' would occasionally respond with 'FOUR'. By segregating the data into groups according to their unique event and then training the model individually, a fully functional model capable of generating rich and engaging cricket commentary was achieved.

The outcomes of the project are promising and can be regarded as a proof of concept for a model that is capable of producing cricket commentary, thus assisting in the expansion of its popularity by being more encompassing and inclusive. The value this grants also acts as a basis for developing a system that may erode entry barriers for those with disabilities, as it can be expanded to further be applied for live textual/speech commentary applications. By applying translation functionality, may further put the sport on the stage internationally.

The key themes are an improved efficiency with the delivery of content, and increased accuracy (nullifies human error), an enhanced viewer experience (environmental, language or disability barriers), result constituency, scalability (multiple sporting leagues and disciplines), and cost reduction.

Section 5: Reflections

[5.1] Challenges Encountered and Solutions

Several challenges were encountered throughout the project:

- The lack of airtime dedicated to umpires in cricket matches made it difficult to collect sufficient data for building a model to recognise umpire gestures. This called for the need to revise our modelling approach.

- The inconsistency in uniforms worn by umpires across different cricket leagues and countries further complicated the task of identifying and classifying umpire gestures. The team adopted a different approach to deal with these challenges by using an on-screen scoreboard to recognise events and statistics. The benefit of this approach is that it enables the extraction of relevant information, such as scores, wickets, and player names, which can then be mapped to events to establish a connection to the relevant commentary.

- Snags in sourcing video clips from original broadcasters: Although access to the source wasn't a problem, downloading clips was not an option and the required data needed to be recorded. Where again difficulties were faced, as these streams were DRM protected, some further research revealed the solution of limiting hardware acceleration for both the browser and recording software.

- While annotating and cleaning data for training, it was recognised that player and team names required replacement with placeholders prior, otherwise, the automatic generation would fabricate its own. Difficulties arose as the player's name included an initial instead of a full first/last name, as that meant the replacement of these initials could alter the meaning of the sentence (as the algorithm replaced the initial letter everywhere in the text). Rows of comments were subsequently removed, cleaning the data. The data loss in this action was fractional, where 10250 rows would still be available.

- VScode and GitHub were set up and updated on all computers during the project, and there were cross-platform difficulties with updating libraries. This was particularly challenging for some group members, who had limited experience in collaborating on VScode and Github, where coding sessions had to be completed and submitted by another member on their behalf.

- The present inadequacy and incompleteness of prior coding methods proved problematic. As a result, the team had to conduct investigation and develop custom solutions, which were often time-consuming and substantial in resource requirements.

[5.2] Limitations and Future Steps

While the NLP-based automated commentary system for cricket matches has succeeded, further work is necessary to address the limiting factors. The accuracy of the umpire model is one of the system's limitations. Even though the model is adept at identifying and tracking the movement of players and the ball, it may occasionally miss certain actions, such as a no-ball or a wide delivery, or confuse certain actions, such as no-ball and four.

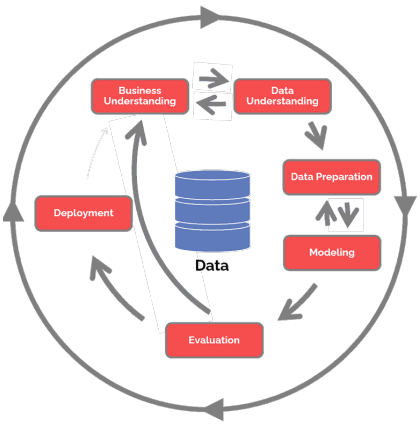
There is also a need for more research and models for cricket commentary. While various commentary datasets have been used, there may still be biases or gaps in the data that may affect the quality of the generated commentary (see the full list in the Ethical Considerations section).

Lastly, applying OCR experienced difficulty extracting information from scoreboards and other visual displays. When updating scores and statistics, this can negatively impact the accuracy of the generated commentary and the timely reporting of the outcome.

Future NLP-based automated commentary system versions could extend beyond the Indian Premier League (IPL) to include international cricket matches. This will require a more diverse and extensive dataset of cricket match commentary, including commentary in various languages and playing styles. Furthermore, it is possible to enhance the system's language capabilities to incorporate translation functionality for other languages. As a result, cricket fans worldwide can enjoy automated commentary in their native language, resulting in a more immersive and personalised viewing experience.

Future work can also explore the automation of commentary sports of other disciplines, where similar NLP techniques can be applied to expand industries outside of cricket. This objective will require the collection of extensive and diverse datasets of commentary for each sport and language and the development of specific models for each associated sport.

Section 6: Synopsis



[6.1] Cross-Industry Standard Process for Data Mining (CRISP-DM)

This report summates the developments and milestones achieved over the course of three weeks by four individuals. It follows a CRISP-DM approach, manipulating raw data to result in a Python script that delivers an output commentary for televised cricket matches.

The planning phase began with the **examination of existing datasets** of previous cricket commentary and determining the project's overall **feasibility** prior to data transformation and model production. This stage included concatenating all text files into a **corpus** and the **application of NLP techniques** (such as tokenisation or sentiment analysis) to analyse data as fragments as opposed to one enormous string. While the planning phase occupied a considerable amount of time, it provided a strong foundation for the project.

The model development phase began by implementing **two models**, one that would **distinguish umpires** from their environment and one that would **interpret** their signalling and gestures with **openCV** and **keras**. Despite the prior planning, we experienced a setback upon realising the finalised umpire-verdict model encountered many flaws, requiring considerable specificity and training that would exceed the provided timeframe. As the uniform colours shift between matches – sometimes resembling the umpire, as the umpire signals threw were occasionally exaggerated or underrepresented - where camera angles would impact the verdict, as the model would experience difficulty reading a moving gesture; it was determined here that the best approach was to revert back into the preparation phase and tackle the problem from another angle.

By asking ourselves what would provide a more consistent avenue for a score prediction, our direction became clear. A **new model** was developed, one that **scrapes scores** and **names** on screen as televised, proving to be a consistent and accurate predictive source. This model employs **openCV** to convert the original video feed and **pytesseract** to perform optical character recognition. This data is then directed into a language model (using **GPT-2** and **hugging face**), then **BERT** to capture and replicate the commentary nuance as seen in the original data frame. With a now reliable model, we were able to move towards the evaluation and application of translation functionality in the commentary.

Throughout the entire assembly process, coding segments were iteratively reviewed by the team for refinement, refining code portions where possible. The team met at least twice weekly to establish goals and gauge progress, offering help where needed. While the model is not yet considered deployable, the finalised product and the umpire-verdict model should both serve as valuable proofs of concept that may be applied to expand an industry.

Milestones

1. **Dataset examination**: Examining existing datasets of previous cricket commentary to determine the feasibility of transforming the data for our project.

2. **Data preparation**: Concatenating all text files into a corpus and applying NLP techniques to analyse the data.

3. **Model construction**: Developing multiple models for umpire signals and televised scores.

4. **Umpire-model setback**: Encountering a setback with the umpire-model, which required too much specificity and training.

5. **Review phase**: Reviewing and refining the product, amending and simplifying code portions where possible. Lastly, reflecting on the process to learn from mistakes, as to avoid in future projects.

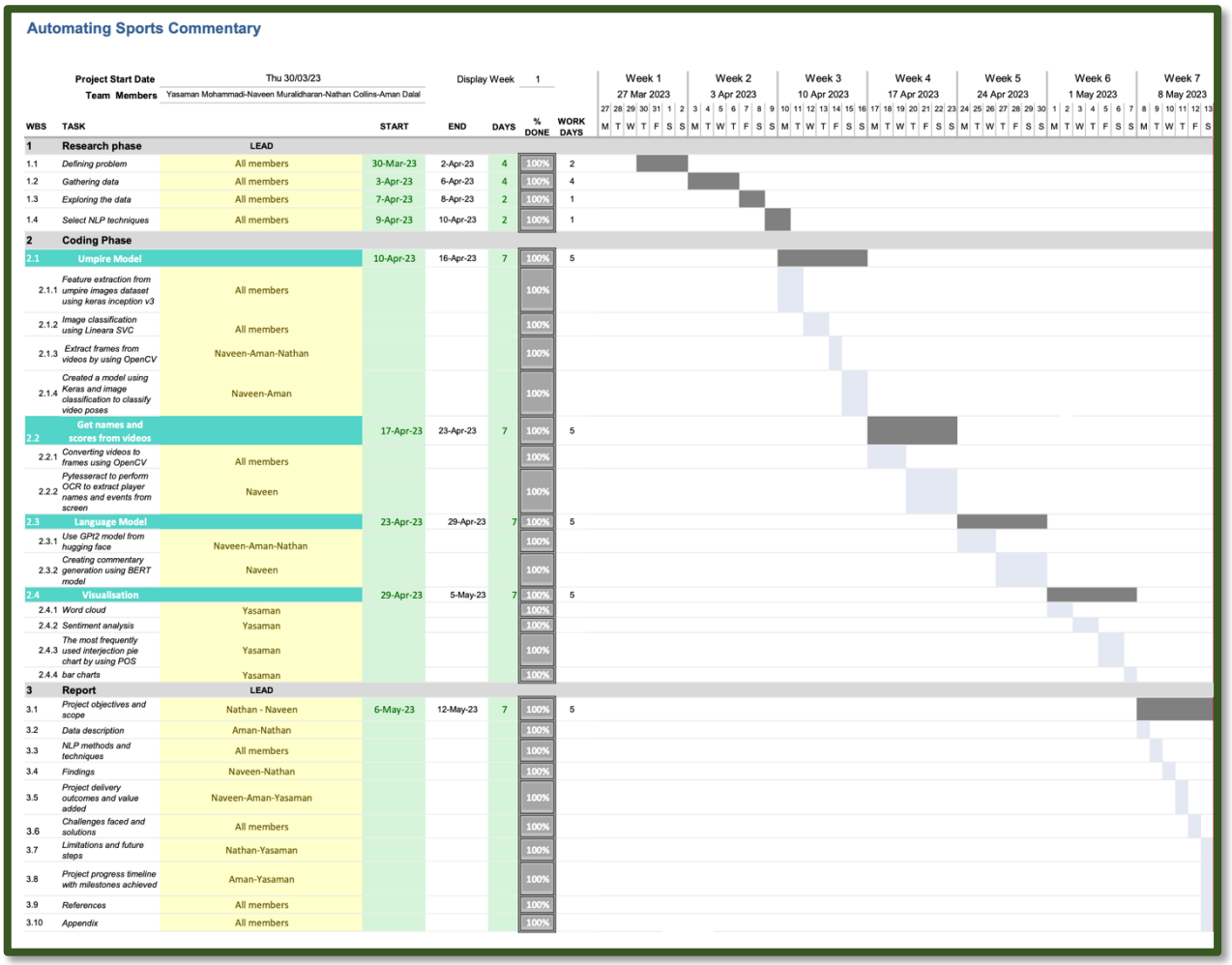


Figure 11 Gantt chart utilised to track group progress and timely met goals.

References

Business Understanding & Data Preparatory Phase

1. Bao, L., Li, J., Xing, Z., Wang, X., & Zhou, B. (2015, March). Reverse engineering time-series interaction data from screen-captured videos. In *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)* (pp. 399–408). IEEE.

2. Innodata Inc. (2021, March 18). Ethical Issues in Computer Vision and Strategies for Success. Innodata. Retrieved from https://innodata.com/ethical-issues-in-computer-vision-and-strategies-for-success/

3. Kumano, T., Ichiki, M., Kurihara, K., Kaneko, H., Komori, T., Shimizu, T., ... & Takagi, T. (2019, June). Generation of automated sports commentary from live sports data. In 2019 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB) (pp. 1-4). IEEE.

4. Li, J., Tang, T., Zhao, W. X., & Wen, J. R. (2021). Pretrained language models for text generation: A survey. arXiv preprint arXiv:2105.10311.

5. Mamoru, D. S., Panditha, A. D., Perera, W. J., & Ganegoda, G. U. (2022, December). Automated commentary generation based on FPS gameplay analysis. In *2022 7th International Conference on Information Technology Research (ICITR)* (pp. 1–5). IEEE.

6. NLPScholar. (2019, May 1). Ethics Sheet: AER. Medium. Retrieved from https://medium.com/@nlpscholar/ethics-sheet-aer-b8d671286682

7. Pavan Sanagapati. (n.d.). Knowledge Graph & NLP Tutorial: BERT, spaCy & NLTK. Kaggle. Retrieved from https://www.kaggle.com/code/pavansanagapati/knowledge-graph-nlp-tutorial-bert-spacy-nltk

8. Song, A., & Chen, K. (2013, March). OpenCV Detection of Athletes in Long Jumping Videos. In *Proceedings of the 2013 International Conference on Information, Business and Education Technology (ICIBET 2013)* (pp. 10–13). Atlantis Press.

9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems (pp. 5998-6008).

10. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners.

11. https://jalammar.github.io/illustrated-gpt2/ (2019)
12. https://www.kaggle.com/datasets/narendrageek/can-generate-automatic-commentary-for-ipl-cricket (2020)

13. Ravi, Aravind, Harshwin Venugopal, Sruthy Paul, and Hamid R. Tizhoosh.

14. "A Dataset and Preliminary Results for Umpire Pose Detection Using SVM Classification of Deep Features." In 2018 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1396-1402. IEEE, 2018.

Appendix

Text Data Features and Descriptions:

ID:
Represents the event entry identification for each unique input into the data frame.

Match_id:
Denotes the corresponding 7-digit match identification number that the event and associated commentary describe.

Team:
A 2 to 3-letter acronym representing the name of the team that the event and associated commentary describe. For example, **SRH** refers to the Sunrisers Hyderabad, or **KKR**, the Kolkata Knight Raiders.

Over_num:
Signifies a bowler's deliveries from his side of the pitch to the batsman. A group of six deliveries is referred to as an “Over”. For the feature Over_num, each set of six deliveries will be categorised by the occurrence of the event. For example, if the event resides in the first over, it is marked as “1st”.

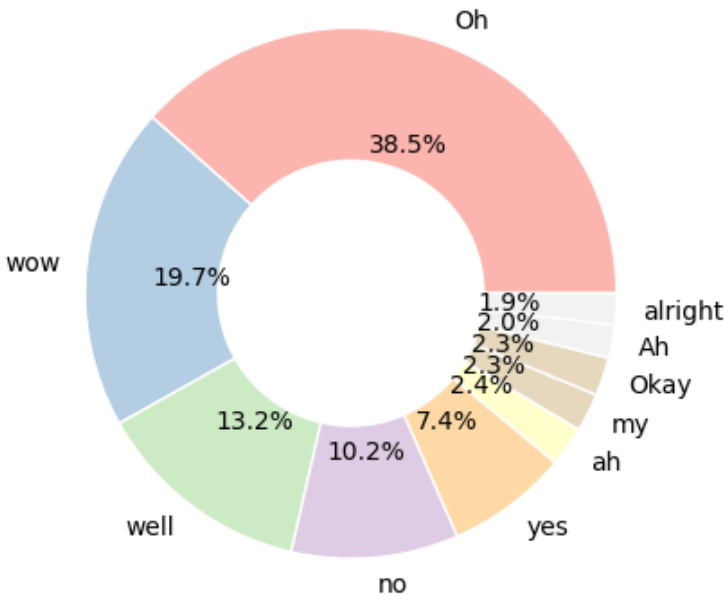
Commentary:
The primary feature of interest. This column will host the application of the majority of NLP techniques for analysis for model production to emulate. This feature lists the batsman feature, followed by the score and a short description of pitch and player descriptions intertwined with emotive expression and critique. For example: "*Chahar to Prithvi Shaw, FOUR, smashed! This is the length that Shaw enjoys a lot. Shorter from Chahar, and it's been dispatched with a nonchalant pull through square leg*".

batsman:
Two names separated by “to”. The first name is described in full and represents the batsman, the second is the first fielder or pitcher to respond to the hit, denoted with the individual’s last name.

score:
The final verdict of each event. This is categorised by first, the quantity of runs achieved by the batsman, such as “**no runs**” or “**2 runs**”, or the final position of the ball, such as “**OUT**”, “**FOUR** or “**SIX**”.

	Label	POS
0	PROPN	[Nehra, Mandeep, FOUR, Mandeep, RCB, Mandeep, ...
1	ADP	[to, for, on, up, over, of, into, to, to, in, ...
2	PUNCT	[,, ,, ., ,, ,, ,, ,, ,, -, -, ., ,, ,, ,, ,, ...
3	ADJ	[first, Full, first, short, third, late, more,...
4	NOUN	[boundary, pads, mid, -, wicket, couple, bounc...
5	CCONJ	[and, and, and, and, and, and, and, and, and, ...
6	DET	[the, the, the, a, the, the, the, the, another...
7	VERB	[needed, put, did, picked, dispatched, end, ha...
8	PART	[to, to, to, n't, to, to, to, Not, to, to, to,...
9	AUX	[be, is, did, was, 's, 's, 's, is, is, has, 's...
10	ADV	[away, just, back, over, Again, hard, just, pr...
11	PRON	[that, it, it, his, he, it, it, he, his, which...
12	INTJ	[alright, Yeah, oh, oh, Oh, Nope, Oops, oh, hu...
13	NUM	[four, 1, 31(16, 4s-4, 6s-1, 4s-3, 1(3, 1, one...
14	SCONJ	[that, as, as, than, for, Despite, before, as,...
15	X	[[, [,], [,], [, [,], [,],], [, [,], c, ...
16	SYM	[/, /, -, -, -, /, /, $, /, $, #, /, -, -, -, ...
17	SPACE	[\n]

Part of speech detection (POS)



Pie chart of most frequently used interjections.