STORY SELECTION AND RECOMMENDATION SYSTEM FOR COLOUR COMMENTARY IN CRICKET

Dr. Chitrakala S[1], Akshaya K[2], Nisha S[3], Srravya C[4]

[1,2,3,4]*Computer Science and Engineering, College of Engineering, Guindy, (India)*

ABSTRACT

During a cricket match, commentary keeps the viewers entertained and updated about the game. Quoting relevant stories related to the current game scenario makes the game more interesting. But the knowledge of commentator, however vast for a human being, is still limited relative to the total set of available stories. A major challenge is to narrate the most relevant stories (past incidents) based on the current (never-before-seen) game state. The paper proposes a solution which is an AI based approach that will assist the colour commentators in effective storytelling that is interesting to the audience, and related to what is actually happening in the game being broadcast.

***Keywords**: Artificial Intelligence, Colour Commentary, Cricket, Information Retrieval, Story Telling.*

I INTRODUCTION

Cricket commentary has been a crucially important conduit for linking fans with games, and inculcating them into its romance and mythology. Cricket would not be the same without its commentators. They bring fresh and colourful insights into the lives of cricket fans. The *main commentator*, also called the *ball-by-ball* commentator is the primary speaker on the broadcast. They focus on describing each play or event of an often fast-moving cricketing event. The *analyst or colour commentator* provides expert analysis and background information, such as statistics, strategy on the teams and athletes, and occasionally anecdotes or light humor. They are usually former athletes or coaches in their respective sports. The term "colour" refers to levity and insight provided by analyst. The most common format for a sports broadcast is to have an analyst/colour commentator work alongside the main/play-by-play announcer.

One of the great things about broadcasting cricket is that there is time to reflect, to dwell on other things apart from the game, because there is always the gap between action and non-action, especially when the fast bowlers are operating. It is a wonderful landscape for anyone who wants to enjoy language and technique, giving immense scope for colour commentary. It is about telling stories. People remember an anecdote, a moment that is unusual in the game. There are a lot of women who follow the game, some do and some don't have the knowledge of it, but listen because they are hearing a story, and might be even a few jokes at the same time.

To narrate such stories, colour commentators need to be experts in the game, having followed it closely for many years. Hence, more often than not, they are former cricketers. Even such players reportedly prepare by reading and absorbing, interviews with the people in the game, the players occasionally, in the lead-up to the match. The veteran

Australian commentator Jim Maxwell explained that he refreshes his memory by looking at a story in the newspaper or ESPNcricinfo or wherever to keep backgrounding himself, especially when there are players playing a Test or an ODI who he has never seen before. Essentially, colour commentators look to have that kind of information at the back of their mind.

Earlier there used to be a lot more informality in the relationship between the players and the media than there is today where their lives are organised, even their dreams, and they have to report back on their wellness. It is hard to get close to the players the way commentators used to in the earlier years, where they might just meet up for a chat or even go out for dinner.

Therefore, in order to assist the colour commetator in bringing out relevant stories during an on-going game without involving tedious preparation on his part, the paper aims to test whether an AI approach can be developed that maps game states to relevant stories, thereby significantly increasing audiences' enjoyment of the broadcast. To test this hypothesis, an AI system is developed that tells stories in the context of cricket. To do so, the proposed system learns offline to connect sports stories to game states, provided with some scored examples of story-game state pairs. This learned mapping is then used during cricket games to suggest relevant stories to a (human) broadcast team or, in the absence of a broadcast team, to autonomously output a relevant story to the audience.

The rest of the paper is organized as follows. Section 2 consists of a summary of related work. Section 3 formulates the problem of mapping sports game states to interesting stories and describes the proposed approach – a combination of information retrieval techniques designed to rank stories based on a given game state, and then ensure the higher-ranked stories are indeed relevant to said game state. Section 4 describes the experimental analysis performed to choose a story ranker, and then use this ranker to select stories for cricket broadcasts. Section 5 evaluates the quality of the story mapping with user studies and demonstrations to professional sports commentators. The paper is concluded with a discussion of lessons learned, future research and applications.

II RELATED WORK

StatSheet [10] and *Narrative Science* [9] are automated systems which write previews for sports games that have not yet happened and summaries about sports games that have already happened. For summaries, they are provided with statistics from a completed game and compose a narrative about the game, with the goal being to provide an interesting summary of game events. For previews, they are provided with statistics from past games, and compose a preview for the game that should entice the reader to watch said game. Neither Statsheet nor Narrative Science operates with live game data and both are unable to solve the problem of providing live stories during a game. They may, however, provide an additional potential database of stories about past games to augment the current system.

Many modern commercial sports video games [6][7] use recorded commentary from professional commentators, in order to emulate the sounds of actual sports broadcasts. The comments are mainly generic, using pronouns and general terms, so that the commentary can be re-used for many games. Storytelling in these games exists, but is limited and repeated. Repetition takes away from the entertainment value of the video game, as it is unrealistic and boring.

Robot World Cup Soccer (RoboCup) [3][4] is a research testbed involving robots playing soccer. There is also a RoboCup simulation league, where the games are not physically played, but are simulated on a computer. Both the physical and simulation leagues provide researchers with a standard testbed in which to evaluate their AI strategies for various goals. In this context, previous work in automated commentary has focused primarily on automated play-by-play commentary. The system obtains the data from the gameplay's main features—the player locations and orientations, the ball location, and the score. Each system generates natural language templates, filling in player and team names where appropriate, then uses text-to-speech software to verbalize the derived commentary. *Dynamic Engaging Intelligent Reporter Agent (DEIRA)* is a similar system, as it performs the same task, but in the sport of horse racing.

Within its live online game summaries, Major League Baseball uses a system called *SCOUT* [5] (Kory 2012) that provides textual analysis of the current game. The viewer is shown information such as the type of pitches thrown during an at-bat and the tendencies of the batter with respect to the pitcher. While SCOUT provides some colour, it is mostly a statistical summary and currently does not tell stories.

There are some attempts within these systems to provide colour commentary, but none go as far as to try to incorporate storytelling. That is, these systems tackle a problem that is different from what this project is aiming to solve—they automate factual commentary with some bias added, but do not implement colour commentary via stories. This system could be used in conjunction with each of these automated play-by-play systems to create fully automated commentary, featuring both play-by-play and colour.

III APPROACH

Fig. 1: Architecture diagram of the proposed system

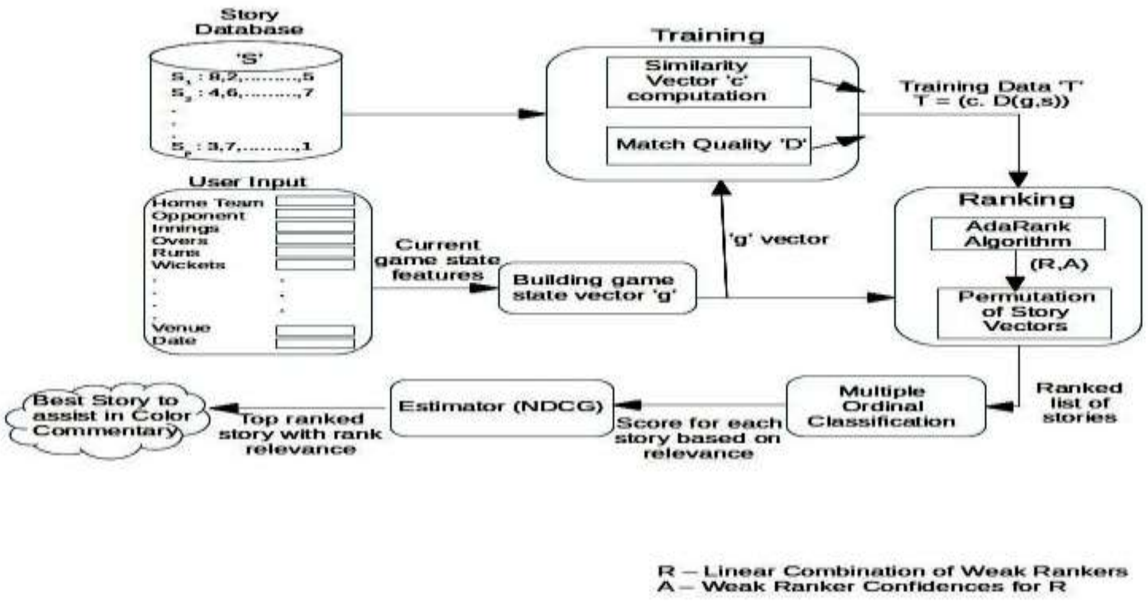
Story Database
'S'
S1 : 8,2,.........,5
S2 : 4,6,.........,7
Sr : 3,7,.........,1
Training
Similarity Vector 'c' computation
Match Quality 'D'
Training Data 'T'
T = (c. D(g,s))
User Input
Home Team
Opponent
Innings
Overs
Runs
Wickets
Venue
Date
Current game state features
Building game state vector 'g'
'g' vector
Ranking
AdaRank Algorithm
(R,A)
Permutation of Story Vectors
Ranked list of stories
Multiple Ordinal Classification
Score for each story based on relevance
Estimator (NDCG)
Top ranked story with rank relevance
Best Story to assist in Color Commentary
R – Linear Combination of Weak Rankers
A – Weak Ranker Confidences for R

Figure 1. Architecture Diagram

The system's goal is to retrieve and recommend stories that will aid the color commentator to narrate relevant interesting incidents during the course of the game, as the color commentator cannot remember as much stories as the system does. Given the game state, the system returns ranked list of stories, based on how appropriate they are for the game state. Steps involved are building the game state vector from the input, training the dataset, ranking the features to sort the stories, computation of match quality for every story vector and evaluation of the result based on the match quality.

3.1 Training

The raw input got from the user cannot be given as input to the main ranking algorithm directly. This user input should be changed to some form that is similar to the story vectors in the database so that they can be compared during the computation stages. So this raw input is processed to obtain a vector of game features. This involves addition of extra features from the existing database to construct a complete game vector 'g'.The training of data is done through Information Retrieval techniques (IR) and machine learning techniques. The training set T (c,D) is a combination of the similarity vector 'c' and the match quality D.

To build this training data set, a set of 'n' game state vectors G=(g1,g2...gn) is taken to form a set of n*q similarity vectors 'c' for all gi є G and all q story vectors from our story vector library S=(s1,s2....sp).

3.1.1 Similarity Vector Computation

Instead of giving the input game state and story features separately to the IR algorithms, a connection is made between the two and fed as input to the IR algorithms which makes it easier for further computation. This connection between the game state and the story features is made by computing a similarity vector for a game state specified by feature vector 'g' and a story specified by feature vector 's'. Every component of the similarity vector 'c' is computed by comparing one or more features of 'g' to one or more relevant features of 's'.

3.1.2 Match Quality Computation

The similarity vector 'c' denotes the degree of relevance of the game state vector 'g' and story vector 's' , but fails to give a scalar value for the same. The match quality is an integer scalar value on the 5-point scale from 0 for a completely irrelevant match to 4 for an absolute match. Thus, the problem is given a game state g, to retrieve a story s of the highest match quality D(g,s).

3.2 Ranking

AdaRank, a listwise algorithm has been adapted in this paper. *AdaRank* forms a "strong" ranker by iteratively selecting and combining "weak" rankers. A weak ranker uses a single component of the similarity vector to rank (i.e., sort) the training data. Each weak ranker has a "vote" on the final ranking, based on how its ordering of over the training data was scored according to a chosen IR scoring metric. In *AdaRank* algorithm, a set of training data *T*, the number of game states *m* in *T*, an IR scoring function *F*, the number of weak rankers to compose the strong ranker *k*, and the number of tie-breaking features to use *y*. *AdaRank* first partitions *T* by its *m* constituent games states. This is done because stories can be meaningfully sorted by the match quality (D) values only for a given game state. The ground-truth rankings are then calculated for possible use in evaluating weak rankers. All weights are initialized to 1/*m* as all game states are initially equally important. The main ranker *R* and its corresponding confidence values *A* are initialized to be empty sets. The set of feature combinations to be considered for use in weak rankers is calculated based on the number of features in each in similarity vector, and the number of features to use for tie-breaking *y*. At each iteration of *AdaRank*, elements of whose first elements have not yet been used in are considered as possible weak rankers. Thus, each feature of similarity vector may only be used once as the main sorter in a weak ranker. For each game state, the weighted score *v* of sorting T_i by *b* (with any remaining ties broken randomly) is calculated, using the scoring function F and current weights *w*. The arguments to F vary based on which scoring metric is used, but all metrics we consider accept θ_i, the match qualities for T_i as input. If the mean weighted score *v* for *b* is greater than the maximum encountered so far in this iteration, the feature combination to be used in the weak ranker r for this iteration is set to *b*. After evaluating all valid elements of B, the best weak ranker for this iteration is added to the main ranker R. Following this, *A* and *w* are updated. The data is reweighted after each iteration so that game states for which stories have been poorly ordered are given more weight.

Algorithm 1: AdaRank Algorithm for sorting story vectors based on the features of the similarity vector

Input:

```
        T: training data
        m: number of game states
        M: IR scoring function
        k: number of iterations
        y: number of tie-breaking features
```

Output:

```
        R: ranker
```

AdaRank (T, m, M, k, y)

```
  Partition T by game state: $T = T_1 U T_2 U... U T_m$
  Sort each $T_i$ by D values, yielding ground truths Ti
  Initialize weights w(G) to 1/m
  Initialize ranker R and weak ranker confidences A to $\varphi$
  Get all combinations B of length y + 1 from T
  for each iteration up to k
    $\mu \leftarrow 0$
    for each combination b in B
      if $b(1) \in R$
        for i = 1, . . . , m
            sort $T_i$ by b, yielding $T_i$
            $\theta_i \leftarrow D(T_i)$
            $v(i) \leftarrow w(i) \cdot M(\theta i, . . .)$
            end for
            if mean(v) $\geq \mu$
                $\mu$=mean(v)
                r←b
            end if
        end if
    end for
    add r to R
    calculate $\alpha$ for r
    add $\alpha$ to A
    update w
end for
```

In order to create the training set we need two parameters. The first parameter is the similarity vector ’c’ which is calculated by comparing vectors’ g’ and ’s’. For example, consider the following:

Vector ’g’ = g1, g2,....., g34,...gn

Vector ’s’ = s1, s2,....., s34,...sp

The g_{34} feature of the game state vector ’g’ is compared to the corresponding s_{34} feature of the story vector ’s’ to produce a value for the match feature ’c34’ in similarity vector ’c’. If both g_{34} = 1 and s_{34} = 1, then c34 = 1. Otherwise c34 = 0. This is in the case of the binary features. For non-binary features, we use feature specific functions. Vector ’c’ = c1, c2,....., c34,...cz

The second parameter of the training set is D which is the match quality. The similarity vector ’c’ is mapped to D(g, s) — the 5-point-scale quality of the match between g and s. A sample training set ’T’ is shown below.

T = (c, D(g,s)) class.

Table I: Training Data T for AdaRank

Story	c1	c2	c3	c4	D
1	0	0	0.875	0.6	4
2	0	1	0.5	0.4	2
3	1	1	0	0	1
4	0	0	0.375	0.9	3

The trained data set T is given as input to the ranker module. Consider the following scenario. The AdaRank algorithm produces R = (c1, c3) with a confidence measure of A = 0.63. This means that if the stories are ranked based on c1 with c3 as tie breaker, then the system produces a ranked list with 63 percent accuracy.

The result of ranking by c1 produces is a vector of ranked stories indicated by pi. For every element in pi, the corresponded match quality is computed. In the final stage of evaluation, the top ’n’ stories are evaluated based on their match quality using NDCG algorithm to produce a single value which is a measure of how relevant the stories are to the input vector.

3.3 Multiple Ordinal Classification

According to DCG, the ranking problem can be cast naturally as a multiple classification (i.e., K = 5 classes), because obviously perfect classifications will lead to perfect DCG scores. While the DCG criterion is non-convex and non-smooth, classification is very well-studied. For example, suppose within a query, the stories are all labeled level 1 or higher.

For every game state that is given as input, the classifier assigns a relevance level y ranging from 0 to 4, with 0 as a poor match, and 4 as a perfect match. If an algorithm always classifies the stories one level lower (i.e., stories labeled level 4 are classified as level 3, and so on), we still have the perfect DCG score but the classification “error” is 100%. This phenomenon to an extent , may provide some “safety cushion” for casting ranking as classification. It appears to us that the regression-based approach is less direct and possibly also less accurate than our classification-based proposal.

For example, it is well-known that, although one can use regression for classification, it is often better to use logistic regression especially for multiple classification.

For multiple classification, we consider the following common surrogate loss function:

$$\sum_{i=1}^{N}\sum_{k=0}^{K-1} -\log(p_{i,k})1_{y_i=k}. \longrightarrow \boxed{\text{Eqn 1}}$$

3.4 Estimator

The estimator is used as a threshold to ensure the top-ranked story which is got as the output of the AdaRank algorithm is worth telling. This estimator accepts the match quality computed in the earlier step by classification and evaluates the top or top 'n' stories as to how good they are. Methods for estimation:

3.4.1 Winner takes all

This metric takes into account the match quality only for the top ranked story, comparing it to the current game state. If the top ranked story’s match quality is equal to or exceeds the threshold, WTA returns 1, otherwise, it returns 0. Thus, WTA is a binary threshold scoring metric whose output rests completely on the top ranked story, ignoring the rest of the ranking.

3.4.2 Average Precision

When we consider scoring metrics other than WTA, they consider more stories than the top ranked story. AP takes the same input as that of WTA, but it also needs the number of positions to consider relevant, N . That is, AP requires the user to specify how much of the ranked list is important.

At each position of the match quality to be considered, AP checks if it meets the threshold t. If so, this position is considered relevant. The precision P at any position i is the total number of relevant stories in the top i positions, divided by i . The average precision, γ, is then calculated, based on the precision at each relevant position and the total number of relevant positions.

3.4.3 Normalised Discounted Cumulative Gain (NDCG)

Here, gain at position i is calculated using the match quality at this position ($2\theta i - 1$). The cumulative gain is the sum of these gains over all positions N to be considered. The discount factor is $1/\log(i + 1)$, increasing the discount as the rank position worsens. Finally, the metric is normalized by dividing by the optimal Discounted Cumulative Gain – that of the ground truth scores $\theta*$ for the given g.

NDCG takes as input the vector of match qualities θ for a ranking π, the number N of positions to consider, and the ground truth match qualities $\pi *$ for the given game g. First, the true DCG, E, and the optimal DCG, V , are initialized to 0 . Then for every position i up to N , E and V are updated by adding the Discounted Gain at i .

Finally, the NDCG is calculated by dividing the true DCG by the optimal DCG.

IV EXPERIMENTAL ANALYSIS

We conducted several experiments to evaluate whether SCoReS improved broadcast quality in any of these applications. User Study I was conducted to ensure commentary was beneficial within our game library before we even tried to improve said commentary.

4.1 User Study I

Table I: User study questions asked to participants

User Study Questionnaire
1) I found this viewing experience enjoyable.
2) I learned something watching this clip.
3) I found this viewing experience interesting.
4) I found the video clip easy to follow.
5) I enjoyed the commentary in this clip.
6) Viewing this clip made me more interested in watching cricket.
7) Viewing this clip made me more interested in watching sports.

In this user study, *System Enabled Commentary* is compared with two different types of commentary. For *No Commentary*, the commentary was removed from the broadcast, and left with the crowd noise. The *Original Commentary* had voiceovers for the original professional commentary, with no stories inserted or present in the original commentary. The *ScoReS Commentary* had a story selected by our system.. Participants were asked to rate each question from 1 to 7 (strongly disagree - strongly agree). To measure the performance of the different types of commentary, the participants' answers to the eight questions listed in Table I were evaluated.

4.2 Evaluation

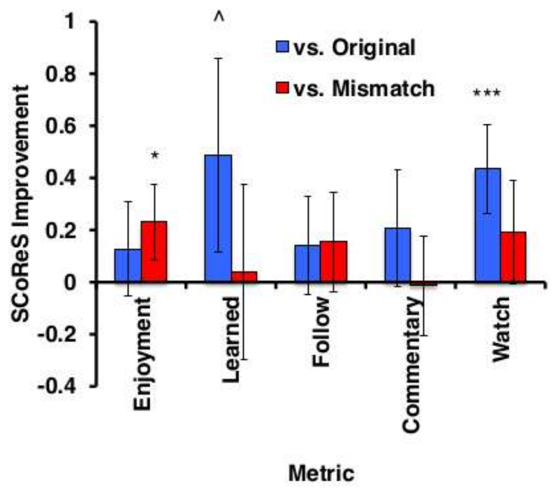

Fig. 2: Comparison of *System Enabled* Commentary against *Original* Commentary as well as *Mismatch* Commentary

The above graph shows the mean difference between *System Enabled Commentary* and both *Original Commentary* and *Mismatch Commentary*. *System Enabled Commentary* was ranked higher than *Original Commentary* for the "Viewing this clip made me more interested in watching cricket" metric. This shows that adding stories to commentary can improve a broadcast. *System Enabled Commentary* was ranked higher than *Mismatch Commentary* for the "I found this viewing experience enjoyable" metric. This shows that intelligently adding stories to commentary can be more enjoyable to the viewer than adding random stories.

V CONCLUSION

In this paper, it has been shown that the proposed system has a statistically significant positive influence on the sports viewing experience across several metrics. It was able to achieve significant improvement in overall enjoyment and increasing interest in watching cricket. To a realistic deployment, it would further improve the entertainment value of sports broadcasts. It also offers many possible future applications along the lines of fully automated commentary.

REFERENCES

[1] Greg Lee, Vadim Bulitko, and Elliot A. Ludvig, "Automated Story Selection for Color Commentary in Sports", IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES,

VOL. 6, NO. 2, JUNE 2014

[2] Greg Lee, Elliot Ludvig and Vadim Bulitko, "Sports Commentary Recommendation System (ScoReS): Machine Learning for Automated Narrative", The Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment

[3] E. Andre, K. Binsted, K. Tanaka-Ishii, S. Luke, G. Herzog, and T. Rist, "Three RoboCup simulation league commentator systems," AI Mag., vol. 76, pp. 57–66, 2000.

[4] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, and H. Matubara, "Robocup, a challenge problem for AI," AI Mag., vol. 18, no. 1, pp. 73–85, 1997.

[5] M. Kory, "Out of left field: Scouting the scout," Baseball Prospectus, May 7, 2012 [Online].

[6] Playstation, "MLB 09: The Show," 2009 [Online].Available:http://us.playstation.com/games-and-media/games/mlb-09-the-show-ps3.html

[7] 2K Sports: MLB 2K7 2007 [Online]. Available: http://2ksports.com/games/mlb2k7

[8] G. Lee, "Automated Story-Based Commentary for Sports," Ph.D. dissertation, Dept. Comput. Sci., Univ. Alberta, Edmonton, AB, Canada, 2012.

[9] Narrative Science, 2012 [Online]. Available: http://www.narrativescience.com/

[10] Statsheet, 2012 [Online]. Available: http://statsheet.com/