Proceedings, The Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment

Sports Commentary Recommendation System (SCoReS): Machine Learning for Automated Narrative

Greg Lee and **Vadim Bulitko**
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada
{*gmlee* | *bulitko*} *@ualberta.ca*

Elliot Ludvig
Princeton Neuroscience Institute
Princeton University
Princeton, New Jersey, USA
eludvig@princeton.edu

Abstract

Automated sports commentary is a form of automated narrative. Sports commentary exists to keep the viewer informed and entertained. One way to entertain the viewer is by telling brief stories relevant to the game in progress. We introduce a system called the Sports Commentary Recommendation System (SCoReS) that can automatically suggest stories for commentators to tell during games. Through several user studies, we compared commentary using SCoReS to three other types of commentary and show that SCoReS adds significantly to the broadcast across several enjoyment metrics. We also collected interview data from professional sports commentators who positively evaluated a demonstration of the system. We conclude that SCoReS can be a useful broadcast tool, effective at selecting stories that add to the enjoyment and watchability of sports. SCoReS is a step toward automating sports commentary and, thus, automating narrative.

Introduction

Sports broadcasting is a billion-dollar industry. Most professional sports are broadcast to the public on television, reaching millions of homes. The television experience differs in many ways from the live viewing experience, most significantly through its commentary.

Much research has been done into the importance of commentary during sports broadcasting. When watching a game on television, "...the words of the commentator are often given most attention." (Duncan and Hasbrook 1988). The commentary has the effect of drawing the attention of the viewer to the parts of the picture that merit closer attention (Kennedy and Hills 2009), an effect called *italicizing* (Altman 1986). Commentary can also set a mood during a broadcast. A commentator who creates a hostile atmosphere during a broadcast often makes the viewing experience more enjoyable for the viewer (Bryant et al. 1982). The descriptions given in a broadcast are so useful that fans often bring radios to live games in order to listen to the commentators (Michener 1976). Also, some sporting venues now support a handheld video device that provides the spectator with in-game commentary (Ross 2012).

The purpose of commentators is to help the viewer follow the game and to add to its entertainment value. One way to add entertainment to a broadcast is to tell interesting, relevant stories from the sport's past. The sport of baseball is particularly suited to storytelling. Baseball is one of the oldest professional sports in North America, existing since 1876. This longevity provides a rich history from which to draw interesting stories. The more popular commentators are known as "storytellers" (Smith 1995), as they augment the games they call by adding stories that connect baseball's past to its present. A typical baseball game lasts about three hours, but contains only ten minutes of action. This leaves two hours and fifty minutes of time where little is happening on the playing field. This downtime is a good time to tell stories (Ryan 1993).

To illustrate, consider the case where one baseball team is trailing by four runs in the ninth inning. As this is not a particularly interesting situation, it may be a good time for a story. An appropriate story might be that of the Los Angeles Dodgers, who on September 18, 2006, were also trailing by four runs in the bottom of the ninth inning. The Dodgers hit four consecutive home runs to tie the game. The broadcast team could tell this story to the viewer, because the situations are similar. Thus, what is needed is a *mapping* from a game state (a particular point in a game) to an appropriate story.

Our automated system (SCoReS) maps the current game state to an interesting, relevant story for possible use by the commentators. SCoReS 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 baseball games to suggest relevant stories to a (human) broadcast team or, in the absence of a broadcast team (e.g., in a sports video game), to autonomously output a relevant story to the audience.

In this paper, we introduce SCoReS and show that it can be an effective and useful tool for a professional broadcast team. The next section formally defines the problem. We then describe our approach to solving the problem and the methodology for empirical evaluation. We conclude with discussion and suggestions for future work.

Problem Formulation

Sports broadcasts typically involve two types of commentators – play-by-play and colour. Play-by-play commentators' main function is to provide factual information about the game, such as describing the actions of the players and providing statistics. Colour commentators have a more loosely defined function – to provide "colour". They do this in many ways. As they are mostly former athletes in the sport at hand, they can provide expert knowledge. They can also relate stories from their playing careers or post-playing careers, of which they have intimate knowledge. Their knowledge of stories, however vast for a human being, is still limited relative to the total set of baseball stories available. This is where Artificial Intelligence (AI) can be of assistance. Computers can both store many more stories than a human brain as well as quickly compute the quality of a match between each story in the library and the game state at hand.

In this work, we attempt to model an unknown target function that maps game states to relevant stories. As "being interesting" is an ill-defined subjective measure, we evaluated the quality of the mapping by incorporating the selected stories into a simulated broadcast and tested the enjoyment of viewers and the interest of professional commentators. Given a story database, the problem is then to retrieve stories most appropriate for never-before-seen game states during live sports broadcasts. The broader the story database, the more likely an appropriate story can be found that matches to any given game state.

Related Research

To the best of our knowledge, there are no deployed AI systems for colour commentating in real-life sports broadcasts. Thus we turn to related fields, such as automated sportswriting, recorded commentary in simulated games, automated play-by-play commentary, and recommendation systems.

StatSheet (Allen 2012) and *Narrative Science* (Frankel 2012) automatically write stories about sports games that have already happened. 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. Neither Statsheet nor Narrative Science operate with live game data and both are unable to solve our problem of providing live stories during a game. They may, however, provide an additional potential database of stories about past games to augment our current system.

Many modern commercial sports video games use recorded commentary from professional commentators, in order to emulate the sounds of actual sports broadcasts. The *2K Sports Major League Baseball* series is a popular group in this genre and has in the past employed Jon Miller and Joe Morgan, formerly the play-by-play and colour commentators for *ESPN's Sunday Night Baseball*. 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. One example is when on a third strike dropped by a catcher, Miller tells a story about such a play by Mickey Owens that changed the course of the 1941 World Series. This is one of few examples of storytelling in the series, and this story is repeated. Repetition takes away from the entertainment value of the video game, as it is unrealistic and boring.

While automated colour commentary is novel, automated play-by-play commentary has received attention. *Rocco* (and *Rocco 2*), *MIKE* and *Byrne* (Andre et al. 2000) are three well-known automated play-by-play systems. They are used for the RoboCup Simulation League (Federation 2010). The three algorithms make use of the *Soccer Server* within the RoboCup Simulation for their game data. They each generate commentary with templates, filling in team or player names depending upon what is happening in the game.

Within its live online game summaries, Major League Baseball uses a system called *SCOUT* (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 has also been much work on automated storytelling in non-sports video games (Roberts and Isbell 2008). Such systems typically generate or adapt a story involving the player, creating a so-called interactive drama. Automated colour commentary in sports poses additional challenges because the storytelling system has no control over the sports game and the player is not involved in the story being told. On the other hand, SCoReS could be adapted to tell stories in non-sports video games (such as real-time strategy games) using the approach that we describe in the next section.

Recommendation systems attempt to predict a user's rating of an item based on other users' ratings and the characteristics of the item (Koren and Bell 2011). This usually requires knowing something about the user, having a corpus of ratings for different items and having a similarity measure for different items. We do not depend on such data in our problem formulation.

Our Approach

Information Retrieval Framework

We approach the problem of automated storytelling in sports as an information retrieval (IR) problem. The game state for which we are seeking an appropriate story is treated as the *query*, and the candidate stories returned by the system are the *documents*. Our system's goal then, is given a game state, to return to the user a ranked list of stories, based on how appropriate they are for the game state. We assume that the game state is available live and that a database of stories has previously been collected.

To compare stories to game states, we extract features from both, such as the score, the teams involved and what type of action is happening (i.e., a home run). Formally, the game state is a vector of n numeric features: $\vec{g} = (g_1, g_2, \ldots g_n)$. Let the binary feature g_1 be 1 if in game state $\vec{g}$ there is a runner on first base. Let the integer feature g_2 be the current inning. Similarly, a baseball story can be described with a vector of p numeric features: $\vec{s} = (s_1, s_2, \ldots s_p)$. For instance, let binary feature s_1 be 1 if the story involves a runner on first base and let integer feature s_2 be the inning number mentioned in the story.

The match quality $D(\vec{g}, \vec{s})$ between a game state $\vec{g}$ and story $\vec{s}$ is an integer on a 5-point scale from 0 for a completely inappropriate match to 4 for a perfect match. This scale is compatible with leading IR scoring metrics, such as Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (NDCG) (Xu and Li 2007). Then the problem is, given a game state $\vec{g}$, to retrieve a story $\vec{s}$ of the highest match quality $D(\vec{g}, \vec{s})$.

Training Data

We solve this problem by using IR techniques and machine learning. A *similarity vector* $\vec{c}$ is computed for a game state specified by feature vector $\vec{g}$ and a story specified by feature vector $\vec{s}$. Each component of vector $\vec{c}$ is the result of comparing one or more features of $\vec{g}$ to one or more relevant features of $\vec{s}$. To compare binary features we use logical connectives. For instance, to match the runner on first base features, a biconditional is taken over the corresponding features: $c = g \leftrightarrow s = 1 \leftrightarrow 1 = 1$. For non-binary features we use feature-specific functions. For instance, to compare the current inning number in g and the inning number in a story s, the similarity feature c is calculated as $(8 - |g - s|)/8$, where values closer to 1 indicate a closer pairing of inning features. Another example is the marquee matchup feature valued between 0 and 1. It indicates how well the story and game state match in terms of the involvement of a strong hitter and a strong pitcher. It is calculated by combining features holding several statistics for the current batter and pitcher in $\vec{g}$ with a story category feature in $\vec{s}$.

The similarity vector $\vec{c}$ indicates how related $\vec{g}$ and $\vec{s}$ are, but does not provide a scalar value. We now need to map $\vec{c}$ to $D(\vec{g}, \vec{s})$ — the 5-point-scale quality of the match between $\vec{g}$ and $\vec{s}$. We use machine learning techniques to create such a mapping from training data $\mathbf{T}$. To build this training data set, we take a set of m game states $\mathbf{G} = \{\vec{g}_1, \ldots, \vec{g}_m\}$ and q stories from our story library $\mathbf{S} = \{\vec{s}_1, \ldots, \vec{s}_q\}$ and form a set of $m \cdot q$ similarity vectors $\vec{c}$. We then label each similarity vector with the ground-truth value of the quality of match between the corresponding game state and the story. Mathematically: $\mathbf{T} = \{(\vec{c}, D(\vec{g}, \vec{s})) \mid \vec{g} \in \mathbf{G}, \vec{s} \in \mathbf{S}, \vec{c}$ is the similarity vector for $\vec{g}$ and $\vec{s}\,\}$.

Ranker

IR algorithms are generally divided into three groups – pointwise, pairwise and listwise (Liu et al. 2008). Pointwise algorithms are regression and classification algorithms, with mean-squared error typically used as an error function. Pairwise algorithms perform an incomplete ordering on the data. Listwise algorithms make direct use of IR metrics to search for a good ranking of documents.

After unsuccessfully experimenting with solely pointwise algorithms (such as k-nearest neighbours) and solely listwise algorithms (such as SVM MAP), we turned to a hybrid approach, involving listwise, pointwise and domain-specific methods. For the main algorithm, we adapted AdaRank (Xu and Li 2007), a listwise algorithm based on AdaBoost (Freund and Schapire 1995). AdaRank forms a "strong" ranker by iteratively selecting and combining "weak" rankers. A ranker performs a mapping: $R : \mathbf{S} \to \{1, 2, \ldots, q\}$ from the story library $\mathbf{S}$ to an ordering of $\mathbf{S}$, for a given $\vec{g}$. A weak ranker uses a single component of the similarity vector $\vec{c}$ to rank (i.e., sort) the training data $\mathbf{T}$. Each weak ranker has a "vote" on the final ranking, based on how its mapping of $\mathbf{S}$ to $\{1, 2, \ldots, q\}$ over the training data was scored according to a chosen IR scoring metric (described below).

Algorithm 1 SCoReS AdaRank.
Input: $\mathbf{T}$: training data, m: number of game states, M: IR scoring function, k: number of iterations, y: number of tie-breaking features.
Output: R: ranker.

```
1:  partition $\mathbf{T}$ by game state: $\mathbf{T} = \mathbf{T}_1 \cup \mathbf{T}_2 \cup \ldots \cup \mathbf{T}_m$
2:  sort each $\mathbf{T}_i$ by $D$ values, yielding ground truths $\mathcal{T}_i$
3:  initialize weights $w(\mathbf{G})$ to 1
4:  initialize ranker $R$ and weak ranker confidences $A$ to $\emptyset$
5:  get all combinations $B$ of length $y + 1$ from $\mathbf{T}$
6:  for each iteration up to $k$ do
7:      $\mu \leftarrow 0$
8:      for each combination $b$ in $B$ do
9:          if $b(1) \notin R$ then
10:             for $i = 1, \ldots, m$ do
11:                 sort $\mathbf{T}_i$ by $b$, yielding $\mathbf{T}'_i$
12:                 $v(i) \leftarrow w(i) \cdot M(\mathbf{T}'_i, \mathcal{T}_i)$
13:             if mean$(v) > \mu$ then
14:                 $\mu \leftarrow$mean$(v)$
15:                 $r \leftarrow b$
16:     add $r$ to $R$
17:     calculate $\alpha$
18:     add $\alpha$ to $A$
19:     update $w$
```

SCoReS AdaRank (Algorithm 1) accepts as input training data $\mathbf{T}$, the number of game states in $\mathbf{T}$, m, an IR scoring function M, the number of weak rankers to build the strong ranker k, and the number of tie-breaking features to use y.

As a small, concrete example, let M be NDCG (Xu and Li 2007), $k = 2$, and $y = 1$. Let the similarity vectors $\vec{c}$ in $\mathbf{T}$ consist of the following features: home run (c_1), strikeout (c_2), inning (c_3), and marquee matchup (c_4). The first two are binary: they are 1 if both the story and game state involve a home run (or both involve a strikeout), and 0 otherwise. The marquee matchup and inning features are computed as previously described.

In line 1, SCoReS AdaRank first partitions $\mathbf{T}$ by its m constituent games states, where i runs from 1 to m. This is done because stories can only be meaningfully sorted by the match quality values (D) for a given game state.

The ground truth rankings $\mathcal{T}$ are then calculated (line 2) for use in evaluating weak rankers (line 12). All weights are initialized to 1 (line 3) as all game states are initially equally important. The main ranker R and its corresponding confidence values A are initialized to be empty sets (line 4). The set of feature combinations to be considered for use in weak rankers, B, is then calculated based on the number of features in each $\vec{c}$ in $\mathbf{T}$, and the number of features to use for tie-breaking y (line 5). In our example, a feature combination can be (c_2, c_4) — the strikeout feature used as the main sorter, with the marquee matchup feature as a tiebreaker.

At each iteration of SCoReS AdaRank, elements b of B whose first elements have not yet been used in R are considered as possible weak rankers (lines 6-19). Thus, each feature of $\vec{c}$ may only be used once as the main sorter in a weak ranker. For each game state, the weighted score v of sorting $\mathbf{T}_i$ by b is calculated, using the scoring function M and current weights w (lines 10-12). If the mean weighted score v for b is greater than the maximum encountered so far in this iteration, then the feature to be used in the weak ranker r for this iteration is set to b (lines 13-15). After evaluating all valid elements of B, the best weak ranker for this iteration is added to the main ranker R (line 16) and A and w are updated (lines 17-19) as in (Xu and Li 2007). The data is re-weighted after each iteration of SCoReS AdaRank, so that examples that have been incorrectly classified so far are given more weight (line 19).

In our example, the final ranker R can be $((c_2, c_3), (c_3, c_4))$ with corresponding A $(0.7, 0.3)$. This means the weak ranker using the strikeout feature to sort and the inning feature as a tiebreaker receives 70% of the vote for ranking $\mathbf{S}$, and the inning feature combined with the marquee matchup feature as a tiebreaker receives 30% of the vote.

Evaluator and Contextualizer

Though the Ranker R output by SCoReS AdaRank provides a ranked list of documents (stories for SCoReS), it does not provide a value for D for these documents. Thus, there is always a top-ranked story for a game state, but SCoReS AdaRank provides no indication as to how "good" the top ranked story is. We added an *Evaluator* to provide an estimate of D. The Evaluator can then be used as a threshold to ensure the top-ranked SCoReS AdaRank story is worth telling. While pointwise algorithms may be unreliable for ranking due to their lack of focus on the top of a ranked list, they can be highly accurate in terms of mean-squared error for game state – story pairs. Finally, we use several domain specific features to determine whether it is a good time to tell a story. An example from baseball would be if there are two strikes on the batter with two out, it would be a bad time to tell a story as the inning could easily end on the next pitch (with a strikeout). Stories can only be told once per game, and only one story is allowed every ten pitches.

SCoReS Overall

Our SCoReS system thus consists of a Ranker (learned by SCoReS AdaRank), an Evaluator (learned by a pointwise algorithm) and a Contextualizer (manually designed) as shown in Algorithm 2. SCoReS processes each game state in a given (novel) game $\mathbf{G}'$ (line 1). If the current game state $\vec{g}$ is appropriate for a story (line 2), we create similarity vectors of $\vec{g}$ and each story in $\mathbf{S}$ (line 3). The similarity vectors are then sorted with the ranker R and confidences A learned by SCoReS AdaRank offline (line 4). The top ranked story is extracted in line 5 and then scored by the Evaluator E in line 6. If the story passes the provided threshold t, and SCoReS is operating autonomously, the story is output to the viewer (line 7). If a broadcast team is using SCoReS, the top three stories can be output.

Algorithm 2 SCoReS.
Input: $\mathbf{G}'$: game states for a live game , $\mathbf{S}$: our story library, R: ranker from SCoReS AdaRank, A: weak ranker confidences for R, E: evaluator, t threshold for evaluator.

```
1: for each game state $\vec{g}$ in $\mathbf{G}'$ do
2:     if should tell a story in $\vec{g}$ then
3:         create $\{\vec{c}\}$, comparing $\vec{g}$ to $\vec{s} \in \mathbf{S}$
4:         rank $\{\vec{c}\}$ with $R$ and $A$
5:         $\vec{s}_\circ \leftarrow$ top ranked story
6:         if $E(\vec{s}_\circ) \geq t$
7:         output $\vec{s}_\circ$ to broadcast team (or viewer)
```

Empirical Evaluation

To evaluate the quality of SCoReS, we conducted a series of empirical studies where we asked potential users to evaluate the system. We examined several possible ways in which SCoReS might improve broadcast quality: by providing generic commentary, adding stories to existing commentary, picking context appropriate stories, or assisting professional commentators to select stories.

We conducted three experiments to evaluate whether SCoReS improved broadcast quality in any of these ways. Experiments A and B were user studies, consisting of participants watching short video clips from baseball games. Experiment A tested whether commentary was beneficial within our game library. Experiment B tested whether there was a reason to tell stories in a broadcast within our game library and whether the stories selected needed to be in proper context. Experiment C was a demonstration of the SCoReS system to professional commentators. This experiment tested whether SCoReS has the potential to be successfully deployed in a professional broadcast setting.

For all of the experiments, the video clips shown were extracted from two minor league baseball games: the July 15, 2009 AAA All-Star game between the International League and the Pacific Coast League and the April 7, 2011 game between the Buffalo Bisons and Syracuse Chiefs (also a AAA baseball game). The stories used in the experiments were taken from Wikipedia and two books of historic baseball stories (Neyer 2008; Lee and Prime 2008). Features for both the game states and the stories were extracted manually. The story database used in training was the same as that used in testing (i.e., the stories presented to participants).

Experiment A

In this experiment, we compared *SCoReS Commentary* to two different types of commentary. For *No Commentary*, we removed the commentary from the broadcast, and left 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*

[§]The crowd noise was artificial since we could not remove commentary without removing the actual crowd noise. After removing all the sound from the original broadcast, we added an audio file of crowd noise. This held for all types of commentary.

[¶]Voicing over the commentary was necessary to ensure that the same voices were used in all the commentary types to prevent any

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

Table 1: User study questions asked to participants in Experiment A. Question 2b) appeared only in Experiment B. Participants were asked to rate each question from 1 to 7 (strongly disagree - strongly agree).

Commentary had a story selected by our system inserted (read) into the original professional commentary by pausing said commentary through the length of the story.

We recruited 16 participants from the local community. To measure the performance of the different types of commentary, we evaluated participants' answers to the six questions listed in Table 1. For both games, each participant saw one clip each with *Original Commentary*, *SCoReS Commentary*, and *No Commentary*. Thus, each participant saw six video clips in total. The clips within a given game were always shown in the same (chronological) order, but the commentary was seen in different orders. For this experiment, SCoReS was trained on $m = 25$ game states ($\mathbf{G}$) and $q = 88$ stories ($\mathbf{S}$), totalling 2200 training data ($\mathbf{T}$). Figure 1 shows that *SCoReS Commentary* ranked significantly higher than *No Commentary* across all metrics. A one-tailed t-test was used to check for significance in the results of all experiments, and a Holm-Sidak test was used to correct for multiple comparisons.

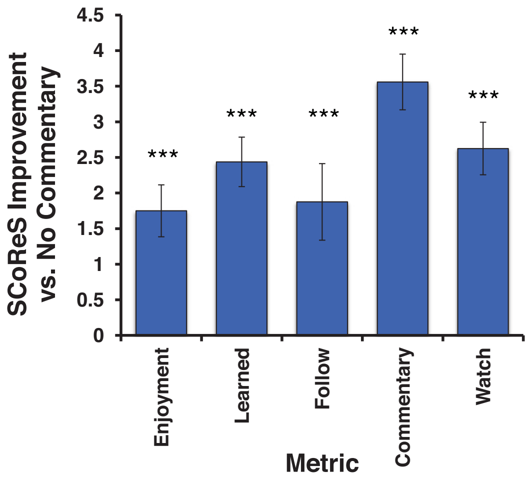

Figure 1: Mean (+/- Standard Error of the Mean) difference between *SCoReS Commentary* and *No Commentary*. *** indicates $p < 0.001$.

Experiment B

Having collected evidence that commentary itself adds entertainment to a game broadcast, we performed a second

biases. We used voiceovers in all video clips for consistency.

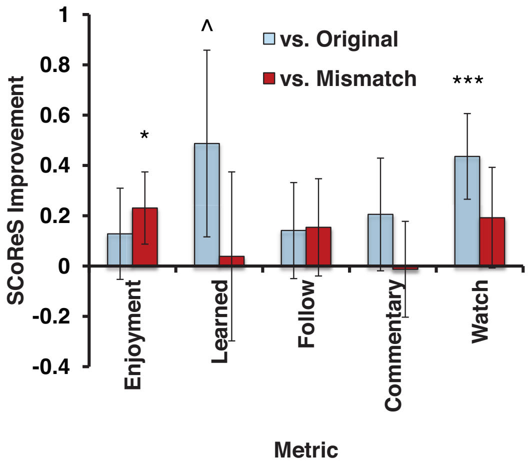

Figure 2: Mean (+/- Standard Error of the Mean) difference between *SCoReS Commentary* and *Original Commentary* or *Mismatch Commentary*. *** indicates $p < 0.001$; * indicates $p < 0.05$; $\wedge$ indicates $p < 0.1$.

user study, replacing *No Commentary* with *Mismatch Commentary*. For *Mismatch Commentary*, we inserted a story in the same place as we would with SCoReS, but instead of the story being selected for that game state, it was actually a story chosen by SCoReS for a different game state, in the other game the participant saw. This allowed us to keep the story pool consistent across the conditions thereby controlling for the overall level of interest.

To better assess the value of the different types of commentary, the questions asked of participants were updated for Experiment B. Participants were asked the first five questions in Table 1 using version 'b' of question 2. We recruited 39 students from the University of Alberta, who were self-described baseball fans. For each of the two games, each participant saw one clip each with *Original Commentary*, *SCoReS Commentary*, and *Mismatch Commentary*. Pilot data showed that participants ranked the first clip they saw lower than others, independently of commentary type. To avoid this bias, we inserted a "dummy clip", which preceded the three video clips of interest. Thus each participant saw eight video clips in total. To choose stories for this experiment, SCoReS was trained on 40 game states and 110 stories, totalling 4400 training data.

Figure 2 shows the mean difference between *SCoReS Commentary* and both *Original Commentary* and *Mismatch Commentary*. *SCoReS Commentary* was ranked higher than the *Original Commentary* for the "Viewing this clip made me more interested in watching baseball" metric, with $p < 0.001$. This shows that adding stories to commentary can improve a broadcast. *SCoReS Commentary* was ranked higher than *Mismatch Commentary* for the "I found this viewing experience enjoyable" metric with $p < 0.01$. This shows that intelligently adding stories to commentary can be more enjoyable to the viewer than adding random stories.

Experiment C

In the third experiment, we demonstrated SCoReS to professional commentators. To evaluate the potential usefulness of SCoReS, we first asked them, “Would you be interested in a system that suggests interesting stories during a game?”. Then we demonstrated SCoReS delivering three stories for four different clips to them. After each clip, we asked, “Would you tell any of the suggested stories?”. After the full demonstration we asked, “Would you be interested in this particular system?”.

The four commentators were Mark Lee and Kevin Weekes, a play-by-play and colour commentating team for the Canadian Broadcasting Corporation (CBC’s) *Hockey Night in Canada* programme, Dan Robertson, play-by-play commentator for various sports for Eastlink Television, and Len Hawley, play-by-play commentator for the University of Acadia men’s varsity hockey team.

All four commentators said they believed a system such as SCoReS would be a useful tool to have at their disposal. When asked about SCoReS itself, they all answered that it would be a great tool not only for baseball, but also for other sports, with a few tweaks. In particular, stories would need to be kept short for hockey broadcasts, as hockey is a faster-paced sport. Each of the four commentators said they would have told a story in two of the four demonstration video clips. Specifically, both Len Hawley and Dan Robertson said they would tell one of the stories had it happened a few days earlier. Mark Lee and Kevin Weekes had no conditions on their positive answers.

Discussion and Future Work

We have shown that SCoReS has a statistically significant positive influence on the sports viewing experience across several metrics. Additionally, national commentators Mark Lee and Kevin Weekes were particularly positive about the system, suggesting its appeal for a broad audience. This indicates that implementing SCoReS in professional commentating may lead to a better viewing experience.

Participants in the user studies were recruited from subject pools. Thus, they did not necessarily watch the video clips at a time of their choosing, as opposed to a fan watching on television. Additionally, professional commentators generally state why stories they are telling are relevant, to give context to the story. This did not happen during the user studies, because this would have biased participants’ answers to some of the questions. Despite these impediments, SCoReS was able to achieve significant improvements in overall enjoyment and increasing interest in watching baseball, and we surmise that in a more realistic deployment, SCoReS would further improve the entertainment value of sports broadcasts.

In the future, we plan to augment SCoReS with baseball facts. These facts would be shorter bits of information from baseball’s past, that could be told quickly. Also, the system would benefit from an automated bot that could mine the web looking for relevant stories, automatically extracting features and updating the story database. A possible future application of SCoReS is automated storytelling for real-time strategy (RTS) games, such as *StarCraft*, as they have their own sets of features and stories, and are often broadcast with commentary. While there is no “pitch” by which to discretize in these games, timesteps could be used instead. To accelerate the generation of training data, a system such as Amazon’s Mechanical Turk could be used. Implementing these ideas will further establish SCoReS as a valuable tool for sports colour commentary.

Acknowledgements

We would like to thank NSERC and iCORE for funding this research. We would also like to thank Minor League Baseball for providing baseball footage, Len Hawley and Jim Prime for helping with commentary, Mark Lee, Kevin Weekes, Dan Robertson, and Len Hawley for helping evaluate SCoReS, and Acadia University (in particular Brenda Trofanenko) for helping with the user studies.

References

Allen, R. 2012. Statsheet. http://www.statsheet.com.

Altman, R. 1986. Television/Sound. *Studies in entertainment: Critical approaches to mass culture,* 39–54.

Andre, E.; Binsted, K.; Tanaka-Ishii, K.; Luke, S.; Herzog, G.; and Rist, T. 2000. Three RoboCup simulation league commentator systems. *AI Magazine* 76:57–66.

Bryant, J.; Brown, D.; Comisky, P.; and Zillman, D. 1982. Sports and Spectators: Commentary and Appreciation. *Journal of Communication* 109–119.

Duncan, M., and Hasbrook, C. 1988. Denial of power in televised women’s sports. *Sociology of Sport* 5:1–21.

Federation, R. 2010. Robocup. http://www.robocup.org.

Frankel, S. 2012. Narrative science. www.narrativescience.com.

Freund, Y., and Schapire, R. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, 23–37.

Kennedy, E., and Hills, L. 2009. *Sports, Media and Society*. Berg Publishers.

Koren, Y., and Bell, R. M. 2011. Advances in collaborative filtering. In *Recommender Systems Handbook*. Springer. 145–186.

Kory, M. 2012. SCOUT. http://www.baseballprospectus.co m/article.php?articleid=16835.

Lee, B., and Prime, J. 2008. *Baseball Eccentrics*. Triumph Books.

Liu, T.; Wang, J.; Zhang, W.; and Li, H. 2008. Listwise approach to learning to rank - theory and algorithm. In *ICML*, 1192–1199.

Michener, J. 1976. *Sports in America*. Fawcett Books.

Neyer, R. 2008. *Rob Neyer’s Big Book of Baseball Legends: The Truth, The Lies and Everything Else*. Fireside.

Roberts, D. L., and Isbell, C. L. 2008. A Survey and Qualitative Analysis of Recent Advances in Drama Management. *ITSSA* 4(2):61–75.

Ross, S. 2012. FanVision. http://www.fanvision.com/.

Ryan, M.-L. 1993. Narrative in real time: Chronicle, mimesis and plot in the baseball broadcast. *Narra.* 1(2):138 – 155.

Smith, C. 1995. *Storytellers: From Mel Allen to Bob Costas : Sixty Years of Baseball Tales from the Broadcast Booth*. MacMillan Publishing.

Xu, J., and Li, H. 2007. AdaRank: a boosting algorithm for information retrieval. In *SIGIR*, 391–398.