BPM (Bron Performance Metric) for Automated NBA Video Captioning

Brown University Department of Computer Science

Wali Siddiqui, Alan Lucero, Angel Alvarado-Reyes, Robayet Hossain

Submitted in partial fulfillment of the requirements CSCI 1470

Submitted: May 2nd, 2025

Table of Contents

INTRODUCTION

Project Proposal

Automatically generating play-by-play captions for NBA basketball videos is an interesting but challenging deep learning challenge. Our goal in this research was to create a model that emulates a human sports commentator by using an NBA video clip as input and producing a descriptive commentary (caption) as output. This task requires the combination of many concepts we've learned over the past semester, such as CNNs and transformers. We approached it using segmentation, CNNS, and a transformer-based sequence-to-sequence learning architecture, which has been proven effective for video captioning in recent research.

Our approach is based on the large NSVA dataset and SportsFormer, a state-of-the-art transformer-based model introduced in 2022. The NSVA (NBA Sports Video Analysis) dataset, which contains more than 32,000 annotated NBA video clips, is critical to the evaluation of our model's accuracy. By fusing several streams of video features using a single transformer architecture, SportsFormer offered a solid foundation and achieved impressive results on a difficult challenge. Our goal was to use different but similar data to recreate SportsFormers' core ideas in our own way, then use that data to try to determine how accurate our model was. We specifically wanted to use TensorFlow to implement the model (the original used a PyTorch implementation) and investigate potential stretch goals like adding audio commentary cues and possibly producing captions in a different language. Our approach to outcomes, difficulties, and lessons discovered during the project is compiled in this reflection document.

METHODOLOGY

Model Architecture

For video captioning, we modified the SportsFormer architecture, which uses a transformer-based encoder-decoder. In our implementation, the caption text is produced by a decoder module after the input video has been processed by an encoder module. To handle various parts of the video, we used SportsFormer's multi-encoder technique instead of a single

encoder. In particular, like the original design, we employ a coarse visual encoder to extract the global scene context from video frames and additional fine-grained encoders to extract specific features (such as the positions of players, balls, and baskets). To recognize players and the ball in every frame, for instance, we used pre-trained computer vision models. We sent these detections into the model as auxiliary feature streams. After that, the transformer architecture conditions the text generation on this rich video representation by fusing the coarse and fine-grained data using cross-attention. Lastly, the caption sequence is generated one word at a time by a transformer-based decoder that was initialized from a language model checkpoint. The model attempts to learn a route of mapping from video frame sequences to word sequences that describe the play, thanks to this end-to-end sequence-to-sequence transformer design. It's important to note that our model is a greatly simplified version of the one used in the original NSVA research paper, and our poor results indicate that the connection between the video sequences and annotations is not complete.

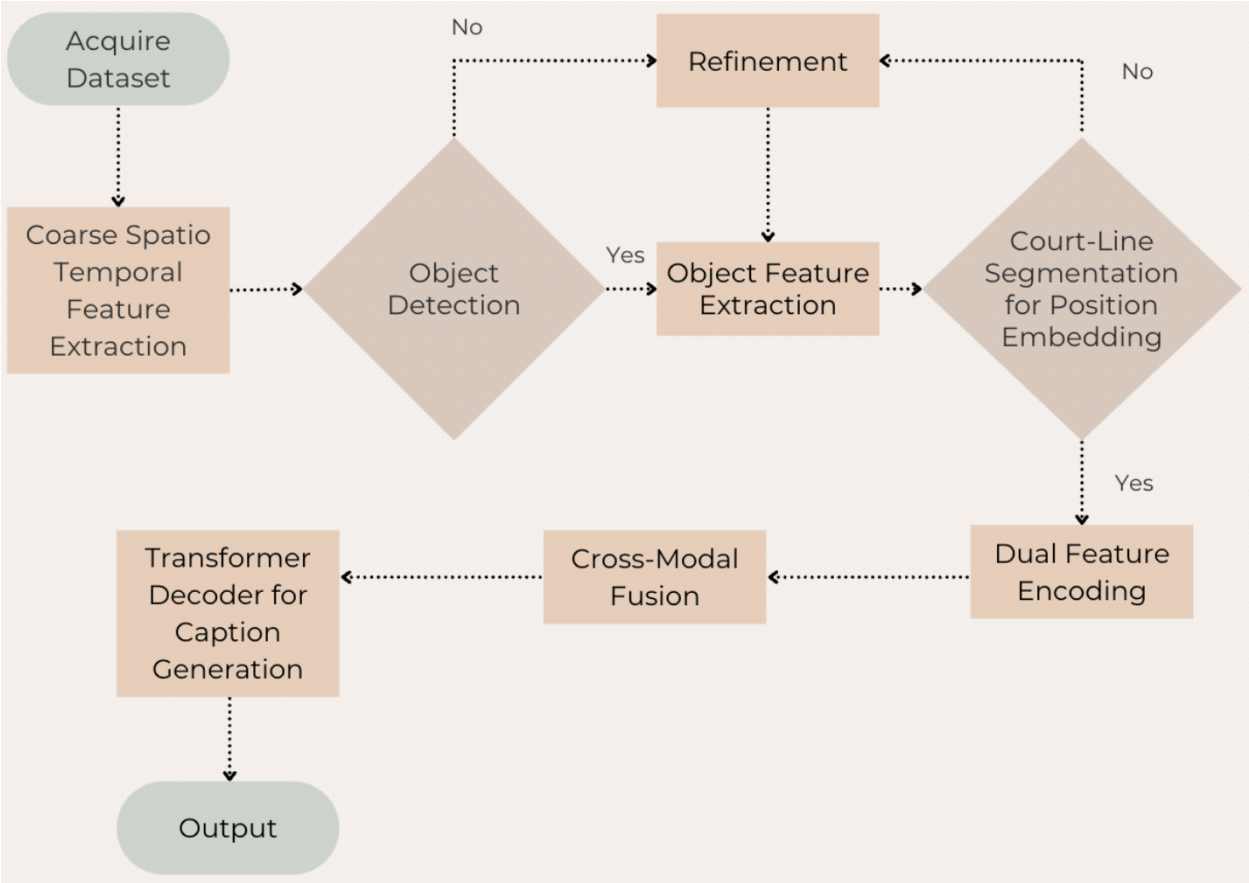

Figure 1: Model architecture that we are seeking to replicate in our project.

Data Handling

Managing the video data was a key component of our approach. We modified our methodology by gathering roughly 165 highlight videos from the 2018–2019 NBA season that were accessible to the general public on YouTube, as the official NSVA dataset was not made available to the public (and the original authors did not respond to us). These highlight videos served as a suitable stand-in for training and assessment since they closely resembled the format and content of the original NSVA dataset, with easily definable beginnings and ends to each clip.

A lot of preprocessing was needed for each video clip, which was usually only a few seconds long. Since it was impractical to read raw video on the go, we downloaded the films using yt-dlp and similar tools, then transformed each video into a series of JPEG image frames to be used as input to our model. After being scaled and normalized, these frames were effectively saved on disk for training, either in organized directories or as NumPy arrays.

Since we used highlight videos in order to train the models, our model operates on high-impact plays, which presumably should have actions that are easier to define. Our reference captions that we are seeking to produce with our models are the commentary closed captioning on the highlight video. This is non-ideal because it introduces variability as the commentator is not robotic in his commentary. The only truly consistent feature of commentary we found was that the commentator would normally identify the star player in the highlight. This is the best we were able to do, as our data had to be webscraped, which posed a huge constraint in our project.

Overall, the acquisition and preprocessing of our data ended up being the most challenging part of the project. It was difficult to deal with the large amount of compute needed to process 80 hours of footage (our intended data set size), and we had to settle for a reduced dataset size and much less preprocessing than the NSVA paper, which greatly affected our results.

Framework Transition

The framework selection was a significant distinction in our project. We developed our replica using TensorFlow; however, the original SportsFormer codebase and tools are implemented in PyTorch (with a few JAX components). We approached this by going over the original code and replicating what we understood into our model. We constructed multi-head attention layers, transformer blocks, and embedding layers similar to those in the PyTorch version of the model using the Keras.

We looked into evaluation metrics for our model and gravitated towards BLEU and METEOR, but eventually settled on just BLEU. Minor hyperparameter adjustments were required in certain instances to consider framework variations (for example, optimizer dynamics in Adam can vary slightly in default settings between frameworks). All things considered, this framework re-adaptation using tensorflow was a learning exercise that compelled us to go beyond merely reusing the original code and gain a thorough understanding of the model's internal operations.

Stretch Goals Update

At the beginning of the project, we ambitiously set two stretch goals: (1) incorporating audio features into the video captioning model, and (2) generating multilingual captions. Although we did not manage to complete these extensions by the project's end, the work we did in preparing for them meaningfully shaped our approach to the project.

Our inability to pursue these stretch goals can be attributed to the huge amount of project time that had to be dedicated to dataset acquisition and preprocessing. Since the official NSVA dataset wasn't publicly available, we sourced about 165 NBA highlight clips from YouTube, encountering significant scraping issues due to cookie management problems, authentication barriers, and online bot detection software. As a result, securing a large, consistent dataset took far longer than anticipated, and we prioritized building a fully functional and high-quality baseline model.

With regards to audio, we believe we would have struggled greatly to incorporate this data into the model. Given the unpredictable nature of the data we were working with, it was unlikely that the audio data would have a meaningfully beneficial impact on our results. We

determined fairly early on that the time investment would not be worth the return. We were, however, much closer to integrating multilingual captions. BLEU itself was created to evaluate the accuracy of translations so we understood where we would need to go with the training to implement this. Once we saw how many of our generated captions contained no real meaning or discernable theme, we figured there was not a sufficient 'ground truth' to evaluate the translations on and instead chose to focus on our accuracy in English.

We are certain that these stretch objectives would have been achievable if web scraping difficulties had been less severe or if access to a bigger, cleaned dataset like NSVA had been available. Still, the thought we put towards the implementation of these ambitious goals gave us a much deeper conceptual understanding of the task at hand and proved critical to our overall final model. The project's foundations set us up for future growth in our ability to execute more advanced goals, and we see these unmet objectives as opportunities that we are now ready to seize rather than as failures.

RESULTS

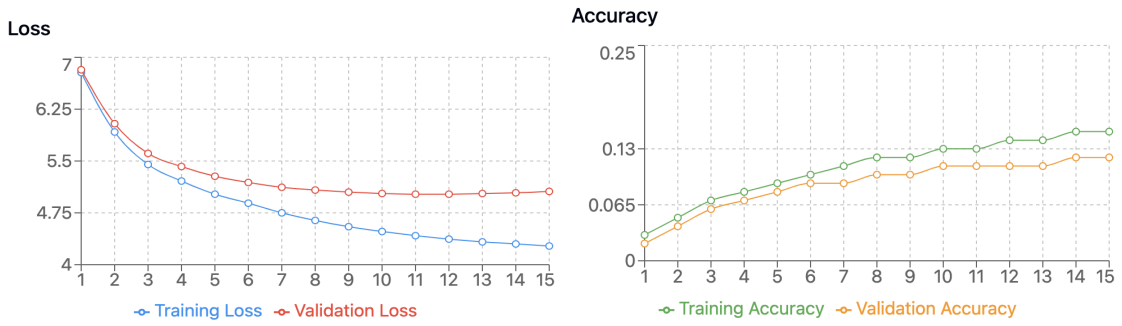

Figure 2: Training and validation loss (left image) lowers to around 4.5 within 15 epochs. Training and validation accuracies (right image) peaked at ~15%.

Figure 2 shows the loss and accuracy of our results. Our validation accuracy peaked at ~15%, indicating that our model was functional, yet inaccurate. Our loss began declining well, indicating that at the start of training, the model was certainly learning well, but it leveled off quickly, indicating that it was likely underfitting early on. This made sense since the amount of data that our model trained on was quite small. Furthermore, there was no pattern to the teams

that were selected for the highlights, so there were likely too many features to learn from, given our limited data and vast variability.

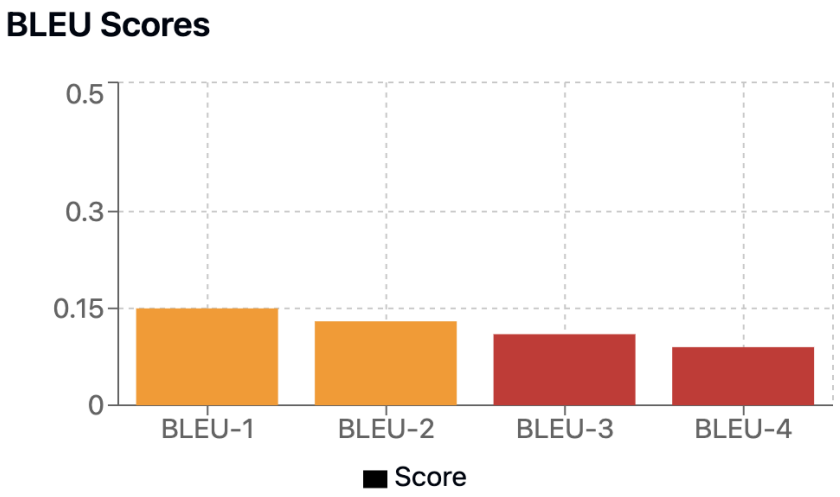

Figure 3: BLEU Scores for Machine Translation Evaluation. The chart presents BLEU-1, BLEU-2, BLEU-3, and BLEU-4 scores, each indicating different levels of n-gram precision in a translation model's output

This chart shows the BLEU (Bilingual Evaluation Understudy) scores for a machine translation model's performance. The BLEU-1 score represents unigram precision, BLEU-2 indicates bigram precision, and so on up to BLEU-4, which includes up to four consecutive words in the comparison. As observed, the scores decrease as the n-gram length increases, which suggests that while the model does well with single-word matches (BLEU-1), its accuracy diminishes with more complex multi-word phrase matches (BLEU-3 and BLEU-4).

Final Validation Loss	0.67
Final Training Accuracy	0.83
Final Validation Accuracy	0.93

Table 1: Validation loss and training accuracy when predicting the next word of a caption

When predicting the next word of a caption, our model performed well, boasting a final validation accuracy of 93.4%, indicating that even while underfitting and with limited data, our model had the potential to learn well.

```
Example 3:
Ground truth: he's getting there hey that's a break
Prediction: he's almost through now hey what a play

Example 4:
Ground truth: curry and he's going to the line looking
Prediction: curry goes to the line and he's focused in

Example 5:
Ground truth: [ applause ]
Prediction: crowd at at at at at at at at
```

Figure 4: Individual examples of ground truth and generated output

Figure 4 shows examples of the captions our model generated and the ground truth captions, which are the actual captions of images that our model is trying to reproduce. Example 5 shows an instance where the ground truth had a very vague and unintelligible output, making it hard for the model to reproduce, which resulted in the incoherent response shown. Example 4 shows an instance where the ground truth has coherent information to reproduce, and the model performs a similar prediction, indicating that for more extensive and sensical captions, our model has the potential to work well. It is important to note that the examples above display some of our best examples (4) and some of our more common ones (5).

Overall, the model's training results suggest that while it was able to learn and improve over time, it faced challenges due to limited data and the complexity of the task. The BLEU scores indicate that the model was better at matching individual words than more complex phrases, highlighting areas where it struggled with longer sequences. However, the high final validation accuracy implies that the model has the potential to perform well with more data and clearer input.

CHALLENGES

The greatest challenge we faced in this project was obtaining the data. The original researchers were unresponsive to our various pleas for their data, so we resorted to web scraping the data from YouTube videos. Thus, all our training and evaluation had to be done using

YouTube footage and auto-generated captioning, which greatly affected our results because these captions are not standardized, unlike the official NBA play-by-play captioning that NSVA used. This introduced variability that greatly affected our model and our accuracy.

Additionally, the segmentation of our NBA highlights greatly affected the accuracy of our model because we struggled to identify the start and end of each clip, and instead chose to segment using set amounts of time. This greatly reduced our model's ability to accurately produce what is happening in each video clip because our segmentations do not coincide with the actual highlights, and could potentially contain parts of two separate highlights.

Towards the end of the project, we were able to get the NBA API to work and load in the play-by-play data for the various NBA games that we scraped from YouTube. Unfortunately, since we were dealing with highlight videos and not full game footage, aligning the play-by-play data to the highlights proved to be a challenge. We were not able to implement the NBA API because we lacked the metadata (mainly time information) necessary to ensure seamless alignment of the game footage with the play-by-play data.

REFLECTION

Looking back on this project, our team has learned a great deal about conducting deep learning research and engineering at scale. There are several key takeaways and reflections worth highlighting.

First, we learned the value of meticulous data management by dealing with a sizable dataset. We discovered that for overall system efficiency, preprocessing and data loading are equally as important as model architecture. The handling of our data was the source of many arguments and debates, and countless hours were spent discussing the optimal method of obtaining the data while minimizing computational expense. One of the more sobering lessons was probably that scalability problems arise everywhere: if done incorrectly, even counting 32,000 video files may become a bottleneck. We now know why industrial initiatives spend so much money on data engineers; with deep learning, the adage "garbage in, garbage out" is true, and it is difficult to guarantee high-quality, well-structured data at scale.

Looking through the code of the original paper was also incredibly humbling. Although we did our best to create an advanced model that could generate captions for the NBA footage,

our model was nowhere near as complex as the model used in the NSVA. Regardless, given the scope of the class, we believe we were able to use the content of the course to the best of our ability to emulate the more advanced model our project sought to replicate. Specifically, we felt well equipped to deal with image classification and object detection issues using YOLOv5, but integrating everything into the transformer and working with the attention blocks felt much more difficult.

More personally, this project served as a crash lesson in teamwork and project management in the context of deep learning research. While still cooperating and evaluating each other's work, we discovered how to assign duties according to our individual abilities. For example, one team member concentrated on data engineering, another on model development, and a third on experimentation and analysis. Our collaboration was put to the test when we integrated the data pipeline, model, training loop, and assessment script. This experience also showed us the importance of clear communication, as evidenced by the agreement on data format contracts and intermediate output check-ins, for example. The iterative nature of research was also demonstrated to us; not everything was successful the first time. When we ran into obstacles, we had to modify our plan (for instance, we changed our timeline and prioritized specific experiments when training was slower than projected). We will use these lessons of flexibility and tenacity in our next endeavors. Finally, it was really satisfying to watch the model gradually learn to "commentate" basketball games. It brought to mind the reason we were interested in deep learning: times when the model generated an especially perceptive or human-like caption made all of the data manipulation and debugging worthwhile. As we wrap up this research, we have a better grasp of deep learning's potential as well as its difficulties. We take pride in our accomplishments and look forward to using these lessons to tackle even more challenging issues in the future.