DL Workshop – Basketball Shot Type Classification

Omer Mazig

October 2023

Abstract

In the domain of sports analytics, the precise classification of basketball shot types - e.g., jump shot, hook shot, layup, dunk, etc. - is a critical endeavor, with the potential to impact players training, performance analysis, and fan engagement. This project introduces an automatic, deep learning solution, by fine-tuning a pre-trained Video Masked Autoencoder (VideoMAE) model, for accurate shot type classification in NBA videos and beyond.

Our methodology leverages the powerful capabilities of VideoMAE[1] - a data-efficient self-supervised video pre-training model – and uses a relevantly fine-tuned version of it as the base model for our classification model. Customized video preprocessing techniques, including data augmentation, are applied to enhance model generalization. The model's capacity to predict shot types, even when subtle visual differences exist between certain types, is a key focus of this project.

This paper outlines the experiments conducted, including dataset creation, model training, and evaluation metrics. Quantitative results demonstrate the model's effectiveness in accurate shot type classification.

Acknowledging the scope and limitations of this project, I suggest potential avenues for future work, emphasizing the model's versatility beyond NBA videos and its applicability in various domains. This work contributes to the evolving field of computer vision in sports analytics, providing valuable insights and practical applications for shot type classification, with implications for coaching, performance analysis, and fan engagement.

1. Introduction

As far as I can discern - pending confirmation from the NBA - the classification of the approximately 200 shots per game into distinct types has traditionally been a manual process. This method, while likely time-consuming, has been subject to human subjectivity and errors. Recognizing the need for a more efficient and consistent approach, this research introduces a

[1] https://arxiv.org/abs/2203.12602

new solution: a deep learning model fine-tuned from a pre-trained Video Masked Autoencoder (VideoMAE). Our aim is to automate the shot type classification process, not only to expedite it but also to enhance its accuracy.

At the core of our solution is the fine-tuned VideoMAE model, a self-supervised video pre-training model. We utilized a pre-trained version that had previously undergone fine-tuning on a similar action dataset, subsequently fine-tuning it ourselves on a dataset of NBA shot videos.

To enable this automated classification system, a pivotal and substantial aspect of our work involved the creation of a specialized dataset. This dataset, consisting of 8000 shot videos lasting about 4-5 seconds each, was meticulously curated. We obtained the videos from the NBA.COM API, tailored their parameters to match the requirements of the underlying VideoMAE model, and condensed them to capture the pivotal seconds encompassing the shot's action. As part of this project, we also provide a readily applicable notebook for the creation of such datasets, allowing users to customize their shot type preferences, video parameters, and other specifications.

In the subsequent pages, we dive further into the specifics of our methodology, dataset preparation, model fine-tuning, and the evaluation metrics used to assess the effectiveness of our shot type classification system. The presentation of quantitative results serves to underscore the model's competence in delivering precise classifications.

2. Method

2.1 Data Collection and Preparation

The creation of a specialized dataset is a pivotal aspect of our project, forming the foundation for the training and evaluation of our classification model. Notably, there was no pre-existing shot type dataset available on the internet, underscoring the necessity for a custom dataset tailored to our specific objectives.

Our dataset consists of evenly distributed 10,000 short videos of 5 basketball shot types:

- Dunk
- Jump Shot
- Floating Jump Shot
- Reverse Layup
- Turnaround Hook Shot

These categories were selected based on their relevance to the classification task, considering the distinctive visual characteristics of each shot type.

To amass this dataset, we embarked on an extensive data collection process, extracting play-by-play information from NBA games from the last 13 years. Each play-by-play entry included details about the type of shot attempted, the time of the event according to the game clock, and other relevant information.

We utilized a versatile data gathering script, which systematically obtained video clips corresponding to the recorded shot attempts. The NBA.COM API, and specifically the open sourced `nba_api` library[2], served as a valuable resource for accessing these video clips. We identified each video clip based on the associated game event from the play-by-play data and downloaded it using another open-sourced library called `youtube-dl`[3]. a few key notes on the video collection:

- For each class, we collected no more than 2 videos from each NBA game. This was to keep the dataset diverse and avoid overfitting.
- We iterated over a randomized order of both the games and the play-by-play entries within each game. This was done to maintain dataset diversity and prevent bias towards a specific year or game portion.

The video clips retrieved from the API contained more than 3-4 seconds of the shot action itself. To optimize the dataset for shot type classification, we engaged in meticulous curation. This involved several key steps:

1. Shot Clock Detection: As part of the play-by-play data, we had the shot time according to the game clock for each shot. We used an OCR library called `tesseract`[4] to detect the game clock reading every second of video.
2. Video Trimming: Once the reading matched the shot time information, we trimmed the video using OPEN CV around that point. Few key notes:
    a. In about 50% of the videos, we failed to detect a matching shot time. Since the NBA provides access to over 1,000,000 shot videos, we moved on from those.
    b. Some play-by-play entries had inaccurately recorded shot times, leading to defective videos with no shot content. To balance video length and defect rate, we settled on a 4-second pre-shot and 1-second post-shot configuration, resulting in mostly 5-second videos with shot content, with ~5% defect chance.

[2] https://github.com/swar/nba_api
[3] https://github.com/ytdl-org/youtube-dl
[4] https://github.com/tesseract-ocr/tesseract

3. Customization of Video Parameters: To ensure compatibility with the underlying Video Masked Autoencoder (VideoMAE) model, we tailored the video parameters to match the model's requirements:
    a. The resolution was changed from **1280x720** to **320x256.**
    b. The FPS was changed from **60** to **30.**
4. Train-Validate-Test Split: random **80%-10%-10%** split.
5. Data Cleaning: Manually detecting and deleting defective videos on validate and test.

The result of this dataset creation effort is a comprehensive collection of shot videos, each lasting approximately 4-5 seconds. This dataset forms the basis for training and evaluating our automated shot type classification system, providing the necessary ground truth data for our model to learn and generalize across different shot types.

The above-mentioned dataset creation process is also provided as part of the repo as a Ready-To-Use notebook[5]. This notebook enables users to create their own specialized datasets, offering the flexibility to specify shot type preferences, video parameters, and other preferences.

2.2 Model Architecture

Pre-trained Video Masked Autoencoder (VideoMAE):

Video Masked Autoencoder (VideoMAE) is a self-supervised video pre-training model that forms the core of our shot type classification system. VideoMAE has demonstrated data-efficient learning capabilities for self-supervised video pre-training, making it a suitable choice for our task.

For this project, we utilized a specific version of VideoMAE that was pre-trained for 1600 epochs in a self-supervised way and fine-tuned in a supervised way on Kinetics-400[6]. The Kinetics-400 dataset, comprising 400 human action classes, includes numerous sport-related actions, making it particularly relevant to our shot classification objectives.

Fine-Tuning Process:

The fine-tuning process is a vital step in adapting the pre-trained VideoMAE model to our specialized dataset. It involves replacing the pre-trained encoder's classification head with a

[5] `create_videos_bank.ipynb`
[6] https://paperswithcode.com/dataset/kinetics-400-1

randomly initialized one, tailored to our specific shot classification problem.

This approach allows our model to inherit valuable features and representations from the pre-trained model, which had already undergone self-supervised pre-training on a large-scale similar dataset (Kinetics-400). This significantly accelerates the training process and enhances its performance, as it leverages the knowledge gained from the pre-trained model.

In our model architecture, we follow the blueprint outlined in the VideoMAE paper[7], with the classifier adjusted to feature only 5 output neurons, as opposed to 400.

2.3 Data Augmentation

For the training dataset transformations, we use a combination of uniform temporal subsampling, pixel normalization, random cropping, and random horizontal flipping. For the validation and test dataset transformations, we keep the transformation chain the same except for horizontal flipping.

2.4 Training

We conducted model training on Google Colab, utilizing the NVIDIA V100 GPU, over the course of 25 epochs. During training, we employed the following hyperparameters:

- Batch size - 4
- Learning rate – 1.5e-5
- Learning rate schedule - Cosine decay
- Optimizer – AdamW

These settings were chosen to achieve optimal training performance and accuracy. We also implemented an early stopping mechanism and loaded the model with the highest accuracy at the end.

2.5 Inference

During the inference phase, we maintain the same batch size but modify our evaluation process. While training, we use only a portion of the video clip and a frame crop, which promotes generalization—a core concept in VideoMAE. However, for predictions, this method resulted in less relevant results, with a 40% accuracy for 5-class prediction.

[7] https://arxiv.org/abs/2203.12602

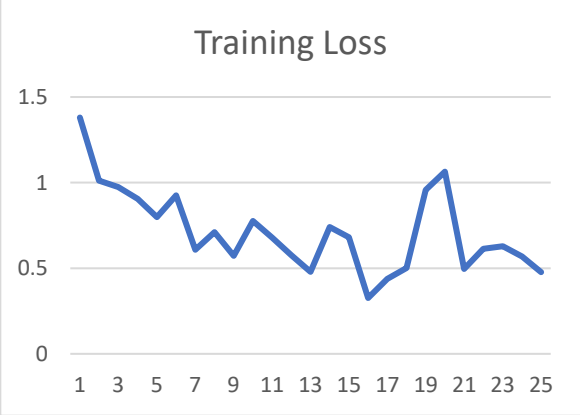
Following the guidance from the VideoMAE paper, we employ a strategy of selecting 5 random ~2-second clips from each video. We apply the same data augmentation techniques as during training, ensuring randomness. For each clip, we compute the class logits, and the prediction is made based on the aggregation of the logits for each class.

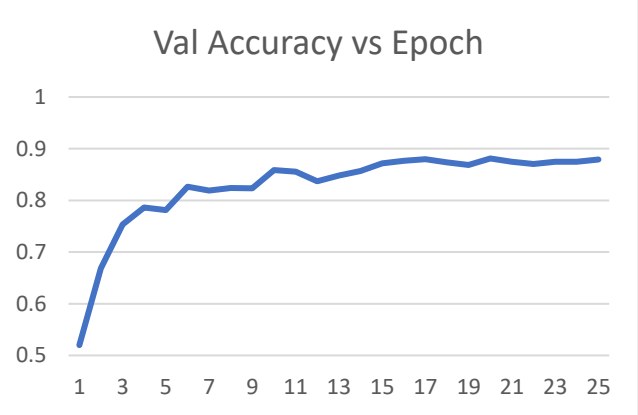
It's worth noting that the original VideoMAE paper used 5 clips multiplied by 3 crops for a total of 15 clips, but we opted for 5 clips to maintain a faster and more memory-efficient inference process, with a similar level of accuracy.

3. Results

In this section, we detail the experiments conducted to assess the performance of our shot type classification model.

We reached accuracy of **87.01%** on the test set, and **87.58%** on validate set.





	precision	recall	f1-score	support
DUNK	0.92	0.88	0.90	194
FLOATING_JUMP_SHOT	0.78	0.85	0.81	188
JUMP_SHOT	0.95	0.94	0.94	191
REVERSE_LAYUP	0.84	0.90	0.87	186
TURNAROUND_HOOK_SHOT	0.86	0.78	0.82	188
accuracy			0.87	947
macro avg	0.87	0.87	0.87	947
weighted avg	0.87	0.87	0.87	947

The results suggest that increasing the number of epochs could improve performance. However, we had limited execution time on Google Colab. The resulted model is on huggingface[8].

We also conducted experiments by approaching the problem as a **multilabel** classification task. In this approach, we maintained the same processes as described earlier, but instead of assigning a single label to each video, we split the label into individual categories and assigned multiple labels to each video based on its content:

Unique classes:
['DUNK', 'FLOATING_JUMP_SHOT', 'JUMP_SHOT', 'REVERSE_LAYUP', 'TURNAROUND_HOOK_SHOT'].
Unique mini classes:
['JUMP', 'SHOT', 'DUNK', 'REVERSE_LAYUP', 'TURNAROUND_HOOK', 'FLOATING'].
Original label: 'TURNAROUND_HOOK_SHOT'
Label: [0, 1, 0, 0, 1, 0]
Video labels: ['SHOT', 'TURNAROUND_HOOK']

And use sigmoid and appropriate loss function.

We achieved an accuracy of **79.62%** and an F1-Score of **88.63%,** which is not as good. The resulted model is on huggingface[9].

Additionally, we initiated a training process with a batch size of 2, focusing on binary classification between a Dunk and a Jump Shot, two extremely visually distinct actions. Remarkably, we attained a **91.83%** accuracy after just 5 epochs, highlighting the efficacy of VideoVMA for simpler classification tasks. The model resulting from this experiment is also available on Hugging Face Model Hub[10].

4. Scope, Limitations, and Future Work

4.1 Scope and Applicability

Our model, trained for shot type classification in NBA videos, offers potential applications in broader contexts. It can be used to classify shooting actions from other broadcasted basketball leagues such as WNBA or EuroLeague – Where no shots are classified by type.

[8] https://huggingface.co/omermazig/videomae-finetuned-nba-5-class-4-batch-8000-vid-multiclass
[9] https://huggingface.co/omermazig/videomae-finetuned-nba-5-class-4-batch-8000-vid-multilabel-4
[10] https://huggingface.co/omermazig/2-class-2-batch-5-epochs

4.2 Challenges and Limitations

The primary challenge revolved around handling a non-synthetic dataset. Developing and refining the automatic dataset creation process, along with curating the dataset, proved to be time-consuming endeavors. Additionally, we encountered limitations due to cooldowns imposed by the NBA's API, which extended the duration required to build new datasets.

Resource constraints also posed challenges, as we had to rely on a paid subscription of Google Colab to access GPUs, which came with limited running time. This limitation hindered our ability to conduct extensive experiments and explore refinements in preprocessing methods and hyperparameters.

4.3 Future Work

While our approach has achieved promising results, it is important to acknowledge the challenges and limitations that come with it.

The primary challenge is optimizing the model's accuracy, especially in distinguishing subtle differences between various shot types. Further exploration of hyperparameters and longer training times could potentially lead to enhanced performance.

Our model's training currently focuses on a subset of 5 shot types, whereas the NBA API lists over 70 shot types. Expanding the model's training to encompass a broader spectrum of shot types will lead to a more robust solution.

Since we have an automatic pipeline for downloading and tagging videos out of a potential “bank” of over million shots, we can also significantly increase the number of videos in the dataset – Leading to better results.

Finally, exploring the multilabel solution for the problem, particularly when dealing with a high number of classes with small differences between them, is an area that warrants further investigation.