CS231N Project Milestone: Classifying Different Dance Styles in Videos

Henry Lin
Stanford University
henryln1@stanford.edu

Christopher Vo
Stanford University
cvo9@stanford.edu

Abstract

The recent advent of deep learning in the field of computer vision has led to an increased interest in the applications of Neural Networks, with breakthroughs in many different areas such as image recognition, style transfer, and image captioning. There has also been a great deal of interest in the application of deep learning to not just static images, but also videos. In this work, we explore the applicability of several approaches to video classification, specifically in the area of dance styles. Due to the wide range of human motion covered in dance, it is an area that can be quickly be generalized to other types of motion detection and classification. In this milestone, we introduce preliminary work that we have done, preprocessing the dataset and setting up a simple CNN model. We also discuss future, more sophisticated models we plan on implementing to compare performances.

1. Introduction

Advances in deep learning have allowed the field of computer vision, particularly image classification, to progress greatly. For example, state-of-the-art models have exceeded human performance on the ImageNet challenge. Powered by breakthroughs in computational hardware, this has also allowed many applications in video processing and classification to succeed. In our project, we aim to classify different dance forms from videos. Dance is an ideal application to study the potential for using deep learning to understand motion due to the unique mix of diversity and similarity between various dances. We plan to develop and compare a more traditional CNN architecture with an experimental model combining CNN and LSTM architectures and evaluating performance on the task of classifying dance videos. Much of our inspiration and dataset comes from a paper by Castro et al [1] which attempts to classify dance videos by experimenting with various neural network architectures. We hope to build on this work by trying different models and feature representations and evaluating the effect of these changes on performance.

2. Problem Statement

2.1. Problem Definition

Specifically, we aim to break dance videos into multi-frame representations that can be processed by our two models. There are 10 dances (classes) and each model will predict which class is the most likely dance for a specific multi-frame section of a video.

2.2. Dataset

Our dataset comes from work by Castro et al from Georgia Tech [1]. In their paper on classifying dance videos, they generated a comprehensive dance dataset containing videos of 10 dynamic and visually overlapping dances. Each dance has 100 videos; each video is 10 seconds long at 30 frames per second. Some frames have an image size of $720 \times 1280 \times 3$ and some have a size of $1080 \times 1920 \times 3$. This dataset was publicly released.

2.3. Expected Results

We hope that our more traditional CNN model achieves similar performance to the frame-by-frame CNN architecture from experiments from Castro et al [1], which is based off state-of-the-art convolutional architectures. Additionally, we hope that the feature representations and intuitions behind our experimental model help capture more video and frame-specific information, leading to better performance on classifying dance videos than our traditional CNN architecture for image classification.

2.4. Evaluation

We will evaluate the performance of our models through various standard classification metrics such as macro-averaged F1-score, and class-wise precision and recall. To compare our models with the paper from Castro et al [1], we will also record test accuracy as that is the only evaluation metric reported in the paper.

3. Technical Apporach

3.1. Data Processing and Representation

We will write functions to load and process the dataset. In particular, one concern is that the size of the dataset may be very large and will not all fit into memory. Therefore, we want to develop methods to load the dataset batch by batch, only reading in the images we are training or testing on when we need them. Additionally, we want to develop methods to process the individual image frames into multi-frame representations. Our initial idea was to represent each multi-frame set as a concatenation of multiple, consecutive 3D frames (height $\times$ width $\times$ color channel) along a fourth dimension. However, we are also looking into other representations that would reduce the size of the multi-frame representation, such as a simple stacking of frames by adding (or averaging) the pixels of each frame in that set or only concatenating after the convolutional layers. Another image preprocessing method we plan on applying to our data is image downsizing. This helps reduce the size of our input data and also deals with the fact that our dataset contains images of various sizes.

3.2. Traditional CNN Architecture

The first model architecture we will attempt to implement is based on popular CNN models for image classification. In particular, our model is similar to a scaled-down version of the VGGNet architecture. Our planned CNN architecture contains three sets of two convolutional layers followed by a max pooling layer. These convolutional layers are similar to VGGNet's layers with filter size of 3 in each dimension and stride of 1 and would perform 3D convolutions instead of 2D convolutions as our input data has one additional dimension due to the concatenation of multiple frames. The max pool layer has size 2 and stride of 2. These three sets of layers are followed by a fully connected layer and softmax cross-entropy loss is used as the loss function. The Adam optimization method is used to train the model.

3.3. Experimental Architecture

We also plan to experiment with a more unique model combining CNN architecture with LSTM architecture. Specifically, we want to train a more traditional CNN model to find a feature representation for individual frames. We then want to feed these individual frames in sequence into an LSTM architecture to predict the final class. The intuition behind this architecture is that it hopefully allows our model to capture the temporal information contained by sequential frames of a video. This would potentially allow us to classify videos better than with traditional image classification techniques.

4. Preliminary Results

4.1. Preliminary Test Model

Initially, we wanted to run a simple model to test both our data processing code and model building code and troubleshoot any issues. We were able to train a simple, one convolutional layer model on a small training set of one video (300 images split into 8-frame sets) per class with a batch size of 4. This model was able to overfit on a validation set of the same size as the training set via batch validation. Batch validation basically drew 4 random image-sets from the validation set to evaluate on every iteration. After using this test model to verify that our code ran smoothly, we tried to run a larger but still scaled-down version of our planned CNN architecture with 1 set of convolutional + pooling layers and a fully connected layer. However, our model took very long to run just one iteration leading us to reevaluate our architecture and representation choices (discussed below).

4.2. Reducing Model Complexity

The computational cost of running the scaled-down version of our CNN architecture was very high, leading us to question our architecture choices. After discussing with a TA (Nish), he pointed out several methods that may help reduce the complexity of our model as well as computational time. Many of these methods we plan on applying to our project and have been discussed earlier. One major cause of high computational cost was our representation of multi-frame image sets. As we concatenated on a fourth dimension, the convolutions performed incorporated an additional dimension on the number of frames per set. This introduced many more computations and parameters for our filters. To change this, we plan on experimenting different representations of the multi-frame image sets. One idea is to stack multiple frames into one combined frame using addition or averaging. The downside to this approach is loss of temporal information. Another method would be to only concatenate after applying convolutional layers to find a representation of the initial frames. Another major cause of high computational cost was the size of our images and dataset. The high resolution images (720×1280 or 1080×1980) combined with dataset size (around 30000 total images) constrained much of memory resources when read into memory. A solution to this would be to load images by batch, only loading the full image when needed. The image size could also be downsampled to a smaller size, similar to popular image classification models such as VGGNet which takes in images of input size 224×224. One other suggestion that our TA had was that frames in a multi-frame representation do not necessarily have to be consecutive but rather 20-30 frames apart as consecutive frames in videos tend to be very similar to each other.

5. References

[1] Castro, Daniel, et al. *Lets Dance: Learning From Online Dance Videos.* Georgia Institute of Technology. 2018. https://arxiv.org/pdf/1801.07388.pdf

[2] Teney, Damien and Martial Hebert. *Learning to Extract Motion from Videos in Convolutional Neural Networks.* The Robotics Institute Carnegie Mellon University. 2016. https://arxiv.org/pdf/1601.07532.pdf

[3] YouTube 8M. *A Large and Diverse Labeled Video Dataset for Understanding Video Research.* Google Research. https://research.google.com/youtube8m/explore.html