Vision Transformers for Video Classification

#Adnan Yousaf

Abstract:

The design, training procedure, architecture, and analysis of a Vision Transformer (ViT) model utilised for video categorization are all thoroughly explored in this study. To divide videos into different classes according to their content, the model is applied to a collection of video frames that have been retrieved. The complex processes including video frame extraction, data preprocessing, model building, and training are explained in detail in the study. The solution is based on PyTorch and uses ViT as the underlying architectural foundation to take advantage of its features.

1. Introduction

Convolutional neural networks, or CNNs, have faced competition from vision transformers, or ViTs, in recent years, particularly in the field of computer vision. These models stand out because they rely on mechanisms of self-attention to accurately capture long-range relationships in images. As a result, ViTs have been used for a variety of applications, from object recognition and image classification to the more complicated field of video classification. With the use of a dataset of painstakingly extracted frames, the current research proposes a novel Vision Transformer model that has been specifically designed for the task of video categorization.

2. Model Design

2.1. Video Frame Extraction

Extraction of frames from video files is a crucial first step in the development of a successful video classification model. In order to do this, the function "extract_frames_evenly" is introduced in this paper. A video file, an output directory, and a user-defined parameter indicating the number of evenly spaced frames to be retrieved from the video source are the function's input parameters. The computer vision library OpenCV, which enables the seamless reading of the video source and the subsequent retention of extracted frames as picture files in the selected output directory, is used to carry out the frame extraction operation.

2.2. Data Preprocessing

The initial stage of data preparation is essential to the model's efficacy. The dataset that serves as the model's foundation is organised around directories, each of which represents a different class. In this dataset, video sequences are methodically processed to consistently extract frames. The length of each sequence that is created from these extracted frames can be customised. The report also explains the use of data augmentation methods to increase the diversity of the training data, such as scaling and random horizontal flipping. These rigorous preparations are followed by the incorporation of the frames into a unique dataset. In order to create batches and enable effective and parallel processing, PyTorch data loaders are used.

2.3. Vision Transformer Architecture

The architectural layout of the Vision Transformer model is its core. The following list provides a summary of this architecture's key elements:

NumFrames: This parameter, which denotes the number of frames allotted to each video, is crucial in deciding how the model behaves.

SequenceLength: The total number of frames in a single input sequence. The model's ability to capture temporal dependencies is significantly influenced by the choice of this parameter.

NumofPatches: This parameter captures the level of spatial detail the model can handle by reflecting the total number of patches in the input frames.

patch_dim: This variable represents a patch's size, which is frequently specified as a constant size (for example, 16x16 pixels).

The parameter **embed_dim**, which denotes the dimensionality of patch embeddings, significantly affects the model's ability to represent data.

Number of attention heads in the self-attention mechanism, **num_heads**. The breadth and intensity of inter-patch interactions are modulated by this parameter.

Positional embeddings are specified for each patch via this parameter, which is a crucial part of the architecture. The model is able to encode spatial information thanks to these embeddings.

transformer_layers: This portion of the architecture, which consists of a stack of TransformerEncoder layers, acts as the feature extraction framework and gives the model the ability to understand complex spatial-temporal patterns in the input frames.

The final element of the architecture, **classification_head** is a linear layer in charge of the ultimate categorization of video footage into separate categories.

2.4. Training

Effectively training the Vision Transformer model necessitates the orchestration of several key components.

The following steps are elaborated upon:

Loss Function: Selecting the right loss function is crucial. A common loss function for classification problems, the Cross-Entropy Loss, is used in the model that is being described.

Optimizer: The optimizer is Stochastic Gradient Descent (SGD). SGD is a fundamental optimisation method that is used to incrementally change model parameters.

Device selection: The training procedure requires a significant amount of computing. As a result, the model is built to run on GPUs that are readily available for rapid training. The hardware you choose has a big impact on how quickly and effectively you can train.

Epochs and Batches: The model is trained across a number of epochs, with each epoch denoting a full iteration through the training dataset. The parallelized and iterative update of model parameters is facilitated by the use of batches, each of which contains a certain number of training samples.

Label Encoding: A LabelEncoder is introduced as a crucial step in the instructional design process. The work of encoding class labels into numerical values is carried out by the LabelEncoder. This is necessary for the calculation of the loss and accuracy measures that follow.

Training Loop: Iteratively updating model parameters are done through the training loop. Both the training and validation processes are included in the loop. The validation dataset is used to evaluate the model's generalizability, whereas the training dataset is used to train the model.

Metrics Tracking: As part of the training process, crucial metrics including loss and accuracy are tracked. These metrics act as important gauges of the model's effectiveness.

3. Results Analysis

The document painstakingly tracks loss and accuracy during training and includes code for training and validation loops. The code provides information about the model's performance after training. The examination of loss and accuracy in both the training and validation phases is part of the outcomes analysis. These metrics give a definite idea of how well the model can divide videos into different classes according to their content.

4. Conclusion

In conclusion, this research introduced a Vision Transformer model created especially for the classification of videos. In-depth descriptions of the model's architecture, training procedure, and data preprocessing have been provided. This codebase offers the foundation for future improvements and investigations and can be useful in classifying video footage. Experimenting with various ViT configurations, honing data augmentation methods, and doing thorough hyperparameter tweaking are potential areas for development.

References:

[1] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In Computer Vision and Pattern Recognition (CVPR), 2017. 1, 2, 4, 6, 7, 8

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. In Proceedings of SIGGRAPH, ACM Transactions on Graphics, 2009. 2

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Computer Vision and Pattern Recognition (CVPR), 2016. 1, 4, 5