Model compression for Video Understanding models to enable Video search on resource-constrained devices

ABHIRAM VADLAPATLA
JI WOONG KIM
KUAN-RU LIOU

Abstract - Advancements in deep learning and computer vision have significantly improved video understanding and semantic search, yet challenges persist due to the high computational demands of state-of-the-art models, particularly on devices with limited resources. This study investigates the application of vision transformers and explores model compression techniques and distributed inference frameworks to address these challenges. We evaluate the efficacy of these technologies on constrained platforms through comprehensive benchmarks, aiming to enhance model interpretability and accessibility of video content without compromising performance. The project seeks to establish benchmarks for selecting efficient models and strategies that adapt to various computational limitations, paving the way for broader applicability in real-world scenarios. This research not only tests the boundaries of current technologies but also provides valuable insights into optimizing video processing techniques for resource-scarce environments.

Additional Key Words and Phrases: Video Understanding, Vision Transformers, Model Compression, Computer Vision, Timesformer, VMA

ACM Reference Format:

Abhiram Vadlapatla, Ji Woong Kim, and Kuan-Ru Liou. 2023. Model compression for Video Understanding models to enable Video search on resource-constrained devices. *ACM Trans. Graph.* 37, 4, Article 111 (November 2023), 4 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The rapid expansion of video content across various platforms has necessitated advanced methodologies for video understanding and semantic search. These technologies stand at the forefront of deep learning and computer vision research, striving to make video content more interpretable and accessible. Traditional approaches such as object segmentation and analysis of optical flow have evolved, incorporating complex neural networks like ResNet and MobileNet. These advancements have significantly improved the ability to capture and interpret the dynamic content within videos.

However, the practical application of these advanced models faces significant challenges, primarily due to the high computational demands and substantial memory requirements. These challenges are exacerbated when deploying these technologies on devices with limited computational resources, such as mobile phones and embedded systems. The variable duration and high data volume of videos further complicate this issue, often making state-of-the-art video understanding models impractical for real-world applications on such devices.

In response to these limitations, this paper explores the potential of vision transformers, a novel class of models adapted from the transformers used in natural language processing, for video understanding tasks. Vision transformers have shown promising results in capturing long-range dependencies within videos, offering an alternative to the convolutional approaches traditionally used. Alongside, we delve into model compression techniques and distributed inference frameworks as viable solutions to mitigate the computational and memory constraints of resource-scarce environments.

Our investigation aims to conduct comprehensive benchmarks to assess the efficacy of these technologies under various constraints. Through this exploration, we seek to establish a set of benchmarks that guide researchers and practitioners in selecting the most efficient models and compression strategies, thereby democratizing access to cutting-edge video understanding technologies and expanding their applicability across a broader array of devices. This paper will detail our methodologies, experiments, and findings in pursuing these objectives.

2 WHY IS IT INTERESTING?

The interest in researching video understanding and semantic search within videos using advanced deep learning techniques is multifaceted and deeply rooted in both the exponential growth of video content and its vast applications across various sectors. As the world generates more video data, the necessity for robust, efficient tools to analyze and extract meaningful insights becomes paramount. These tools have the potential to revolutionize industries such as surveillance, autonomous driving, healthcare, and content management by providing advanced capabilities for real-time activity recognition, traffic condition analysis, patient monitoring, and personalized content recommendations.

However, the challenge lies in the variable duration of videos and the significant computational demands required to process them, particularly on devices with limited resources like smartphones and embedded systems. This makes the research crucial as it addresses how to deploy sophisticated video analysis models on such devices without compromising performance. By focusing on cutting-edge technologies like vision transformers and exploring model compression techniques and distributed inference frameworks, the project not only aligns with the latest developments in AI but also tackles practical limitations head-on.

The research aims to establish benchmarks and evaluate the performance of these technologies under various constraints, providing actionable insights that can guide the design of more efficient systems. This has considerable implications for both academic research and practical applications, making the project not only academically

Authors' addresses: Abhiram Vadlapatla; Ji Woong Kim; Kuan-Ru Liou.

0730-0301/2023/11-ART111 $15.00
https://doi.org/XXXXXXX.XXXXXXX

rich but also industrially relevant. The intersection of technological innovation and real-world applicability encapsulates why this research area is of significant interest and importance.

3 RELATED WORK

3.1 TimeSformer

The TimeSformer[1] is a pioneering model that adapts the transformer architecture, originally developed for natural language processing, to handle video understanding tasks. Unlike conventional methods that typically rely on convolutional neural networks (CNNs), the TimeSformer leverages the self-attention mechanism to process video clips as integral units. This approach allows the model to effectively capture temporal dynamics across frames, thus enhancing its ability to understand complex video sequences. Introduced by Bertasius et al. in 2021, the TimeSformer decomposes the video into spatial and temporal tokens, applying separate attention mechanisms to each, which reduces computational complexity and allows for more scalable training across longer video sequences. This model has shown exceptional performance in tasks such as action recognition, making it a robust choice for deep video analysis.

3.2 VMAE (Video Masked Autoencoder)

The Video Masked Autoencoder (VMAE)[2] represents a novel approach in video processing by leveraging the principles of self-supervised learning, particularly the autoencoding technique, to enhance video understanding without the need for extensive labeled datasets. Inspired by the success of masked language models in NLP, VMAE works by masking portions of video input — typically frames or patches within frames — and then reconstructing the missing content. This process encourages the model to learn rich, representative features of video content, including temporal coherence and contextual understanding between frames. The ability of VMAE to generate high-quality video embeddings from unlabeled data makes it highly effective for a variety of applications, such as video classification, anomaly detection, and more nuanced video-based inference tasks. By reducing reliance on labeled training data, VMAE offers a cost-effective and scalable solution for video understanding, particularly in scenarios where annotated videos are scarce.

4 DATASET

The Kinetics 400 dataset[3], a robust collection used for action recognition research, comprises approximately 650,000 videos categorized into 400 distinct classes of human actions. Each video is carefully annotated with a single action class and is about 10 seconds in length. For our project, we have selected a sample of 1000 videos from the validation portion of the dataset. This selection is facilitated by a CSV file provided with the dataset, which details the category of each video.

5 METHODS

5.1 Pruning

Pruning is the model compression technique that removes parameters that are not important. It is claimed that some of the essential parameters are used to make inference instead of involving the whole parameters. Based on this concept, the pruning compression technique is proposed. There are two different pruning approaches: the first approach is Train-Time Pruning, and the other approach is Post-Training Pruning. Train-Time Pruning is done during training simultaneously. Otherwise, post-pruning is the pruning that occurs after training is completed. From our experiments, we only performed the pruning after train.

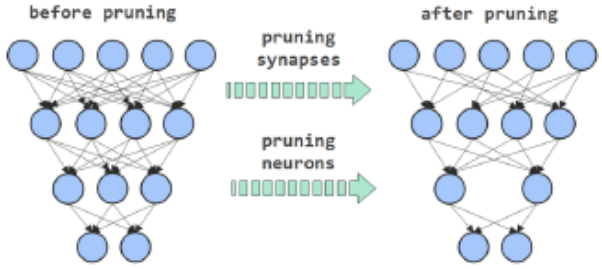

Fig. 1. Pruning [6]

5.2 Quantization

Quantization is the technique that reduces the parameters by mapping the parameters into specific sections. It reduces the larger memory format of the parameters to lower memory format. Through the mapping, the complex bit-wise numbers convert into small bit-wise numbers so that each parameter has reduced size of the parameters. Similar with pruning, there are two types of pruning exist depending on the application of the pruning, there are also pre and post quantization. Post quantization is the quantization that is performed after training and it has risk that leads to accuracy degradation. Otherwise, the pre-quantization performs weight conversion during the training.

6 IMPLEMENTATION

6.1 Profiling Models

As specified above, there are modules within the models that consume the most resources, and we aim to prune these modules while monitoring the sacrifice of accuracy. We profile the two models, and the results are in Fig. 3 and Fig. 4. To identify these modules, we analyze the modules within the models that consume the most memory. The more CPU usage indicates the more time is used for the modules, so we aim to prune the modules that use the most CPU based on CPU usage monitoring results.

6.2 Model Compression Strategy

6.2.1 Pruning. We performed two different pruning strategies: global pruning and local pruning. For unstructured pruning, we adopted global pruning, which considers entire layers and neurons. And for structured pruning, we performed local pruning, which focuses more on individual weights. Structured pruning is more likely to zero out parameters on a larger range of the whole parameters. Otherwise, unstructured pruning focuses more on a relatively small number of parameters than structured pruning.

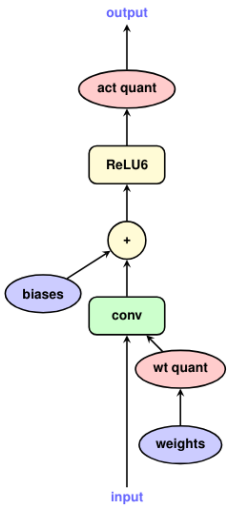

Fig. 2. Quantization[5]

Name	Self CPU %	Self CPU	CPU total %	CPU total	CPU time avg	# of Calls
model_inference	5.39%	438.652ms	100.00%	8.141s	8.141s	1
aten::matmul	0.35%	28.819ms	37.68%	3.068s	63.900ms	48
aten::bmm	33.49%	2.726s	34.78%	2.831s	58.980ms	48
aten::linear	0.02%	1.654ms	27.52%	2.240s	26.353ms	85
aten::addmm	22.27%	1.813s	27.48%	2.237s	26.315ms	85
aten::softmax	0.04%	3.402ms	12.72%	1.036s	43.150ms	24
aten::_softmax	12.72%	1.035s	12.72%	1.035s	43.144ms	24
aten::copy_	8.96%	729.259ms	8.96%	729.259ms	3.013ms	242
aten::mul	6.84%	557.190ms	6.85%	557.589ms	23.233ms	24
aten::reshape	0.02%	2.034ms	3.80%	309.421ms	879.037us	352

Self CPU time total: 8.141s

Fig. 3. Timesformer Profiling

Name	Self CPU %	Self CPU	CPU total %	CPU total	CPU time avg	# of Calls
model_inference	8.12%	80.932ms	100.00%	996.179ms	996.179ms	1
aten::linear	0.12%	1.180ms	26.79%	266.835ms	3.655ms	73
aten::addmm	24.07%	239.827ms	26.55%	264.534ms	3.624ms	73
aten::softmax	0.05%	491.000us	25.07%	249.783ms	20.815ms	12
aten::_softmax	25.02%	249.292ms	25.02%	249.292ms	20.774ms	12
aten::matmul	0.22%	2.195ms	21.49%	214.115ms	8.921ms	24
aten::bmm	20.93%	208.509ms	21.04%	209.563ms	8.732ms	24
aten::div	7.69%	76.620ms	7.71%	76.778ms	6.398ms	12
aten::gelu	5.08%	50.586ms	5.08%	50.586ms	4.215ms	12
aten::copy_	3.54%	35.291ms	3.54%	35.291ms	345.090us	102

Self CPU time total: 996.179ms

Fig. 4. VMAE Profiling

6.2.2 Quantization. We performed two different quantization strategy which are Int8 and Float16, which used to optimize neural network models by reducing the precision of numerical data. Int8 quantization converts floating-point numbers to 8-bit integers, which drastically reduces the model size and enhances computational efficiency, particularly on hardware optimized for integer operations. This method is highly beneficial for deployment in environments with limited computational resources, though it may slightly impact model accuracy due to the reduced numerical precision. On the other hand, Float16 quantization, also known as half-precision, involves reducing data from 32-bit to 16-bit floating points. This format is well-suited for modern GPUs and TPUs that support fast float calculations, offering a good balance between computational speed and accuracy retention, making it ideal for applications where higher numerical precision is necessary.

7 RESULT

7.1 Pruning

Model	Model Size	Sparsity	Accuracy	Average Inference time
Timesformer (non-pruning)	488.19 MB	0	70.0 %	9.83s
(Global Pruning) Attention layer + Dense	488.19 MB	15.95 %	67.5 %	7.98s
(Local Pruning) Attention layer	488.19 MB	6.38 %	12.5 %	6.73s
(Local Pruning) Dense	488.19 MB	4.26 %	55 %	6.725s
(Local Pruning) Attention and Dense	488.19 MB	10.64 %	7.5%	6.701s

Fig. 5. Timesformer Pruning

Model	Model Size	Sparsity	Accuracy	Inference time
VMAE (non-pruning)	346.20MB	0	75.0%	0.84s
(Global Pruning)	346.20MB	19.63%	72.5 %	0.754 s
(Local Pruning) Attention Only	346.20MB	6.01%	42.5%	0.762s
(Local Pruning) Dense Layer intermediate output	346.20MB	12.00%	25.0 %	0.826s
(Local Pruning) attention_output + Dense Layer intermediate layer qkv	346.20MB	18.02%	7.5 %	0.855s

Fig. 6. VMAE Pruning

We analyzed the models and attempted to compare the pruning results based on modules. Since both VMAE and Timesformer contain attention modules, we aimed to observe the pruning results when pruning the attention module and dense layer module. We observed the performance differences between attention layer pruning, dense layer pruning, and pruning on both attention layer and dense layer.

Through pruning both the Timesformer and VMAE, we could observe that the model size does not change. That is because the way PyTorch performs pruning is by masking the parameters to make them zero instead of removing the actual parameters. For this reason, both before pruning and after pruning have the same model size. For the inference section, as expected before pruning, the pruned model has worse accuracy than the non-pruned model. Especially, we could observe that the more pruned the model, the worse its accuracy compared to the non-pruned model. It gives us experimental results indicating that balancing efficiency and accuracy would be very significant.

For the inference time for the pruned model and the non-pruned model, we could observe that the pruned model could have faster inference time than the non-pruned model, and this occurs for both VMAE and Timesformer architectures.

For global pruning, we could observe that global pruning results in bigger sparsity than local pruning, but the accuracy of global pruning is higher than local pruning. We can infer that this happens because global pruning considers more global contexts than local pruning.

7.2 Quantization

Through the quantization for both the Timesformer and VMAE, the model size reduced to a quarter in int8 qunatization. This is because the original 32-bit floating point weights are converted into 8-bit integers. However, the model size remain the same in float16 qunatization. Some architectures automatically pad or align float16 data to 32 bits for optimal processing efficiency, which could explain why there is no substantial change in model size with float16 quantization.

For the inference time for int8 qunatization and float 16 quantization, we expected it to be faster than original model. The most common reason for slower inference post-quantization is hardware compatibility. If the hardware where the model is deployed does not natively support the quantized data formats, it may need additional operations to convert data back and forth between supported and quantized formats, adding overhead. For example, if the hardware optimally processes 32-bit floats and the model is quantized to 16-bit floats or 8-bit integers, the hardware might need extra cycles to handle these formats.

For the accuracy, in int8 qunatization Timesformer model has better performance than original one. This may because quantizing weights and activations to int8 can introduce a form of noise to the model, akin to a regularization effect. This can potentially help the model generalize better to new data by reducing overfitting. The limited precision forces the model to focus on more significant patterns rather than minute, potentially noisy details.

The decrease in accuracy for the VMAE model after int8 quantization is a common effect due to the reduced numerical precision inherent in this type of quantization.

Model	Model Size	Inference time	Accuracy
Timesformer	488.19 MB	9.83s	70%
INT8	126.36 MB	14.87s	72.5%
FP16	488.19 MB	10.4s	70.0 %

Fig. 7. Timesformer Quantization

Model	Model Size	Inference time	Accuracy
VMAE	346.20 MB	0.84s	75.0%
INT8	90.65 MB	1.417s	73.0%
FP16	346.20 MB	1.26s	75.0 %

Fig. 8. VMAE Quantization

8 CONCLUSION

We applied two model compression techniques (Pruning and Quantization) to two state of the art video models. We observe that int8 is the best quantization technique for both the models without impacting accuracy. For pruning, a global pruning performs the best with minimal reduction in accuracy.

9 FUTURE WORK

- We plan to explore the issues with fp16 quantization. Post quantization, the serializability of the model storage format is not allowing for a reduction in model size. We want to explore this further.
- Extensive testing: We plan to extensively test these compressed models further on all the categories of Kinetics 400 dataset as well as new action categories. This will help in solidifying the benchmarks.

REFERENCES

[1] Bertasius, Gedas, Heng Wang, and Lorenzo Torresani. "Is space-time attention all you need for video understanding?." ICML. Vol. 2. No. 3. 2021.
[2] Tong, Zhan, et al. "Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training." Advances in neural information processing systems 35 (2022): 10078-10093.
[3] Kay, Will, et al. "The kinetics human action video dataset." arXiv preprint arXiv:1705.06950 (2017).
[4] Wang, Jinghan, Guangyue Li, and Wenzhao Zhang. "Combine-net: an improved filter pruning algorithm." Information 12.7 (2021): 264.
[5] Jacob, Benoit, et al. "Quantization and training of neural networks for efficient integer-arithmetic-only inference." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
[6] Wang, Jinghan, Guangyue Li, and Wenzhao Zhang. "Combine-net: an improved filter pruning algorithm." Information 12, no. 7 (2021): 264.