An efficient Action Detection from First Person Vision with Attention Model

Axel Straminsky, Julio Jacobo, and M. Elena Buemi

D. de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires

Abstract. The goal of this work is to propose possible improvements on one of the latest models for Video Action Recognition based on currently existing attention mechanisms. We took a model architecture that uses 2 sub-models in paralell: one based on Optical Flow and the other based on the video itself, and proposed the following improvements: adding mixed precision in the training loop, using a Ranger optimizer instead of SGD, and expanding the Attention Mechanism. The video database used for this work was the EGTEA+ that is a action database of first person videos of daily activities.

Keywords: First Person Vision · Human Computer Interaction · Action Recognition · attention module

1 Introduction

The recognition of activities and objects in videos captured with cameras mounted on people is a topic great relevance due to the problems it can be appplied to, such as, health care assistance, non-invasive monitoring, various fields in robotics, etc.

In recent years, the use of wearable cameras has become more widespread, in large part due to the decrease in the cost of these devices and to their high resolution. The differences between the 3rd person view and the first person view can be found in [1].

This work uses a model with an architecture based on 2 sub-models (also called flows) in parallel [1], one of them to recognize movement (using Optical Flow) and the other to recognize the objects in the image. The authors of [1] have introduced in each of these flows a spatio-temporal attention mechanism called STAM (spatiotemporal attention module), and training with *gaze supervision* (that is, guided by the gaze of the camera). This results in better recognition performance since the model concentrates more on the object of interest, leaving aside the background noise.

In this work, we propose methods to achieve improvements both in terms of training time, and in the precision of the model. To improve training time, we use Mixed Precision, which allows training with a larger batch size than the one originally used in [1], thus accelerating convergence, with a very small loss in precision. On the other hand, for precision, we show different variants of the activation functions, and also changes in the number of layers of the attention mechanism. In addition, we present different optimizers that are more complex than the original implementation (vanilla SGD), with the aim of improving both convergence time and precision in the evaluation stage.

2 Methodology

In this section we present the attention based mechanisms that were modified, and the results that were obtained with these changes.

2.1 Attention mechanisms

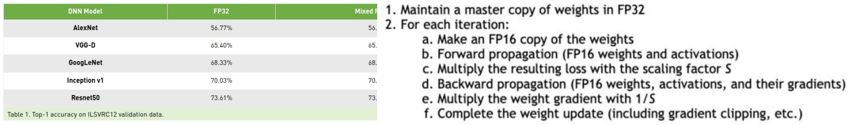
In [2], two attention mechanisms are proposed: soft and hard. Soft attention can be interpreted as a global form of attention, where attention is devoted, in varying degrees, to all features (words, image patches, pixels, or hidden states) in the input data. The local or hard attention only attends to a specific part of the input features at a time. In [3] a similar distinction was proposed for neural machine translation.

3 Proposed methods

In this work, we propose methods to achieve improvements both in terms of training time, and in the precision of the model. To improve training time, we use Mixed Precision, which allows training with a larger batch size than the one originally used in [1], thus accelerating convergence, with a very small loss in precision. On the other hand, for precision, we show different variants of the activation functions, and also changes in the number of layers of the attention mechanism. In addition, we present different optimizers that are more complex than the original implementation (vanilla SGD), with the aim of improving both convergence time and precision in the evaluation stage.

3.1 Mixed Precision

Mixed precision [4][5] is a technique that uses both 32-bit and 16-bit float registers during training, in order to speed it up and at the same time to consume less GPU memory with no loss of accuracy. In the Fig.1 it can be observed that the reported models do not suffer performance losses when using Mixed Precision, compared to not using it.



DNN Model	FP32	Mixed F
AlexNet	56.77%	56.
VGG-D	65.40%	65.
GoogLeNet	68.33%	68.
Inception v1	70.03%	70.
Resnet50	73.61%	73.

Table 1. Top-1 accuracy on ILSVRC12 validation data.

(a)

1. Maintain a master copy of weights in FP32
2. For each iteration:
    a. Make an FP16 copy of the weights
    b. Forward propagation (FP16 weights and activations)
    c. Multiply the resulting loss with the scaling factor S
    d. Backward propagation (FP16 weights, activations, and their gradients)
    e. Multiply the weight gradient with $1/S$
    f. Complete the weight update (including gradient clipping, etc.)

(b)

Fig. 1: (a) Performance variation using FP32 and Mixed Precision, (b) Mixed Precision algorithm. [5]

The general algorithm to implement mixed precision is as follows: it consists first of casting the weights of the network from FP32 to FP16, and, as a second stage, making the forward and backward pass using FP16. However, due to this, the gradient can end up being very small, so when updating the weights, this is done in FP32, previously scaling the error (loss) by a factor S.

3.2 Optimizers

One of the objectives of this work was to experiment with techniques to accelerate the convergence times of training. To achieve this, we replaced the default optimizer (SGD) with a more advanced one called Ranger, which is a combination of the LookAhead [6] method and the Rectified Adam (RAdam) [7] optimizer. RAdam is a variant of the Adam [8] optimizer, which adds a dynamic adjustment of the learning rates and thus reduces the variance of the weights in training, which improves the convergence of the model towards a local minimum that is better than the one it would have been achieved with the standard version of Adam. LookAhead is a method inspired by recent advances in the understanding of error surfaces, and consists of maintaining two sets of weights and interpolating between them, allowing one of these sets of weights to update faster, and thus explore, while letting the set of slower weights provide long-term stability. The result of this is a reduction in variance and less sensitivity to a suboptimal choice of hyperparameters during training, as well as an acceleration of convergence. The dataset used is the EGTEA Gaze+[9], which contains 28 hours of video composed of 86 unique sessions of 32 different people. There are 106 categories of actions and 10321 instances of action. Each instance of action is a video segment where the person completes an action, and all frames within that segment have the same label. The original paper trains the model for 64,000 iterations. Since this number of iterations would imply approximately 1 week of training with the current hardware, the model is retrained with 3000 iterations and its performance is evaluated again with that number. In this case and in all subsequent experiments, the models are trained with mixed precision, since this allows a considerable acceleration of the training time required to reach an acceptable solution.

In all cases, the original learning rate reduction strategy, MultiStepLR, which reduced the learning rate only in iterations 300 and 1000, is replaced by ReduceLROnPlateau, which reduces it by a factor of 0.5 each time the model does not improve its performance in the validation set for 5 iterations in a row.

3.3 Improvements done before experimentation

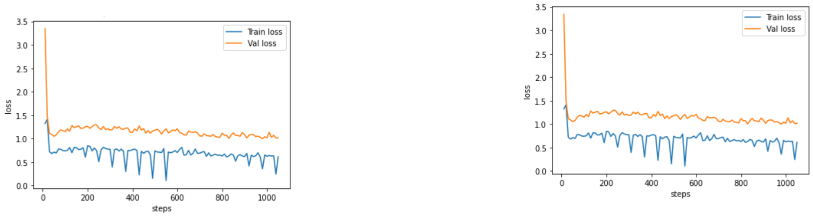
A validation set was added with its corresponding metrics (a validation set was not used in the original code) and a mixed precision was added for training.

3.4 Original Model

The results with the test set (RGB) were the following: This table shows the results for the following experiments: (1) using the original videos in RGB without the optical flow; (2) Change from SGD optimizer to Ranger and (3) Expanded Attention Mechanism with Ranger Optimizer.

	Orig.	Our Model			Lu Model
		(1)	(2)	(3)	
Accuracy	0.6365	0.633	0.6147	0.6251	0.6356
Mean class acurracy	0.5634	0.550	0.5113	0.5366	0.5634

Table 1: (1)RGB, (2)Change fron SGD optimizer to Ranger , (3)Expanded Attention Mechanism



(a) Model loss using Ranger optimizer

(b) Model loss using stacked STAM modules + Ranger

Fig. 2: Graphics

4 Conclusions and future work

With these experiments, we show that it is possible to achieve a precision similar to that obtained in [1], but using less computational resources and time. Future work will involve modifying the structure of the STAM module itself, in order to try to achieve better precision with the test set.

References

1. M. Lu, D. Liao, and Z.-N. Li, "Learning spatiotemporal attention for egocentric action recognition," Oct 2019. 1, 2, 4
2. K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," 2016. 2
3. M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015. 2
4. 2
5. Nvidia, "Mixed-precision training of deep neural networks," Oct 2017. 2
6. M. R. Zhang, J. Lucas, G. Hinton, and J. Ba, "Lookahead optimizer: k steps forward, 1 step back," 2019. 3
7. L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," 2020. 3
8. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. 3
9. Y. Li, M. Liu, and J. M. Rehg, "In the eye of the beholder: Gaze and actions in first person video," *ArXiv*, vol. abs/2006.00626, 2020. 3