View Meta-Reviews

Paper ID
518

Paper Title
RNNPool: Efficient Non-linear Pooling for RAM Constrained Inference

META-REVIEWER #1

META-REVIEW QUESTIONS

1. Please recommend a decision for this submission.
Accept (Spotlight)

3. Please provide a meta-review for this submission. Your meta-review should explain your decision to the authors. It should augment the reviews and communicate how the reviews, author response, and discussion were used to arrive at a decision. Dismissing or ignoring a review is not acceptable unless you have a good reason for doing so. If you want to make a decision that is not clearly supported by the reviews, perhaps because the reviewers did not come to a consensus, please justify your decision appropriately, including, but not limited to, reading the submission in full and writing a detailed meta-review that explains your decision.
This paper presents a promising technique to downsample for rapidly downsampling spatial dimensions in CNNs. It significantly reduces memory requirements and can be expected to not have much runtime overhead. The experiments are well done and the reviewers agree the paper should be accepted.

We thank the reviewers for their helpful comments. The code and models will be open-sourced.

Reviewer 1

Inference speed test: Along with peak RAM, we report inference FLOPs for all the models. We see a consistent reduction in FLOPs per inference using RNNPool. Resource-constrained devices like microcontrollers, the key focus of our work, do not have parallelism – so the reduction in FLOPs translates reasonably accurately to faster inference; we will add inference numbers on Cortex M-class devices in the manuscript. On hardware with parallelism, the application of RNNPool operator to multiple patches in an activation map can be fully parallelized. For face detection on 320×240 image, the RNNPool layer has about 4800 patches that can be executed in parallel – sufficient for many cloud hardware.

Inappropriate comparison of experiments: The main goal of Table 1 (that includes DenseNet121 on ImageNet-10) is to show that RNNPool works well with a variety of architectures. While we agree that DenseNet121 result might be an artifact of a big model applied to small ImageNet-10 dataset, we observe a similar trend for smaller models like MobileNet also (that are in fact more critical for the resource-constrained settings that we study). We observe a similar trend for ImageNet-1K as well (Table 3) but focus mainly on MobileNets due to resource-constrained devices setting. Table 2 also shows that pooling/downsampling operators significantly affect accuracy in both DenseNet121 and MobileNetV2. Finally, the rationale behind ImageNet-10 can be found in Appendix A.1.

Reviewer 2

FLOPs of ReNet: In Table 2 second block, we compare the accuracy of various downsampling methods while keeping the working RAM same for each. Therefore, ReNet and RNNPool were placed in the same position and have the same shape of input-output feature maps. So when the output map from RNNPool has $4h_2$ channels, the hidden state size of the RNNs in ReNet was taken to be $2h_2$ to ensure an output map of $4h_2$ channels. And even then ReNet's accuracy is significantly worse than RNNPool on multiple tasks. Furthermore, in ReNet each input patch is flattened and passed as a single input to an RNN which leads to a huge matrix multiplication operation. Finally, ReNet authors suggest using GRU or LSTM as the RNN unit; we use GRU as it is more efficient. RNNPool uses FastGRNN as the RNN unit which also contributes to the reduction in FLOPs.

Similarity to ReNet: We respectfully disagree with the reviewer. The global and local contexts do not depend on overlapping or non-overlapping patches but are captured by the constituent operations in a layer. As mentioned in L106-123, ReNet applies each horizontal and vertical RNN sweep on the *entire* image in *one pass* and the resulting hidden states provide the output feature map. Even if ReNet uses overlapping patches, it will still apply both RNNs in *one pass* over the image. In contrast, RNNPool is applied *patchwise* (it is never applied to the entire image in one pass) and only the *final* states from each RNN run are used for output feature map. So, the intuition and goal of the two methods seem quite different. We do present a comparison with ReNet (seen as a downsampling operator) in Table 2 on two different tasks and with three different base models across 2 datasets.

Comparison to DNN compression methods: RNNPool is a complementary technique to model compression, i.e., we can use model compression (e.g. pruning, quantization) on RNNPool based models as well to get additional reduction in working RAM, model size & inference FLOPs. To illustrate this, we created the RNNPool-Face-Quant (Table 4) model using a standard quantization based compression method that maintains accuracy with about $4\times$ compression.

Reviewer 3

Quantization methods: See response to R2; we will add references to the manuscript. As mentioned above, RNNPool is complementary to and supports quantization (and other standard model compression techniques).

Training from scratch: RNNPool is a new building block (pooling operator with parameters) for CNNs. As is the case with any architecture involving a new building block, training the model end-to-end provides the best performance. XNOR-Nets are also trained from scratch because of the formulation and could be coupled with RNNPool based models if needed during end-to-end training.

Training of RNNPool & underlying difficulty: The instances of RNNPool used in our models typically require a sequence length of 8 for the RNNs, and thus are reasonably stable to train. We use the same hyperparameters as the base CNN models to train the RNNPool variants. We do observe that the RNNPool based variants require more epochs to train than the base CNN models. But, overall, the training complexity of RNNPool based models is similar to the base CNN models. We will add training accuracy vs epochs plot in the appendix.

Reviewer 4

Thank you for the positive feedback. The pointer to SqueezeNAS and making a stronger case for RNNPool with the argument is helpful. We will include them in the next revision and fix the citation to "Seeing AI".

Deployment on a microcontroller: We will open-source the code to deploy RNNPool based models on ARM Cortex M-class microcontrollers, and add some of the details in the appendix.

View Reviews

Paper ID
518

Paper Title
RNNPool: Efficient Non-linear Pooling for RAM Constrained Inference

Reviewer #1

Questions

1. Summary and contributions: Briefly summarize the paper and its contributions.
The authors proposed a RNNPooling block downsampling size of activation maps over patches up to 64*64 decreasing memory and computation cost. With parameters of RNNs, it could retain information against traditional pooling operator and play a role in incremental model capacity.

2. Strengths: Describe the strengths of the work. Typical criteria include: soundness of the claims (theoretical grounding, empirical evaluation), significance and novelty of the contribution, and relevance to the NeurIPS community.
It's meticulous of the experiments of designing datasets of small 8bit monochrome images to illustrate fitting ability of RNNPooling and formal derivation of memory cost. Additionally, the authors evaluated their approach by using Visual Wake Words dataset. By inserting RNNPooling at the beginning and the end of network the compiled model achieved an accuracy score within 0.6% of the baseline but with far smaller memory cost (250->33.KB).

3. Weaknesses: Explain the limitations of this work along the same axes as above.
- Absence of inference speed test. The title, Efficient Non-linear Pooling for RAM Constrained Inference, should involve not only detailed calculation of RAM cost but also speed change of inference, especially the introduction of RNN which would usually have large negative impact upon the parallelism of computation. unfortunately, the authors seem to ignore this significant aspect of efficiency.

- Inappropriate comparison of experiments. The performance of a model largely depends on adaption of model capacity and task difficulty. Big model usually is not in an advantageous position against smaller one in small dataset. The experiment, densenet121 on imagenet10, seems like to illustrate efficiency with big model in small dataset. it's still in the air that whether tremendous saving of memory and computation cost without obvious accuracy decline by replacement of C1, P1, D1 and T1 in densenet121 to RNNPooling is as consequence of superiority of RNNPooling or just regular results of an easier task. it is more convincing that the authors do the experiment on ImageNet1K.

---------- After rebuttal ----------

the author's feedback very well addressed my questions on inference speed. For the performance comparison I still think it would be more complete to show the results of all the architectures in various settings on each dataset. Overall, this paper proposes a simple idea but might have good impact in many application domains, I thus recommend acceptance.

4. Correctness: Are the claims and method correct? Is the empirical methodology correct?
The paper sounds technically correct.

5. Clarity: Is the paper well written?
The writing is good and the structure is well organized.

6. Relation to prior work: Is it clearly discussed how this work differs from previous contributions?
Yes

7. Reproducibility: Are there enough details to reproduce the major results of this work?

Yes

9. Please provide an "overall score" for this submission.
6: Marginally above the acceptance threshold.

10. Please provide a "confidence score" for your assessment of this submission.
4: You are confident in your assessment, but not absolutely certain. It is unlikely, but not impossible, that you did not understand some parts of the submission or that you are unfamiliar with some pieces of related work.

11. Have the authors adequately addressed the broader impact of their work, including potential negative ethical and societal implications of their work?
Yes

Reviewer #2

Questions

1. Summary and contributions: Briefly summarize the paper and its contributions.
An efficient and effective RNN-based pooling operator (termed RNNPool) is proposed for rapidly downsampling activation map sizes, which is more effective than traditional average or max pooling, and also more effective than ReNet. RNNPool can be simply embedded into most existing CNN based architectures by replacing several stacks of convolutions and pooling layers or one pooling layer. Extensive experiments show the RNNPool operator can significantly decrease computational complexity and peak memory usage for inference while retaining comparable accuracy for different vision tasks.

2. Strengths: Describe the strengths of the work. Typical criteria include: soundness of the claims (theoretical grounding, empirical evaluation), significance and novelty of the contribution, and relevance to the NeurIPS community.
1. Good presentation and well written with implementation details of the algorithm.
2. Simple yet effective idea by replacing the traditional pooling methods+convolutions with RNNPool operator.
3. The insightful and thorough experiments, e.g. comprehensive comparison with other operators, ablation study and expansive application for different tasks

3. Weaknesses: Explain the limitations of this work along the same axes as above.
1. RNNPool is very similar to ReNet, which also replaces convolution+pooling layer with 4 RNNs by sweeping horizontally and vertically in both directions. In my understanding, the differences between these two operators are the number of RNNs, the order of sweeping, and the intermediate hidden states. The authors claim ReNet cannot capture local features by flattening non-overlapping patches. I cannot agree with it. ReNet is also flexible to adjust the non-overlapping patches into overlapping patterns and extract strong local features. If not, more experiments should be added to support your claims (line 110).
2. Give more texts to describe the main idea of Figure 1 for better understanding.
3. Several typos: ``P_{xx}^{h}" in line 13 of Algorithm 1 -> P_{xx}^{r}; ``showsn" in line 245 ->shown
4. Need more comparison to other DNN compression methods, e.g. pruning methods [21, Ref-1-3]

[Ref-1] Nisp: Pruning networks using neuron importance score propagation, in CVPR, 2018
[Ref-2] Towards Optimal Structured CNN Pruning via Generative Adversarial Learning, in CVPR, 2019
[Ref-3] Gate Decorator: Global Filter Pruning Method for Accelerating Deep Convolutional Neural Networks, in NeurIPS, 2019

4. Correctness: Are the claims and method correct? Is the empirical methodology correct?
At the same input and RNNs hyperparameters, ReNet generates hidden state size of h_1 + h_2, which should be smaller than 4*h_2 with 2 shared RNNs in RNNPool. Why the FLOPs of ReNet is larger than RNNPoolLayer in Table 2?

I appreciate the authors' feedback.

The authors' feedback well answer my concerns. I stand by my recommendation to accept the paper.

5. Clarity: Is the paper well written?
Yes

6. Relation to prior work: Is it clearly discussed how this work differs from previous contributions?
Yes

7. Reproducibility: Are there enough details to reproduce the major results of this work?
Yes

8. Additional feedback, comments, suggestions for improvement and questions for the authors:
Refer to weaknesses for more improvements.

9. Please provide an "overall score" for this submission.
7: A good submission; accept.

10. Please provide a "confidence score" for your assessment of this submission.
2: You are willing to defend your assessment, but it is quite likely that you did not understand central parts of the submission or that you are unfamiliar with some pieces of related work. Math/other details were not carefully checked.

11. Have the authors adequately addressed the broader impact of their work, including potential negative ethical and societal implications of their work?
Yes

Reviewer #3

Questions

1. Summary and contributions: Briefly summarize the paper and its contributions.
This paper proposes to reduce the memory and computation of neural networks at inference time through the use of an RNN-based pooling scheme that downsamples activations more aggressively as compared to conventional pooling mechanisms.

2. Strengths: Describe the strengths of the work. Typical criteria include: soundness of the claims (theoretical grounding, empirical evaluation), significance and novelty of the contribution, and relevance to the NeurIPS community.
- It is a relatively novel approach to pooling and the consequent reduction in memory and computational requirements.

- Sufficient empirical proof has been provided to show that the method is effective in practice without compromising on the accuracy of the networks.

3. Weaknesses: Explain the limitations of this work along the same axes as above.
- The networks need to be retrained after changing the pooling units to RNNpool

- It is not explained if it is more difficult to train the networks when using such pooling, in the light of the fact that RNN's are harder to train than feed-forward convolutional networks.

- Comparisons have not been made to methods of quantization, which also reduce memory and computational requirements without compromising on accuracy. In this regard there is only one paper [37] cited, which is just related to mixed precision networks. There are several others that deal with integers or just binary network, which could be cited:

[Ref-1] "XNOR-net: Imagenet classification using binary convolutional neural networks." ECCV 2016
[Ref-2] "Data-Free Quantization through Weight Equalization and Bias Correction" ICCV 2019

4. Correctness: Are the claims and method correct? Is the empirical methodology correct?

Yes, the claims and method seem correct.

5. Clarity: Is the paper well written?
The paper is mostly well-written but clarity can still be improved. In particular, Section 3.1 can be better written to explain how training of the networks is done with RNNpool units. It is not obvious what the RNNpool layers train on. One has to assume that the input to the RNNpool layers is the rows and columns of the activation matrices on which the pooling is applied for each example.

6. Relation to prior work: Is it clearly discussed how this work differs from previous contributions?
Yes, the distinction has been made clear.

7. Reproducibility: Are there enough details to reproduce the major results of this work?
Yes

8. Additional feedback, comments, suggestions for improvement and questions for the authors:
- The main shortcoming of the approach is that the network has to be retrained from scratch when using the RNNpooling units. In some sense the authors are suggesting a new architecture to replace a bulkier one. Given this, the benefits of lower computational and memory requirements quickly pale in comparison to methods that can either quantize networks to binary values [Ref-1 and the likes] or that can quantize the networks without requiring retraining [Ref-2]

- Otherwise, the approach seems novel and effective in reducing the memory and computational requirements at inference time.

- An aspect the authors do not touch upon is the difficulty of training the network after plugging in the RNNpool units. This aspect can not be ignored since RNN's are known to be harder to train as compared to feed-forward networks.

-- post-rebuttal --

According to the authors, in effect, having replaced the pooling modules, the model is a new one and needs to be retrained from scratch. This means that most of the benefits of the fully trained models are at risk since we may not obtain the same performance again. It is likely training becomes more difficult or takes longer after replacing the modules. This needs to be stated and demonstrated if possible. Given the response of the authors, I will stick to my previous rating of the paper

9. Please provide an "overall score" for this submission.
6: Marginally above the acceptance threshold.

10. Please provide a "confidence score" for your assessment of this submission.
4: You are confident in your assessment, but not absolutely certain. It is unlikely, but not impossible, that you did not understand some parts of the submission or that you are unfamiliar with some pieces of related work.

11. Have the authors adequately addressed the broader impact of their work, including potential negative ethical and societal implications of their work?
Yes

Reviewer #4

Questions

1. Summary and contributions: Briefly summarize the paper and its contributions.
In neural networks for image classification, it is typical to gradually downsample an image from full resolution to an output vector of size 1x1xN, where N is the number of categories to classify. Downsampling also occurs in neural networks for other computer vision tasks such as semantic segmentation and object detection. When downsampling, it is typical to first increase the number of channels, and then downsample. This requires more working memory than is required for other layers in the network. To address this, instead of doing the "increase channels, then downsample" approach, RNNPool instead uses an RNN to downsample.

2. Strengths: Describe the strengths of the work. Typical criteria include: soundness of the claims (theoretical grounding, empirical evaluation), significance and novelty of the contribution, and relevance to the NeurIPS community.
In my opinion, the authors are correct that the details of how you downsample inside your neural net are a common reason for running out of memory when running computer vision inference on a microcontroller. This is practical problem that I have personally dealt with in my research, and the authors propose a compelling solution. Note that running out of memory on a microcontroller is much more painful than on a server, because there is often no ""swap"" memory, so the device freezes up.

The improvements enabled by RNNPool are clearly conveyed in experiments on the ImageNet-1k and Wider Face datasets. Specifically, introducing the RNNPool operation into the MobileNet v2 network significantly improves peak memory usage (savings of up to 10x) and actually improves accuracy too.

--- Update after reading the rebuttal ---
I continue to recommend this paper for acceptance.

3. Weaknesses: Explain the limitations of this work along the same axes as above.
It would be interesting to see the network run on a microcontroller. But, I know this would require a lot more engineering. As it is, I am already convinced that the work presented in this paper does make a real advance for TinyML on a microcontroller.

4. Correctness: Are the claims and method correct? Is the empirical methodology correct?
Looks good.

5. Clarity: Is the paper well written?
Yes.

6. Relation to prior work: Is it clearly discussed how this work differs from previous contributions?
One thing a reader might ask is "couldn't you simply use vanilla max-pooling to downsample, but just wait until after downsampling to increase the number of channels?" One way to answer this question is to point to SqueezeNAS [1], which searches for efficient neural architectures for semantic segmentation. Interestingly, even on a constrained computational budget, the architecture-search "prefers" to put a lot of computation just before downsampling (see Figures 8 and 9 of the SqueezeNAS paper). So, RNNPool may be one of the only ways to preserve accuracy without inflating the number of channels (and the amount of memory) just before downsampling.

[1] @inproceedings{2019_SqueezeNAS,
author = {Albert Shaw and Daniel Hunter and Forrest Iandola and Sammy Sidhu},
title = {{SqueezeNAS}: Fast neural architecture search for faster semantic segmentation},
booktitle = {ICCV Neural Architects Workshop},
year = {2019}
}

7. Reproducibility: Are there enough details to reproduce the major results of this work?
Yes

8. Additional feedback, comments, suggestions for improvement and questions for the authors:
In the broader impact section, there is a hyperlink to "Seeing AI." I think it would be better to cite this instead of using a hyperlink.

9. Please provide an "overall score" for this submission.
9: Top 15% of accepted NeurIPS papers. An excellent submission; a strong accept.

10. Please provide a "confidence score" for your assessment of this submission.
4: You are confident in your assessment, but not absolutely certain. It is unlikely, but not impossible, that you did not understand some parts of the submission or that you are unfamiliar with some pieces of related work.

11. Have the authors adequately addressed the broader impact of their work, including potential negative ethical and societal implications of their work?
Yes