ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

Pruning Weightless Neural Networks

Zachary Susskind[1], Alan T. L. Bacellar[2], Aman Arora[1], Luis A. Q. Villon[2], Renan Mendanha[2], Leandro S. de Araújo[3], Diego L. C. Dutra[2], Priscila M. V. Lima[2], Felipe M. G. França[2,4], Igor D. S. Miranda[5], Mauricio Breternitz Jr.[6], and Lizy K. John[1] *

1- UT Austin, Austin, USA, 2- UFRJ, Rio de Janeiro, Brazil,
3- UFF, Niterói, Brazil, 4- IT-Porto, Portugal
5- UFRB, Cruz das Almas, Brazil, 6- ISCTE, Lisbon, Portugal,

Abstract. Weightless neural networks (WNNs) are a type of machine learning model which perform prediction using lookup tables (LUTs) instead of arithmetic operations. Recent advancements in WNNs have reduced model sizes and improved accuracies, reducing the gap in accuracy with deep neural networks (DNNs). Modern DNNs leverage “pruning” techniques to reduce model size, but this has not previously been explored for WNNs. We propose a WNN pruning strategy based on identifying and culling the LUTs which contribute least to overall model accuracy. We demonstrate an average 40% reduction in model size with at most 1% reduction in accuracy.

1 Introduction

Weightless neural networks (WNNs) are a class of neural model which use lookup-based neurons called RAM nodes to perform computation. Inputs to a RAM node are binary values. These inputs are concatenated to form an address, and the node outputs the binary value stored at that location in its internal lookup table [1]. The key strength of WNNs is the ability of nodes to learn non-linear functions of their inputs, which is not possible for individual neurons in a conventional deep neural network (DNN). In principle, this allows for accurate models with very few RAM nodes.

WNNs have historically suffered from impractical memory requirements. The size of a RAM node grows exponentially with its number of inputs, meaning it is usually impossible to provide all inputs to a single node. The conventional solution to this is to instead partition the inputs between many smaller RAM nodes. This makes it feasible to train WNNs for non-trivial applications, but still requires a considerable amount of memory for an accurate model. More recent work [2] recognized that the contents of these RAM nodes are highly sparse, and replaced their lookup tables with hash-based Bloom filters. While this technique reduces the size of the RAM nodes, it does not reduce their number, and the need to compute hash functions adds a new source of computational overhead.

*This research was supported in part by (a) Semiconductor Research Corporation (SRC) Task 3015.001/3016.001; (b) CAPES - Brazil - Finance Code 001; (c) CNPq - Brazil; (d) Fundação para a Ciência e a Tecnologia, I.P. (FCT): ISTAR Projects: UIDB/04466/2020 and UIDP/04466/2020, and; (e) FCT/COMPETE/FEDER, FCT/CMU IT Project FLOYD: POCI-01-0247-FEDER-045912. Any opinions, findings, conclusions or recommendations are those of the authors and not of the funding agencies.

In DNNs, pruning techniques are used to eliminate unimportant weights and connections. This process can greatly decrease the size and complexity of models with little to no penalty to accuracy [3]. In this paper, we propose a pruning mechanism for WNNs based on identifying and eliminating the RAM nodes which contribute the least to overall model accuracy. To our knowledge, this represents the first effort to prune WNNs. We report results on the MNIST, FashionMNIST, Letter, and Satimage datasets, and find that a 24-70% model size reduction is possible with $\leq 1\%$ accuracy penalty and a 49-90% reduction with $\leq 3\%$ penalty.

2 Background

WiSARD (Wilkie, Stoneham and Aleksander's Recognition Device) [4] was the first WNN to achieve commercial success, and is the baseline many subsequent models are built on. WiSARD is intended for classification tasks. A WiSARD model contains submodels, referred to as *discriminators*, for each output class, which are in turn composed of RAM nodes. Inputs to the RAM nodes are assigned using a random mapping, which is shared between discriminators. During inference, the outputs of the RAM nodes in each discriminator are summed to produce a *response* value, and the index of the discriminator with the strongest response is taken as the predicted class.

Bloom Filters: A Bloom filter is a hash-based data structure for approximate set membership composed of a lookup table L and independent hash functions $\{h_1, \dots, h_k\}$. To test whether a value x is in the Bloom filter, the `AND` reduction of entries at hashed addresses $\bigwedge\{L[h_1(x)], \dots, L[h_k(x)]\}$ is computed. Bloom Filters were previously used to replace LUTs in RAM nodes in Bloom WiSARD [2], and greatly reduced model size with a minimal impact to accuracy. An example of the Bloom WiSARD model and its inference behavior is shown in Figure 1.

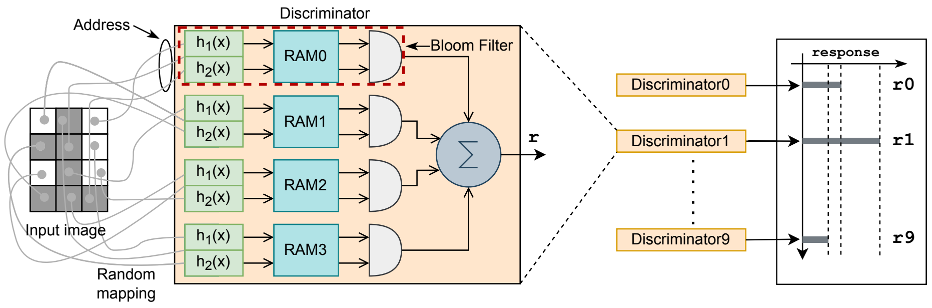

Fig. 1: A depiction of the Bloom WiSARD [2] WNN model. In this example, the input image contains "1", and the corresponding discriminator produces the strongest response.

Thermometer Encoding: Traditionally, inputs to WiSARD are binarized by comparing them against their mean value in the training data. A thermometer encoding is a multi-bit unary encoding which instead compares values against a series of increasing thresholds [5]. Thermometer encoding gives increased resolution and accuracy at the cost of model size.

Feedback-based Training: WiSARD is conventionally trained using a one-shot learning rule. However, we need to incorporate feedback to enable fine-tuning after pruning. The key challenge is that Bloom filter entries are binary, which makes them incompatible with traditional gradient-based methods. We use a training rule based on the one described in [6] for binary neural networks. We treat filter entries as floating point values between -1 and 1 and binarize them using the unit step function. During backpropagation, gradients bypass this unit step, and are therefore not canceled. This technique, known as the straight-through estimator, has not previously been used to train WNNs.

Ensembles: An ensemble combines multiple weak classifiers in order to create a single stronger classifier. Ensembles are frequently used in contexts such as gradient boosting [7]. We have found that ensembles of very small WiSARD models are both smaller and more accurate than singular larger WiSARD models when trained using the feedback-based technique.

3 Pruning Methodology

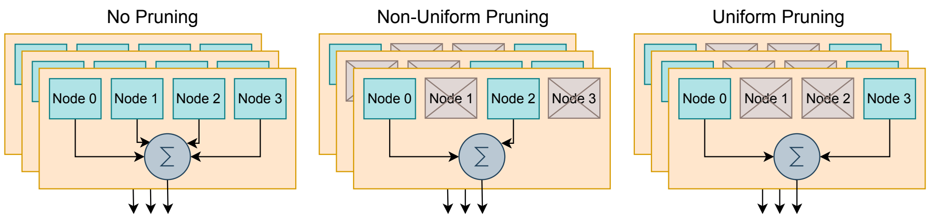

Fig. 2: Weightless models, showing (a) No pruning, (b) Non-uniform pruning, where different RAM nodes are pruned in different discriminators, and (c) Uniform pruning, where the same RAM nodes are pruned in all discriminators

We implement and train WiSARD models using all enhancements described in Section 2, including Bloom filters, thermometer encoding, model ensembles, and the feedback-based learning rule. Pruning is performed as a post-training step. As shown in Figure 2, we consider two different variants of pruning: a non-uniform technique, where pruned nodes in different discriminators may be at different indices, and a uniform technique, where the indices of pruned RAM nodes are shared between discriminators.

The first step in the pruning process is to identify how much each RAM node contributes to producing the correct output. For node j in discriminator i, the response score s_{ij} is given by:

$$s_{ij} = \sum_{d \in D} N_{ij}[d](M\delta(l[d] - i) - 1)$$

Here, D is a dataset with M classes, d is a sample within D, $N_{ij}[d]$ is the output of the RAM node, $l[d]$ is the correct label for d, and δ is the Kronecker

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

delta function.

Once scores have been computed for all nodes in the model using the training data, a fixed fraction of the RAM nodes with the lowest scores are culled, replacing them with a constant 0 value. This has the side effect of lowering the maximum responses of the discriminators. Furthermore, the impacts on the responses of different discriminators may be unequal. For instance, a node which always outputs 0 and a node which always outputs 1 will have identical scores (assuming an equal number of samples in each class), but clearly have different contributions towards their discriminators' responses. To compensate for this, we learn a set of biases, which are added to the outputs of each discriminator.

Lastly, we perform a fine-tuning training pass using the pruned model and the feedback-based learning rule. In practice, we find that this can recover a significant portion of the accuracy that was lost from pruning, particularly when a higher proportion of nodes are culled.

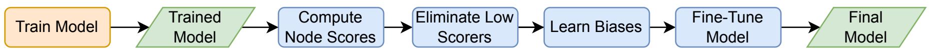

Fig. 3: Summary of the pruning process for a trained model

The pruning process as a whole is summarized in Figure 3. Since we replace lookup tables with Bloom filters in our model, we also need to consider the impact of pruning on the number of hash function computations required. All discriminators in a model have the same input mapping, so by reusing the same set of hash functions between Bloom filters, we only need to compute hashes for the first discriminator, and can reuse them elsewhere. However, it is possible that the filter at some index j may be pruned in the first discriminator, but not in some other discriminator in the model. In this case, it is still necessary to compute the hash functions for this filter. To avoid this, the uniform pruning technique (shown in Figure 2) culls filters at the same indices in each discriminator. The filters culled are those which have the lowest maximum response score in any discriminator.

4 Experimental Results

We use the MNIST [8] dataset as an illustrative example to show the impact of pruning in WNNs. Our baseline weightless model is an ensemble of 6 submodels with a total parameter size of 373 KiB and test accuracy of 98.49%. We prune this model using both the uniform and non-uniform methods in increments of 10% up to 90%, and then increments of 2% up to 98%. The results of this pruning sweep are shown in Figure 4.

Up to a 70% pruning ratio, both methods achieve approximately equal post-pruning accuracies, with <1% degradation. Past this point, both methods begin to substantially degrade accuracy, with uniform pruning having a far larger impact. The implication of this result is that there is a relatively small set of

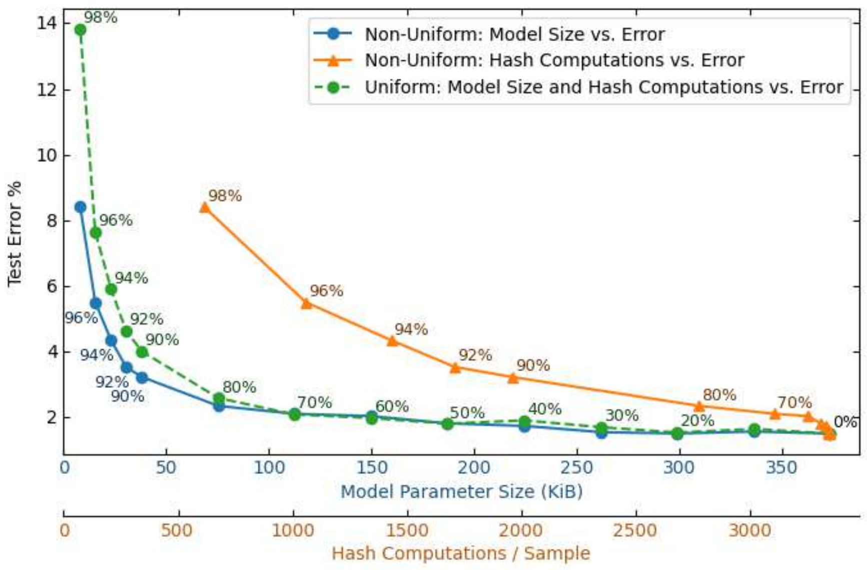

Fig. 4: Results of pruning a weightless model for the MNIST dataset by 0-98%. When using uniform pruning, model size and hash computations follow identical curves.

RAM nodes which are critical for model accuracy, but this set is not identical for all discriminators.

The number of hash function computations required for non-uniformly pruned models is not significantly improved for pruning ratios below 70%, and even at a 98% pruning ratio the reduction is only 82%. On the other hand, the number of hash functions required for uniformly-pruned models is directly tied to the pruning ratio, as expected.

Dataset	Bloom WiSARD [2]		Our Unpruned		Conservative Pruning		Aggressive Pruning	
	Size (KiB)	Accuracy	Size (KiB)	Accuracy	Size (KiB)	Accuracy	Size (KiB)	Accuracy
MNIST	819	91.5%	373	98.49%	112	97.91%	38.1	95.99%
FashionMNIST	-	-	2786	89.20%	1533	88.62%	412	86.61%
Letter	91.3	84.8%	30.9	93.13%	23.6	92.27%	15.8	90.28%
Satimage	12.7	85.1%	5.53	88.30%	3.66	87.30%	1.17	85.35%

Table 1: Pruning results for studied datasets. We identify pruning ratios with conservative ($\leq 1\%$) and aggressive ($\leq 3\%$) accuracy penalties.

We also investigated uniform pruning with the FashionMNIST [9], Letter [10], and Satimage [11] datasets. Results are shown in Table 1. We do not present non-uniform results for these datasets, since, as our results on MNIST show, it does not provide superior accuracy except at very high ratios, and the overhead in hash computations is significant. MNIST, Letter, and Satimage were used in Bloom WiSARD [2]; our baseline (unpruned) models are both smaller and more accurate. FashionMNIST is an image classification dataset which is

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

structurally similar to MNIST (28x28 greyscale images) but considerably more challenging [9]. Uniform pruning achieves an average (geometric mean) pruning ratio of 40% with $\leq 1\%$ decreased accuracy and 74% with a $\leq 3\%$ decrease.

5 Conclusion

In this work, we propose a pruning technique for weightless neural networks. Our models are an extension of memory-efficient prior work [2], which reduced the size of the RAM nodes in a model, but not their number. Pruning entire RAM nodes allows us to further reduce model sizes at a reasonable impact to accuracy. Our experiments across four datasets show that we can reduce model size by an average of 40% with at most a 1% reduction in accuracy. Potential avenues for future work include reducing the sizes of the remaining Bloom filters after pruning, eliminating entire submodels from within an ensemble (a method proposed but not explored in [12]), and exploring the robustness of pruned models to random errors. This work is a further step toward enabling WNNs in memory-constrained environments such as embedded devices.

References

[1] Igor Aleksander, Massimo De Gregorio, Felipe França, Priscila Lima, and Helen Morton. A brief introduction to weightless neural systems. In *17th European Symposium on Artificial Neural Networks (ESANN)*, pages 299–305, 04 2009.

[2] Leandro Santiago, Leticia Verona, Fabio Rangel, Fabricio Firmino, Daniel S Menasché, Wouter Caarls, Mauricio Breternitz Jr, Sandip Kundu, Priscila MV Lima, and Felipe MG França. Weightless neural networks as memory segmented bloom filters. *Neurocomputing*, 416:292–304, 2020.

[3] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? In I. Dhillon, D. Papailiopoulos, and V. Sze, editors, *Proceedings of Machine Learning and Systems*, volume 2, pages 129–146, 2020.

[4] I. Aleksander, W.V. Thomas, and P.A. Bowden. WISARD·a radical step forward in image recognition. *Sensor Review*, 4(3):120–124, 1984.

[5] Andressa Kappaun, Karine Camargo, Fabio Rangel, Fabrício Firmino, Priscila Machado Vieira Lima, and Jonice Oliveira. Evaluating binary encoding techniques for wisard. In *2016 5th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 103–108, 2016.

[6] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1, 2016.

[7] Thomas G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[8] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010.

[9] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[10] David J. Slate. Letter recognition data set. UCI Machine Learning Repository.

[11] Ashwin Srinivasan. Statlog (landsat satellite) data set. UCI Machine Learning Repository.

[12] Leopoldo A.D. Lusquino Filho, Luiz F.R. Oliveira, Aluizio Lima Filho, Gabriel P. Guarisa, Lucca M. Felix, Priscila M.V. Lima, and Felipe M.G. França. Extending the weightless wisard classifier for regression. *Neurocomputing*, 416:280–291, 2020.