A CONTEXTUAL DISCRETIZATION FRAMEWORK FOR COMPRESSING RECURRENT NEURAL NETWORKS

Aidan Clark *
UC Berkeley
aidanbclark@berkeley.edu

Vinay Uday Prabhu & John Whaley
UnifyID Inc
{vinay,john}@unify.id

ABSTRACT

In this paper, we address the issue of training Recurrent Neural Networks with binary weights and introduce a novel Contextualized Discretization (CD) framework and showcase its effectiveness across multiple RNN architectures and two disparate tasks. We also propose a modified GRU architecture that allows harnessing the CD method and reclaim the exclusive usage of weights in $\{-1, 1\}$, which in turn reduces the number of power-two bit multiplications from $O(n^3)$ to $O(n^2)$.

1 INTRODUCTION

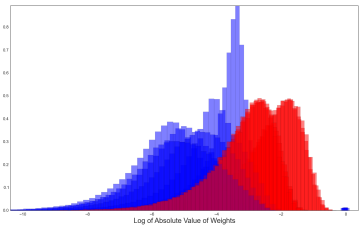
Courbariaux et al. (2015) introduced *BinaryConnect*: a feed-forward neural network with weights constrained to lie in the set $\mathcal{B} = \{-1, 1\}$, and showcased that this network could potentially result in a speed-up by a factor of 3 at training time while also reducing the memory requirement by a factor of at least 16. Since this work, efforts such as BinaryNets (Courbariaux et al. (2016)), XnorNets (Rastegari et al., 2016) and Ternary Weight Networks (Li et al., 2016) have extended these results to feed-forward and Convolutional Neural Networks (CNNs) and have achieved near state-of-the-art results on almost of all the popular benchmarking datasets. While CNNs have received a lot of attention in the general DNN-compression literature, many of the ideas developed remain largely untested on Recurrent Neural Networks (RNNs). Only recently, works such as Ott et al. (2016); Hou et al. (2016) have focused on RNN-compression and even though they show that quantization can be applied to recurrent networks, both papers showcased that the loss in performance encountered by using compressed RNNs with 1-bit weights was prohibitively large.
In this paper, we propose a weight discretization strategy, which we term as *Contextual Discretization*, which results in RNNs with 1-bit weights that can demonstrably learn effectively. We test our method with several models across 2 disparate tasks and conclude by showing that a small alteration to the recurrent networks is equivalent to Contextual Discretization and allows us to maintain the original $\{-1, 1\}$ weights of *BinaryConnect* in a recurrent network.

2 METHODOLOGY AND EXPERIMENTS

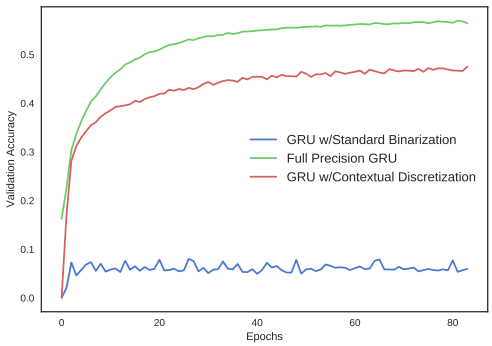
Ott et al. (2016) noted that they were unable to successfully train a recurrent neural network when all weights were discretized: only seeing any improvement after a large number of epochs, long after the other models had converged. To resolve this, we run an experiment training a Gated Recurrent Unit(GRU)-RNN on the Shakespeare dataset task [1] with full precision, extracting the weights of the trained network after 50 epochs. We see in Fig 1a that the distribution of hidden-to-hidden weights is *narrower* than that of input-to-hidden weights by several orders of magnitude, and we notice the existence of a great many hidden-to-hidden weights of extremely small size. Based on this observation, we train a GRU as before, with deterministic ternarization and with all weights binarized. Knowing that hidden-to-hidden weights tend to be smaller than input-to-hidden weights, we divide the acceptable values for weights into two: hidden-to-hidden weights are constrained to lie in the set

*This work was done as part of the UnifyID AI-fellowship, Fall 2016 session

[1] This entails training a character-level recurrent network for next-token prediction on the concatenated works of Shakespeare (Karpathy, 2015). We use the last 196,608 characters as a validation set, leaving us 4,128,768 characters to train on.



(a) A distribution of the weights (from all layers) of a GRU trained with full-precision. Here we plot the logarithm of the absolute value of each weight. Red represents weights from the input-to-hidden matrices, Blue represents weights from the hidden-to-hidden matrices. Biases are excluded, but were distributed in a similar fashion as the hidden-to-hidden weights.



(b) Validation accuracy for the GRU for the Shakespeare dataset

Figure 1: GRU for the Shakespeare dataset

$\{2^{-7}, -2^{-7}\}$, and input-to-hidden weights are constrained to lie in the set $\{2^{-3}, -2^{-3}\}$. We note that, while this discretization is more complicated, it still eliminates all high precision multiplications and requires only storing one bit per weight. Additionally, we no longer wish to clip gradients by the global norm, since there is an intentional imbalance in weight magnitude, and instead simply clip each gradient by value to lie in $[-1, 1]$. This process, which we call **Contextual Discretization**, leads for the first time to a successfully trained RNN with wholly discretized binary weights. The validation accuracy curves are as shown in Fig 1b.

Buoyed by this successful RNN-discretization, we now embark on a completely new *Ma vlast task*[2] and test the efficacy of Contextual Discretization on plain vanilla RNNs (V-RNNs) as well as ClockWork-RNNs (CW-RNNs).

As in Koutnik et al. (2014), we train a two-layer Clockwork RNN with 512 output neurons in each layer, with a fully-connected layer on top to produce the final value for each timestep. For this experiment, we compare the CW-RNN with the standard V-RNN. Furthermore, we train three variants of each network: one with full precision weights, one with binary weights of magnitude 0.5, and one with contextually binarized weights with hidden-to-hidden weight magnitude of 0.125 and input-to-hidden magnitude of 0.5. The results are summarized in Fig 2a and Fig 2b.

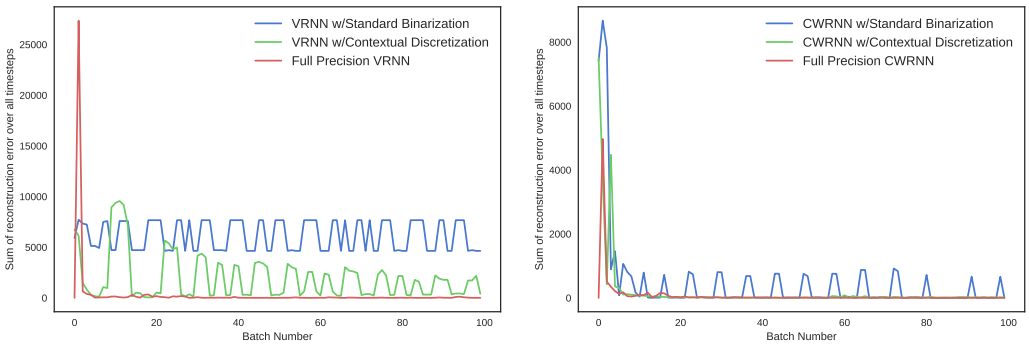
We see that both the full precision CW-RNN and the full precision V-RNN are able to accurately learn the target sequence. For the V-RNN, however, we see in Fig 2a that the standard binarized network is completely unable to improve its performance over time, with the loss oscillating repeatedly without improvement. The contextually discretized V-RNN, however, is able to improve, and though it too oscillates, the contextually discretized V-RNN improves notably, approaching the accuracy of the full precision model.

The CW-RNN, which was practically designed for this task, fairs better, and in 2b we see that the CW-RNN with standard binarization is able to perform fairly well, but it also oscillates (though with less magnitude), and is unable to consistently maintain good performance. However quite notably, the contextually discretized CW-RNN does not oscillate at all and is able to match the performance of the full precision CW-RNN with only a minor lag in convergence.

2.1 Modified GRU for recovering weights constrained to the binary set

We have seen that Contextual Discretization solves the convergence issues faced by standard fully binary recurrent networks. Though these networks are truly binary, they have one pitfall in that the discrete weights are not strictly in $\mathcal{B}$. While only 1-bit of information is needed per weight, there are four distinct weights contained in the network. Furthermore, one of the original points of the

[2]In this task, a small sub-sequence of the *Má vlast*, the classical symphony .wav music file is isolated (in this case, 3400 inputs). The recurrent network is then trained, starting with a zero-valued hidden state, to output the values of the sub-sequence, *being given no input* (to the first layer). In this way, the network can be seen as learning a compressed version of the sequence in its trained weights.



(a) VRNN (b) CWRNN

Figure 2: Reconstruction error analysis for the Ma Vlast dataset

Courbariaux et al. (2015) paper was that matrix multiplications can be replaced with a sequence of additions and subtractions instead of high-precision multiplications. While we have still replaced the high-precision multiplications, floating point bit shifts ($O(n^3)$ of them in total) still remain. This is faster, but not as fast as possible. To rectify this, we propose a slightly modified GRU unit, governed by the following rules:

$$
\begin{aligned}
z_t &= \sigma\left(\alpha W_{zh} h_{t-1} + \beta W_{zx} x_t + b_z\right) \\
r_t &= \sigma\left(\alpha W_{rh} h_{t-1} + \beta W_{rx} x_t + b_r\right) \\
\hat{h}_t &= \tanh\left(\alpha W_{\hat{h}h}\left(r_t \bullet h_{t-1}\right) + \beta W_{\hat{h}x} x_t + b_{\hat{h}}\right) \\
h_t &= \left(1 - z_t\right) \bullet h_{t-1} + z_t \bullet \hat{h}_t
\end{aligned}
$$

The only difference is in the additional of the α and β parameters before each of the weight matrices. We restrict these to be powers of two, effectively making each multiplication a massively parallel bit shifting. We once again restrict the weights of *all* weights in the network to lie in $\{-1, 1\}$. Imagining that $\alpha = 2^{-2} = 0.25$, this network is exactly equivalent to a Contextually Discretized GRU where the weights are constrained to lie in $\{-0.25, 0.25\}$. This network, therefore, can be made identical to the Contextually Discretized network from before. By extracting out the multiplicative factor, we once again only need to compute additions and subtractions for each of the $O(n^3)$ operations in the weight-activation matrix multiplication and have reduced the number of power-two bit shifts down to only $O(n^2)$ on the resulting product. This means that our network can take advantage of specialized hardware for computing $\{-1, 1\}$ binary weight multiplications and retain the advantages of Contextualized Discretization.

3 Conclusion and Future Work

Addressing the inability of 1-bit RNNs to effectively learn (as showcased in (Ott et al., 2016; Hubara et al., 2016)) we have introduced a Contextualized Discretization technique[3], and shown that specifying the allowable weights in a particular fashion allows us to train 1-bit recurrent neural networks. This approach cannot take advantage of specialized hardware for $\{-1, 1\}$ weight matrix multiplication, and we therefore propose a small change to the structure of the Recurrent Networks, which allows us to reclaim the exclusive usage of weights in $\{-1, 1\}$, and reduces the number of power-two bit multiplications from $O(n^3)$ to $O(n^2)$. The weight scales for the GRU were empirically established by observing the weights of a full precision network, but we intentionally did not examine the weights of the full-precision Clockwork RNN, and we were still able to significantly improve over naive binarization. On one hand, this suggests that contextual discretization of recurrent weights is generalizable, since the same scaling works on two very different networks and two very different problems, but it also suggests that performance could be brought even closer to full-precision networks by more careful weight scaling.

[3]All the code pertaining to the experiments are shared at `https://github.com/Cogitans/BRNN`.

REFERENCES

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in Neural Information Processing Systems*, pp. 3123–3131, 2015.

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. *arXiv preprint arXiv:1602.02830*, 2016.

Lu Hou, Quanming Yao, and James T Kwok. Loss-aware binarization of deep networks. *arXiv preprint arXiv:1611.01600*, 2016.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *arXiv preprint arXiv:1609.07061*, 2016.

Andrej Karpathy. The unreasonable effectiveness of recurrent neural networks. *Andrej Karpathy blog*, 2015.

Jan Koutnik, Klaus Greff, Faustino Gomez, and Juergen Schmidhuber. A clockwork rnn. *arXiv preprint arXiv:1402.3511*, 2014.

Fengfu Li, Bo Zhang, and Bin Liu. Ternary weight networks. *arXiv preprint arXiv:1605.04711*, 2016.

Joachim Ott, Zhouhan Lin, Ying Zhang, Shih-Chii Liu, and Yoshua Bengio. Recurrent neural networks with limited numerical precision. *arXiv preprint arXiv:1608.06902*, 2016.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pp. 525–542. Springer, 2016.