Systematically designing better instance counting models on cell images with Neural Arithmetic Logic Units

Ashish Rana[1], Taranveer Singh[3], Harpreet Singh[3], Neeraj Kumar[4], Prashant Singh Rana[5], and Vinay Arora[6]

Thapar Institute of Engineering and Technology, India
{`arana_be15, tsingh_me17, harpreet.s, neeraj.kumar, prashant.singh, vinay.arora`}@thapar.edu

Abstract. The big problem for neural network models which are trained to count instances is that whenever testing range data goes to higher counts than training range data generalization error increases i.e. prediction error for images that are outside training range increases. Consider the case of automating cell counting process where more dense images with higher cell counts are commonly encountered as compared to images used in training data. By making better predictions for higher ranges of cell count we are aiming to create better generalization systems for cell counting. With architecture proposal of neural arithmetic logic units (NALU) for arithmetic operations, task of counting has become feasible for higher numeric ranges which were not included in training data with better accuracy. In our study we incorporate these units in already existing Fully Convolutional Regression Network (FCRN) and U-Net architectures in the form of residual concatenated layers. We carried out a systematic comparative study with the newly proposed changes and earlier base architectures. This comparative study results are evaluated in terms of optimizing regression loss across cell density map over an image. We achieved better results in cell counting tasks with our newly proposed architectures having residual layer concatenation connections. We further validated our results on custom created high count dataset created from BBBC005 synthetic cell count dataset and obtained even better results where testing images have higher counts. These results confirm that above mentioned numerically biased units does help models to learn numeric quantities for better generalization results on high count data.

IEEE Keywords—Neural Arithmetic Logic Units, Cell Counting, Fully Convolutional Regression Networks, U-net.

1   Introduction

Ability to generalize concepts is fundamental component of intelligence and core for designing smart systems [1,2]. Neural networks simulates this behavior with hierarchical learning of concepts. When it comes to automation, counting is an important task from machine vision application [3] to cell counting [4]. While neural networks manipulates numerical quantities but it is not associated with systematic generalization [5,6]. These networks fail to generalize as evident from high generalization error while predicting quantities that lie outside the training numerical range [7]. This highlights memorization behavior in neural networks instead of generalization abilities for a given task. This is especially problematic for cell counting tasks as images with higher cell counts that were not part of training are common to encounter in real applications.

Neural accumulators (NAC) and neural arithmetic logic units (NALU) [7] are biased to learn systematic numerical computation and performs relatively better than non linear activation functions for arithmetic operations. This numerical bias of learning computations makes them excellent choice for counting tasks which are essentially is an increment addition operation only. Deep learning models generally take either segmentation approach with explicit counting trainer or end-to-end counting via a regression loss. In this paper we will go through the latter approach [4] in detail for automation of cell counting process. As cell counting is cumbersome task and dense cell images with higher cell counts containing data outside training numeric range are common in real world scenarios. Achieving true cell automation with less generalization errors is the prime objective of this paper.

In regression loss approach Fully convolutional regression networks [2] and U-net [8] architectures learns mapping between an image I(x) and a density map D(x), given by $F\colon I(x) \rightarrow D(x)$ ($I \in R^{m \times n}$, $D \in R^{m \times n}$) for a $m \times n$ pixel image. Later on, variations of these base architectures with different activation functions are implemented in this paper and their prediction performance is compared with newly proposed NAC/NALU concatenated architectures of FCRN and U-net. The concatenated connections adds numerical bias to layers of the network and behaves in similar manner like ResNets [9]. But, instead of addition of inputs from previous layer the input from previous is passed through these numerical units for capturing numerical bias and concatenated with the main network layer as shown in figure 1. On surface it appears that these proposed architectural changes leads to accuracy improvements due to increased model capacity with these numerically biased units adding more parameters for learning. But, our results with custom high count dataset for testing created from BBBC005 [10] reflects increased generalization counting abilities. As for higher cell count images our model gives higher relative improvement in mean absolute error(MAE).

Our concatenation based residual architecture utilizes the fundamentals of batch normalization like specified identity mapping architecture [11] in ResNets. But, instead of using convolution operation directly this network leverages numerical bias information obtained from NAC and NALU operations applied on
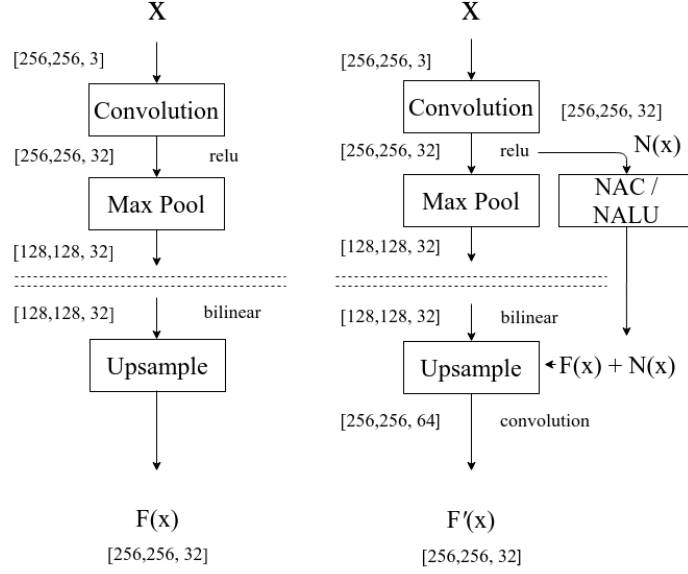
X   X

[256,256, 3]   [256,256, 3]

Convolution   Convolution   [256,256, 32]

[256,256, 32]   relu   [256,256, 32]   relu   N(x)

Max Pool   Max Pool   NAC / NALU

[128,128, 32]   [128,128, 32]

[128,128, 32]   bilinear   [128,128, 32]   bilinear

Upsample   Upsample   ◄ F(x) + N(x)

                                     [256,256, 64]   convolution

F(x)   F′(x)
[256,256, 32]   [256,256, 32]

Fig. 1. Abstract representation of proposed modified architecture (*Right*) with either NAC or NALU as compared to previous Fully Convolution Regression Networks (*Left*). Here, as clearly specified with dimensional analysis of tensor blocks (*Right*) the layers in the newly proposed model are concatenated and again squeezed into same size as earlier base model architecture.

input layer and then finally uses convolution operation on the concatenated layer. Before and after this concatenation of this numerical bias learning operation, batch normalization is carried out and output of this operation is added back again to our next main network layer, as shown in figure 1.

With means of this paper we introduce changes in current regression based model architectures for end-to-end counter training and produce systems with improved accuracy. Also, we validate our trained models on a different specially tailored validation dataset with approximately seven times higher counts of cells as compared to training dataset created from BBBC005 synthetic cell dataset [10]. Overall, experimental results demonstrates that supplementing the base architectures with NAC and NALU helped in achieving better results and improved relative MAE for higher count test images.

2   Related Work

Intuitive numerical understanding is important in learning and by adjunct important in deep learning [1] for creating better models with higher generalization capabilities. Counting objects [12–16] in given image is a widely studied task. Trained models for counting tasks either use a deep learning model to seg-

ment instances of given object then count them in a post-processing step [17] or learn end-to-end predict count via a regression loss [4]. Networks like Count-ception [18] added the concept of average over redundant predictions with its deep Inception family network. Also, recent architectures like ResNets [19], High-way Networks [20] and Densenets [21] advocate linear connections like Count-ception to promote better learning bias. Such models have better performance, though additional computational overhead due to increased depth of given ar-chitectures do arise. Our work highlights the generalization capabilities of the network, that extrapolate well on unseen parts of solution space which highlights underlying structure of behavior governing-equations [22].

We introduce architectural changes in models that learns and preserves input information which is numerically biased with reference to input layers. It is somewhat similar to ResNets [9], which are easier to optimize and gain accuracy with increasing depth. With our experiments we aim to highlight that models with numerically biased concatenated residual functions helps in achieving better results with their addition in the form of a comparative study with original architectures. Also, with our results in this paper we demonstrate with our results that backpropagation learns this numerical bias without any explicit numeric quantity being provided as input implying that better computer vision counters can be trained with this module when added to existing convolutional neural network architectures.

Density based estimation doesnt require prior object detection or segmenta-tion [12,15,23]. In previous years, several works have investigated this approach. In [15], the problem is stated as density estimation with a supervised learning algorithm, $D(x) = c^T \phi(x)$, where $D(x)$ represents ground-truth density map, and $\phi(x)$ represents local features and parameters c are learned by minimiz-ing the error between predicted and true density with quadratic programming over all possible sub-windows. In [23], regression forest is used to exploit patch-based idea for learning structured labels, then for new input image density map is estimated averaged over structured patch-based predictions. Also, in [12] an algorithm is used that allows fast interactive counting with ridge regression.

Cell counting [15] problem is classified into supervised learning problem that learns mapping between an image $I(x)$ and a density map $D(x)$, denoted by F: $I(x) \rightarrow D(x)$ $(I \in R^{m \times n}, D \in R^{m \times n})$ for a $m \times n$ pixel image, see figure 2. Density function $D(x)$ function is defined on pixels in given image, integrat-ing this map over an image region gives an estimate of number of cells in that region. CNNs [24, 25] are quite popular in the bio-medical imaging because of their simple architecture and achieve great results. Like in mitosis detection [26], neuronal membrane segmentation [27] and analysis of C. elegans embryos devel-opment [28]. Previously, fully convolutional regression networks (FCRNs) and Count-ception have given state-of-the-art results in cell counting, with potential for cell detection of overlapping cells.

Also U-Nets [8], typically fully convolutional network uses a modified version of architecture proposed by Ciresan et al. [27] as latter is slow and trade-off between localization and use of context are present. In U-Nets pooling operations
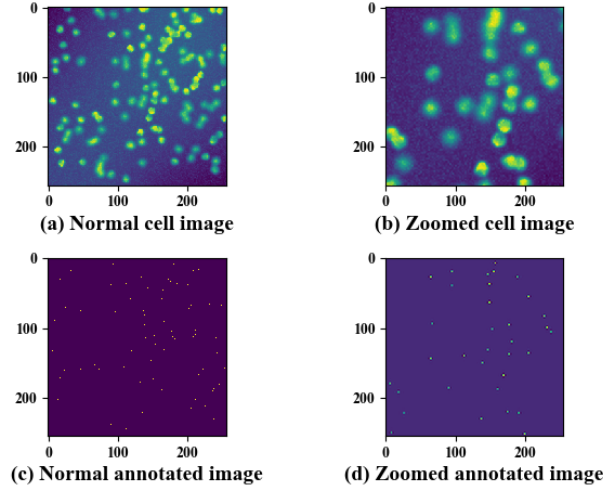
(a) Normal cell image

(b) Zoomed cell image

(c) Normal annotated image

(d) Zoomed annotated image

Fig. 2. The left column represents training image in first row and corresponding annotated image in second row from fluorescent synthetic dataset [29]. The second column of highlights corresponding zoomed view for better visual comprehension of the images in the dataset.

are replaced by upsampling operations to supplement usual contracting network. For localization high resolution features from contracting path are combined with unsampled output. Based on this information a successive convolution layer then learn to assemble more precise output. For our experimentation we selected FCRN and U-net based on simplicity and relative similarity in their architectures with the difference being that U-net uses inputs from previous layer for better localization.

3   Experiments

In this section first we conceptually explore NACs and NALUs. Then compare their addition capabilities with multi-layer perceptrons equipped with different activation functions. With this study we aim to select NAC/NALU designed variants which can best approximate the counting behavior and compare them with standard FCRN and U-net neural network architecture for regression loss approach in the following experiment done on synthetic dataset [29]. For validation of counting generalization capabilities, our trained models are tested against different synthetic cell image dataset with approximately seven times higher counts than training data.

3.1   Visual understanding of NACs and NALUs

NACs [7] supports accumulation of numerical quantities additively, a desirable bias for linear exploration while counting. It is special type of linear layer with transformation matrix **W** being continuous and differentiable parameterization for gradient descent. $W = \tanh(\hat{W}) \odot \sigma(\hat{M})$ consists of elements in [-1, 1] with bias close to 1, 0, and 1. See figure 3 for ideation of this concept with following equations for NAC: $a = Wx$, $W = \tanh(\hat{W}) \odot \sigma(\hat{M})$ where $\hat{W}$, $\hat{M}$ are learning parameters and W is transformation matrix.

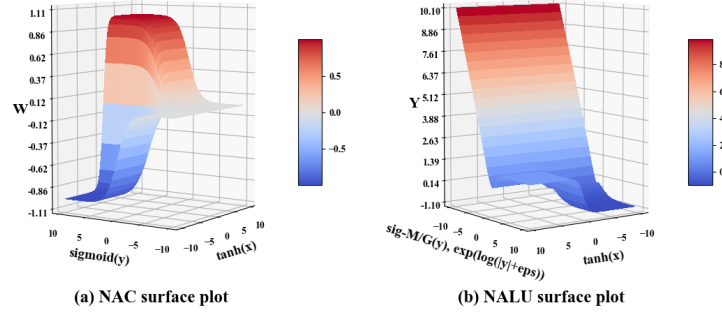(a) NAC surface plot   (b) NALU surface plot

Fig. 3. Left: NACs are biased towards learning [-1, 0, 1] as highlighted with large plateau regions around these values. This means its outputs are either addition or subtraction of input vectors not scaling. **Right:** Approximate surface curve of NALU with some dimensional constraints for 3-D plotting. It highlights the ability to scale, along with earlier numerical biases around [-1, 0, 1] as shown with plateau region surfaces.

For complex mathematical operations like multiplication and division we use NALUs. It uses weighted sum of two sub-cells, one for addition or subtraction and another of multiply, division or power functions. It demonstrates that neural accumulators (NACs) can be extended for learning scaling operations with gate-controlled sub-operations. See figure 3 for ideation of this concept with following equations for NALU: $y = g \odot a + (1-g) \odot m$; $m = \exp W(\log(|x| + \epsilon))$, $g = \sigma(Gx)$ where m is subcell that operates in *log* space and g is learned gate, both contains learning parameters.

3.2   Comparative analysis of addition operation

Here, we use neural networks with NACs/NALUs and multilayer perceptrons (MLP) with different activation functions but same structures. These are trained

with two randomly generated inputs from uniform distribution a and b with each having 2^{14} data points for training. Prediction capabilities on test data with values ranging up to 10 times the training range are evaluated as part of this experiment. Refer figure 4 to observe architecture for both these trained models in this comparative study.
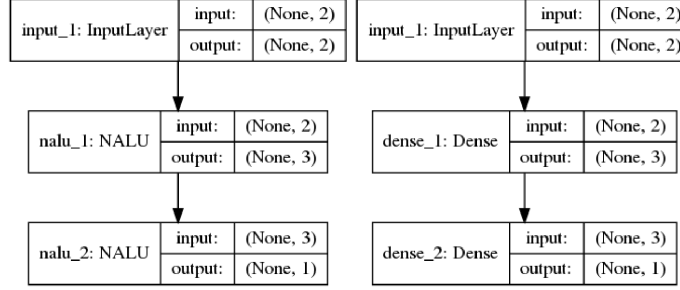

Fig. 4. The model on *left* represents neural network with NAC or NALU units and on *right*, a MLP which is trained with different activation functions. Each variant of above stated models have two inputs, three hidden units, one output.

Comparative analysis is summarized in table 1 with MAE as an accuracy measure for MLP variants, NAC, NALU and its variants with changed learned gate for extrapolation. Also, in NALU-Tanh and NALU-Hard Sigmoid the learning gate $g's$ is changed to observe any improvements in NALU's performance based on this change.

Table 1. Comparative result summarization for multiple models

Layer Configuration/Activations	Mean Absolute Error ($a+b$)
Linear MLP	3.63×10^{-6}
Sigmoid MLP	29.830
Tanh MLP	15.743
ELu MLP	0.019
ReLU MLP	0.001
Leaky ReLU MLP	9.83×10^{-4}
PReLU MLP	0.001
NAC	2.70×10^{-6}
NALU	2.71×10^{-6}
NALU-Hard Sigmoid	3.24×10^{-6}
NALU-Tanh	3.18×10^{-6}

From these results stated in table 1 and visualized in figure 5 we conclude that Linear, LeakyReLU, ReLU activations and NAC, NALU, NALU-Tanh modules were the top performers in extrapolation task for numeric addition operation task. Hence, these top performers are used further in cell-counting task on synthetic dataset for learning end-to-end counting mechanism.
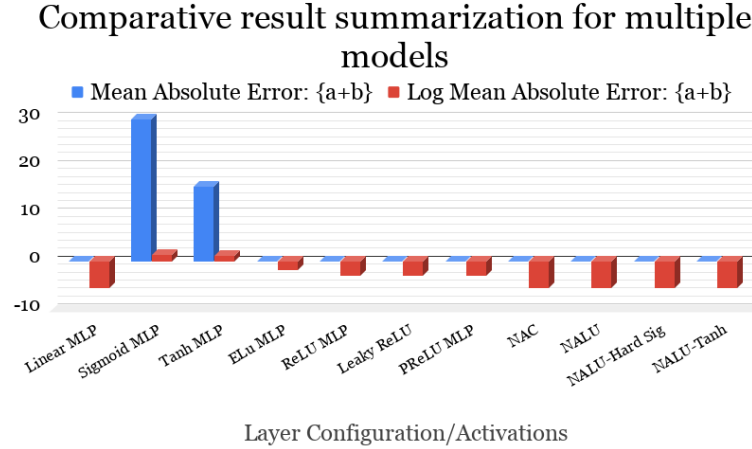

Fig. 5. Above visualization of MAE and $\log_{10}$(MAE) for different models learning identity mapping, demonstrates that our standard activation functions like *sigmoid* and *tanh* doesn't perform that well with higher data ranges during testing. Whereas, *NACs* and *NALUs* clearly overpowers the identity learning task and *linear*, *relu* based activation functions somewhat manages to provide acceptable results.

3.3   Cell counting experiment

[1]

In this experiment section the first subsections elaborates the datasets used for training and validation of our trained models, plus the data augmentation techniques used in our experiment. After that we elaborate onto different architectures used for training having different activation layers on standard architectures and residual concatenated connection modules on modified proposed model architectures.

Datasets and data augmentation Synthetic dataset which is generated by system [29]. 200 highly-realistic synthetic fluorescence microscopic images of bacterial cells are used for experimentation with a 75/25 train-test split for training

---

[1] *Code repository:* https://github.com/ashishrana160796/nalu-cell-counting

each model architecture and its variants. Images are having average of 17464 cells.

For validation of trained models and checking true generalization capabilities we use BBBC005 from the the Broad Institutes Bioimage Benchmark Collection [10]. This dataset is comprised of 600 images have a corresponding foreground mask which alters the focus on these images and ground truth images are completely in-focused version before any guassian filter application. We take a subset of this dataset with highly focused *F1* images only and their corresponding ground truth image for validating our model. And after that we coalesce each image 16 times in 4x4 grid manner with random vertical and horizontal flips to create a final high cell count images for our validation dataset, see figure 6. The ground truth images are accordingly changed with same randomness and they are eroded also to match the fluorescent synthetic dataset on which the models are trained on.
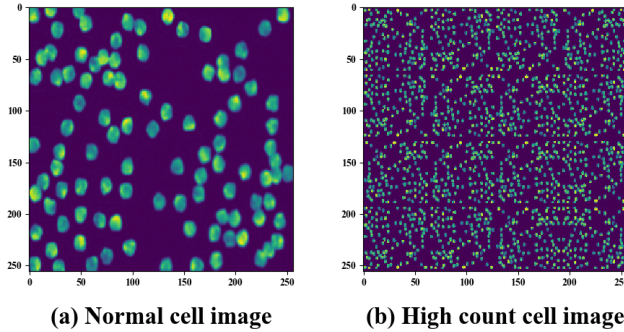

(a) **Normal cell image**   (b) **High count cell image**

Fig. 6. *Left:* Original image from BBBC005 dataset. *Right:* Repeated image generated from a *4X4* grid repetition operation with random horizontal and vertical flips of base image shown in left.

Data augmentation with elastic deformations to training images is applied for teaching network the desired invariance and robustness properties, like specified in figure 7. These elastic deformations are introduced in the form of angular shear in the training images. Translation and rotation invariance along with robustness to gray value variations and deformations is main focus of augmentation process for microscopic images. Disfigurement using random displacement vectors on a coarse 3x3 grid are also generated. These data augmentation techniques especially are helpful for our custom data which is created just by repeating the original image in order to supplement a more robust dataset for the model to train on.

Defining regression task and architecture details In training dataset ground truth is provided as dot annotation corresponding to each cell image.
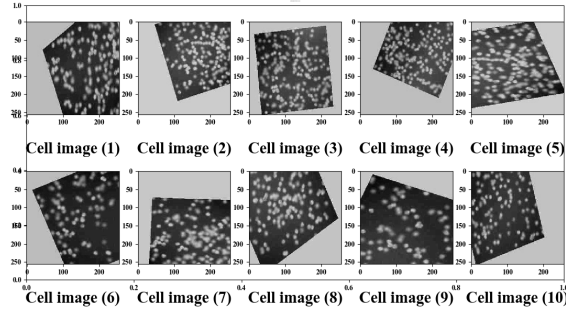
Fig. 7. The above figure illustrates different cell augmentation techniques in action like elastic deformations, random rotations and plane shifting for the input images adding more robustness to the trained model.

For training, dot annotations are represented by Gaussian and density surface D(x) which is formed from superposition of Gaussians. The optimization task is to regress density surface from corresponding image I(x). This is achieved by training convolutional neural networks (CNN) using mean square error between output heat map and target density surface as the loss function. Hence, at inference given an input I(x), the model predicts density heat map D(x).

FCRNs are inspired from VGG-net, we only used small kernels of size 3x3 pixels for designing our network. Feature maps are increased for avoiding spatial information loss. Activation layers like convolution-ReLU-Pooling are popular in CNN architectures [24]. Here, we have altered these layers to create different models which contains some numerical bias in the form of residual connections and regularized by batch normalization. The first layers contains convolutions-pooling operations, then we undo spatial reduction by upsampling operations for end-to-end model training. Also, for dimensional compatibility of residual NAC or NALU modules we append their output to corresponding main network layer across depth dimension and batch normalize the output after appropriate convolutional operation. See figure 8 for comparison between earlier original model and newly proposed architecture along with architecture parameter details.

U-net is modification upon the previously discussed FCRN architecture by having large number of feature channels for upsampling to propagate context information to high resolution layers. That makes expansive path almost symmetric to contracting path yielding a u-shape. Similar to above FCRN's optimization problem formulation the evaluation parameters remains the same. Residual concatenated connection addition with NACs and NALU units along with batch normalization is done for adding numerical bias information to the main network. Also, U-net architecture used in this paper is more computationally expensive than FCRN having approximately thrice the number of parameters leading to more feature learning capacity and is expected to perform better
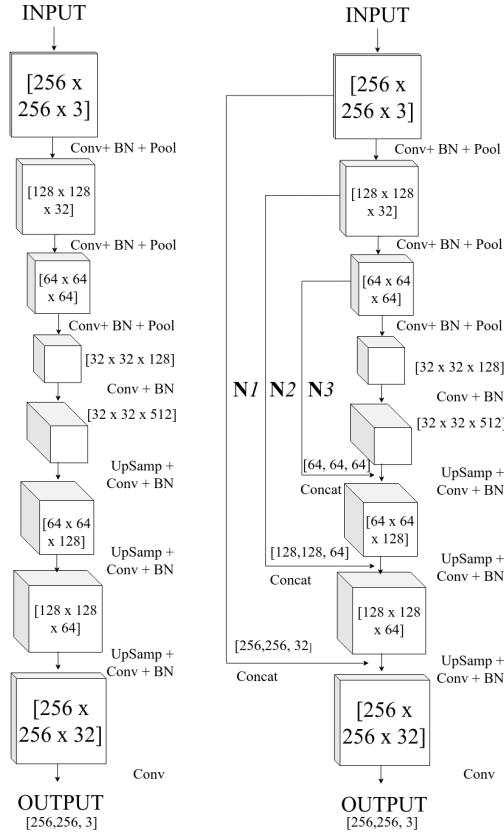
Fig. 8. *Network Architecture Left:* FCRN with 3x3 convolution operations. *Network Architecture Right:* FCRN added with residual concatenation connections of **N***i* numerically biased units. Also, for dimensional compatibility **N***i* residual layers are fed to corresponding main network layers and after that normalized for regularization. At last, 1x1 **Conv** operation output in the form of density map of result image is generated .

Conv + BN + Pool: A 3x3 convolutional operation with batch normalization regularization and 2x2 max pooling layer.

Unsample+Conv + BN: Unsampling the image and then applying convolutional operation with batch normalization regularization.

Concat: Feature maps concatenated along depth dimension.

N*i*: NAC or Variants of NALUs as residual concatenated connection.

for prediction tasks. See figure 9 for comparison between earlier original U-net model and newly proposed architecture along with parameter details.

For the concatenation of residual connections of these units the dimensional consistency is maintained by added them along the depth dimension with the main network. After that upsampling, convolutional and batch normalization

regularization operations are applied accordingly for these units to merge with the base network. FCRNs implementation resembles that of MatConvNet [30] as upsampling in Keras is implemented by repeating elements, instead of bilinear sampling. In U-nets, low-level feature representations are fused during upsampling, aiming to compensate the information loss due to max pooling.

4   Results

MAE is the metric used in this paper for measuring results for cell counting on the synthetic cell dataset [29] and custom BBBC005 synthetic modified high cell count validation dataset.

– **Mean Absolute Error (MAE):** Mean Absolute Error (MAE): The mean absolute error is an average of the difference between the predicted value and true value.

$$AE = \|e_i\| = \|y_i - x_i\| \tag{1}$$

$$MAE = \sum_{i=1}^{n} \frac{|e_i|}{n} \tag{2}$$

– **Relative Improvement Percentage** Relative Improvement Percentage (RIP): Here, in context of this paper it defined as percentage improvement in MAE of a given model with respect to baseline ReLU models for FCRN and U-net architectures. In below equation, M_r is MAE from baseline ReLU model and M_i is model under consideration.

$$RIP_\% = ((M_r - M_i)/M_r) * 100 \tag{3}$$

Result table 2 compares baseline FCRN, U-net architectures with new numerically biased ResNet like connection modules with NACs and NALUs units under current training setup. With our setup we able to obtain similar results as mentioned in earlier reference papers and also we have equipped earlier model architectures with different regularization activations as specified in the table. From earlier ReLU implementation clearly Linear and LeakyReLU activation regularization based baseline models have performed well. Also, for both model structures NAC and NALUs residual modules have outperformed all the earlier specified regular FCRN architecture. And similar results are extended by U-net model evaluation where NALU layer concatenation based U-net outperforms all the models trained for our experiment.

Result table 3 compares performance of above trained models on a new validation dataset containing much higher cell counts for measuring performance on extrapolation capabilities counting tasks. For validation set we have used 300 images of size 256x256 pixels with cell counts averaging around 120012. Here also, NAC and NALU based residual concatenation module based models outperforms earlier architectures for counting tasks. But, this time the relative improvement is even more for NALU based FCRN and U-net models showcasing better generalization abilities of trained models.

Table 2. Result summarization for trained models

FCRN-Models	MAE	U-Net-Models	MAE
ReLU	3.43	ReLU	1.78
LeakyReLU	3.39	LeakyReLU	1.74
Linear	3.34	Linear	1.73
NALU-tanh	3.21	NALU/tanh	1.56
NALU	**3.17**	NALU	**1.42**
NAC	3.23	NAC	1.63

Table 3. Validating trained models for extrapolation cell counting tasks

FCRN-Models	MAE	U-Net-Models	MAE
ReLU	3.04	ReLU	2.87
LeakyReLU	2.99	LeakyReLU	2.62
Linear	2.85	Linear	2.47
NALU-tanh	2.32	NALU-tanh	1.95
NALU	**2.27**	NALU	**1.87**
NAC	2.40	NAC	1.92

Relative improvement in predictions is visualized in figure 10 against ReLU based activation base result for comparison with other activation layer changes in FCRNs/U-nets and concatenation layer NALU/NAC residual connection addition in FCRNs & U-nets. It includes averaged out comparison from multiple executions of training and testing runs for both interpolation testing and extrapolation validation counting tasks for FCRN and U-net variant models with respect to ReLU based FCRN and U-net model. From, this figure it is clearly highlighted that models with NAC and NALUs residual modules have better generalization capabilities for extrapolation counting tasks i.e. they are better generalizers for this given cell counting task with increase in relative improvement in prediction as compared to base ReLU implementation. Here, for measuring generalization capabilities of model in extrapolation task relative improvement metric is selected as it bring more perspective to the improved performance on validation dataset. The performance has shown a general increment for every model as validation dataset consist of highly focused images. Hence, measuring relative improvement in results is more appropriate decision to get understanding of true improvment in model prediction capabilities.

This figure 10 shows more increase in relative improvement as we move right towards horizontal axis for both testing and validation task with extrapolation where in validation extrapolation task NAC/NALU models from which we can conclude that trained models are having better generalization abilities with some

learned numerical bias in their trained weights with which even better predictions for higher count cells is made.

5   Summary

We were able to demonstrate that addition of newly proposed NAC and NALU units in existing architectures in the form of residual concatenation connection layer modules achieves better results. With numerically biased residual connections, higher accuracy for more dense images having higher counts of cells is achieved. Hence, producing more generalized cell counters that provides better predictions for real life use-cases. Finally, for code implementation details and other supplementary experimental results refer to this paper's url https://github.com/ashishrana160796/nalu-cell-counting.
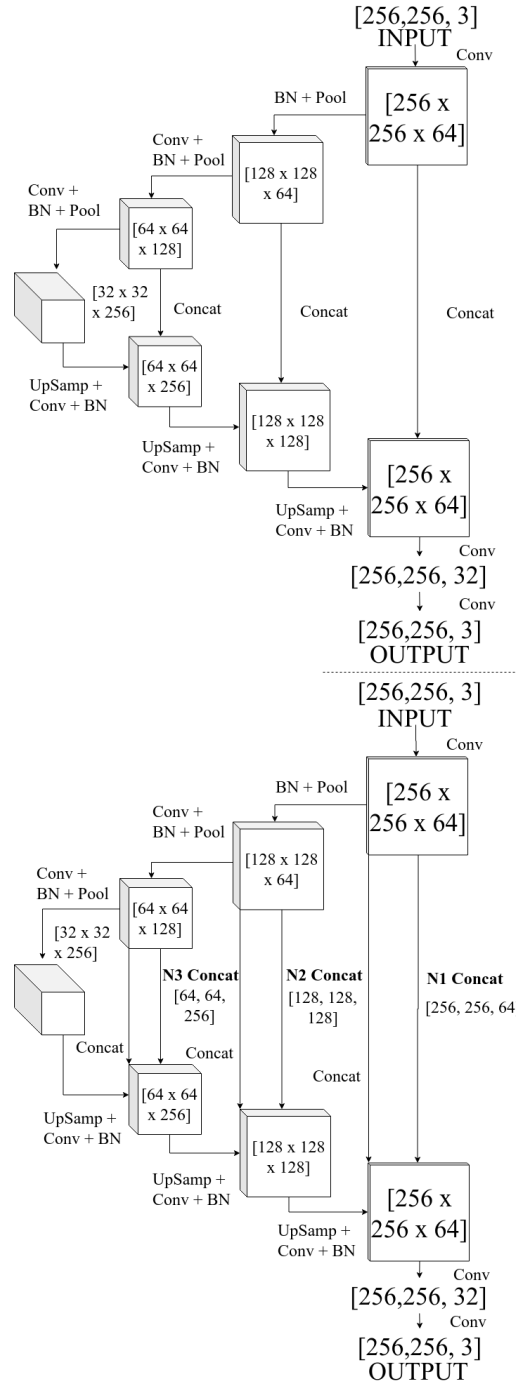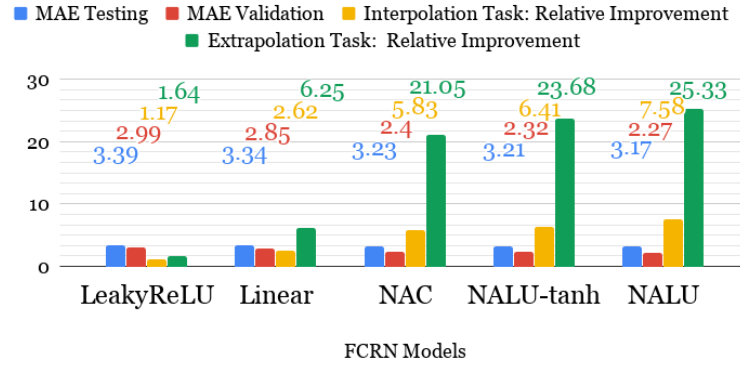
[256,256, 3]
INPUT

[256 x 256 x 64]   Conv

BN + Pool

Conv + BN + Pool

[128 x 128 x 64]

Conv + BN + Pool

[64 x 64 x 128]

Conv + BN + Pool

[32 x 32 x 256]

Concat

Concat

Concat

UpSamp + Conv + BN

[64 x 64 x 256]

UpSamp + Conv + BN

[128 x 128 x 128]

UpSamp + Conv + BN

[256 x 256 x 64]

Conv

[256,256, 32]

Conv

[256,256, 3]
OUTPUT

[256,256, 3]
INPUT

Conv

[256 x 256 x 64]

BN + Pool

Conv + BN + Pool

[128 x 128 x 64]

Conv + BN + Pool

[64 x 64 x 128]

Conv + BN + Pool

[32 x 32 x 256]

N3 Concat
[64, 64, 256]

N2 Concat
[128, 128, 128]

N1 Concat
[256, 256, 64]

Concat

Concat

Concat

UpSamp + Conv + BN

[64 x 64 x 256]

UpSamp + Conv + BN

[128 x 128 x 128]

UpSamp + Conv + BN

[256 x 256 x 64]

Conv

[256,256, 32]

Conv

[256,256, 3]
OUTPUT

Fig. 9. *Network Architecture Top:* U-net with 3x3 convolution operations and increasing depth of filters from 64 to 256 for feature abstraction & learning and after that these layers are fed to upsampling layers. *Network Architecture Bottom:* U-nets main network added with residual concatenation connections of **N**i numerically biased units and after that batch normalized for regularization. Similar, operation abbreviations used as stated in figure 8
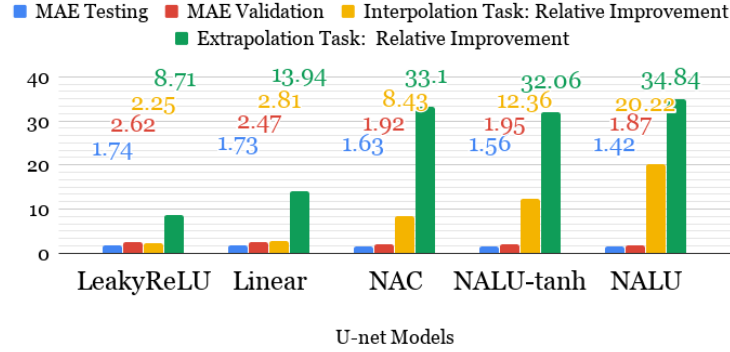
Fig. 10. Moving towards right in above figure and measuring RIP metric there is sharp increase with NALU & NAC units based models demonstrating even more increase in relative improvement with respect to baseline ReLU models for extrapolation tasks. For interpolation task there is highest 9% improvement and for extrapolation there is 23% improvement with NAC units as residual connection.

References

1. Dehaene, Stanislas. The number sense: How the mind creates mathematics. OUP USA, 2011.
2. Gallistel, C. Randy. "Finding numbers in the brain." Philosophical Transactions of the Royal Society B: Biological Sciences 373.1740 (2018): 20170119.
3. Baygin, Mehmet, et al. "An Image Processing based Object Counting Approach for Machine Vision Application." arXiv preprint arXiv:1802.05911 (2018).
4. Xie, Weidi, J. Alison Noble, and Andrew Zisserman. "Microscopy cell counting and detection with fully convolutional regression networks." Computer methods in biomechanics and biomedical engineering: Imaging & Visualization 6.3 (2018): 283-292.
5. Fodor, Jerry A., and Zenon W. Pylyshyn. "Connectionism and cognitive architecture: A critical analysis." Cognition 28.1-2 (1988): 3-71.
6. Marcus, Gary F. The algebraic mind: Integrating connectionism and cognitive science. MIT press, 2018.
7. Trask, Andrew, et al. "Neural arithmetic logic units." Advances in Neural Information Processing Systems. 2018.
8. Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.
9. He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
10. Ljosa, Vebjorn, Katherine L. Sokolnicki, and Anne E. Carpenter. "Annotated high-throughput microscopy image sets for validation." Nature methods 9.7 (2012): 637-637.
11. He, Kaiming, et al. "Identity mappings in deep residual networks." European conference on computer vision. Springer, Cham, 2016.
12. Arteta, Carlos, et al. "Interactive object counting." European conference on computer vision. Springer, Cham, 2014.
13. Chan, Antoni B., Zhang-Sheng John Liang, and Nuno Vasconcelos. "Privacy preserving crowd monitoring: Counting people without people models or tracking." 2008 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2008.
14. Segu, Santi, Oriol Pujol, and Jordi Vitria. "Learning to count with deep object features." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2015.
15. Lempitsky, Victor, and Andrew Zisserman. "Learning to count objects in images." Advances in neural information processing systems. 2010.
16. Zhang, Cong, et al. "Cross-scene crowd counting via deep convolutional neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
17. Hernndez, Carlos X., Mohammad M. Sultan, and Vijay S. Pande. "Using Deep Learning for Segmentation and Counting within Microscopy Data." arXiv preprint arXiv:1802.10548 (2018).
18. Paul Cohen, Joseph, et al. "Count-ception: Counting by fully convolutional redundant counting." Proceedings of the IEEE International Conference on Computer Vision. 2017.
19. He, Kaiming, et al. "Identity mappings in deep residual networks." European conference on computer vision. Springer, Cham, 2016.

20. Srivastava, Rupesh Kumar, Klaus Greff, and Jrgen Schmidhuber. "Highway networks." arXiv preprint arXiv:1505.00387 (2015).
21. Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
22. Brunton, Steven L., Joshua L. Proctor, and J. Nathan Kutz. "Discovering governing equations from data by sparse identification of nonlinear dynamical systems." Proceedings of the National Academy of Sciences 113.15 (2016): 3932-3937.
23. Fiaschi, Luca, et al. "Learning to count with regression forest and structured labels." Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012). IEEE, 2012.
24. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.
25. LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.
26. Cirean, Dan C., et al. "Mitosis detection in breast cancer histology images with deep neural networks." International Conference on Medical Image Computing and Computer-assisted Intervention. Springer, Berlin, Heidelberg, 2013.
27. Ciresan, Dan, et al. "Deep neural networks segment neuronal membranes in electron microscopy images." Advances in neural information processing systems. 2012.
28. Ning, Feng, et al. "Toward automatic phenotyping of developing embryos from videos." IEEE Transactions on Image Processing 14 (2005): 1360-1371.
29. Lehmussola, Antti, et al. "Computational framework for simulating fluorescence microscope images with cell populations." IEEE transactions on medical imaging 26.7 (2007): 1010-1016.
30. Vedaldi, Andrea, and Karel Lenc. "Matconvnet: Convolutional neural networks for matlab." Proceedings of the 23rd ACM international conference on Multimedia. ACM, 2015.