WAVELET POOLING FOR CONVOLUTIONAL NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Convolutional Neural Networks continuously advance the progress of 2D and 3D image and object classification. The steadfast usage of this algorithm requires constant evaluation and upgrading of foundational concepts to maintain progress. Network regularization techniques typically focus on convolutional layer operations, while leaving pooling layer operations without suitable options. We introduce Wavelet Pooling as another alternative to traditional neighborhood pooling. This method decomposes features into a second level decomposition, and discards the first-level subbands to reduce feature dimensions. This method addresses the overfitting problem encountered by max pooling, while reducing features in a more structurally compact manner than pooling via neighborhood regions. Experimental results on four benchmark classification datasets demonstrate our proposed method outperforms or performs comparatively with methods like max, mean, mixed, and stochastic pooling.

1 INTRODUCTION

Convolutional Neural Networks (CNNs) have become the standard-bearer in image and object classification (Nielsen, 2015). Due to the layer structures conforming to the shape of the inputs, CNNs consistently classify images, objects, videos, etc. at a higher accuracy rate than vector-based deep learning techniques (Nielsen, 2015). The strength of this algorithm motivates researchers to constantly evaluate and upgrade foundational concepts to continue growth and progress. The key components of CNN, the convolutional layer and pooling layer, consistently undergo modifications and innovations to elevate accuracy and efficiency of CNNs beyond previous benchmarks.

Pooling has roots in predecessors to CNN such as Neocognitron, which manual subsampling by the user occurs (Fukushima, 1979), and Cresceptron, which introduces the first max pooling operation in deep learning (Weng et al., 1992). Pooling subsamples the results of the convolutional layers, gradually reducing spatial dimensions of the data throughout the network. The benefits of this operation are to reduce parameters, increase computational efficiency, and regulate overfitting (Boureau et al., 2010).

Methods of pooling vary, with the most popular form being max pooling, and secondarily, average pooling (Nielsen, 2015; Lee et al., 2016). These forms of pooling are deterministic, efficient, and simple, but have weaknesses hindering the potential for optimal network learning (Lee et al., 2016; Yu et al., 2014). Other pooling operations, notably mixed pooling and stochastic pooling, use probabilistic approaches to correct some of the issues of the prior methods (Yu et al., 2014; Zeiler & Fergus, 2013).

However, one commonality all these pooling operations employ a neighborhood approach to subsampling, reminiscent of nearest neighbor interpolation in image processing. Neighborhood interpolation techniques perform fast, with simplicity and efficiency, but introduce artifacts such as edge halos, blurring, and aliasing (Parker et al., 1983). Minimizing discontinuities in the data are critical to aiding in network regularization, and increasing classification accuracy.

We propose a wavelet pooling algorithm that uses a second-level wavelet decomposition to subsample features. Our approach forgoes the nearest neighbor interpolation method in favor of an organic, subband method that more accurately represents the feature contents with less artifacts. We compare our proposed pooling method to max, mean, mixed, and stochastic pooling to verify its validity,

and ability to produce near equal or superior results. We test these methods on benchmark image classification datasets such as Mixed National Institute of Standards and Technology (MNIST) (Lecun et al., 1998), Canadian Institute for Advanced Research (CIFAR-10) (Krizhevsky, 2009), Street House View Numbers (SHVN) (Netzer et al., 2011), and Karolinska Directed Emotional Faces (KDEF) (Lundqvist et al., 1998). We perform all simulations in MATLAB R2016b.

The rest of this paper organizes as follows: Section 2 gives the background, Section 3 describes the proposed methods, Section 4 discusses the experimental results, and Section 5 gives the summary and conclusion.

2 BACKGROUND

Pooling is another term for subsampling. In this layer, the dimensions of the output of the convolutional layer are condensed. The dimensionality reduction happens by summarizing a region into one neuron value, and this occurs until all neurons have been affected. The two most popular forms of pooling are max pooling and average pooling (Nielsen, 2015; Lee et al., 2016). Max pooling involves taking the maximum value of a region R_{ij} and selecting it for the condensed feature map. Average pooling involves calculating the average value of a region and selecting it for the condensed feature map. The max pooling function is expressed as:

$$a_{kij} = \max_{(p,q)\in R_{ij}} (a_{kpq}) \tag{1}$$

While average pooling is shown by the following equation:

$$a_{kij} = \frac{1}{|R_{ij}|} \sum_{(p,q)\in R_{ij}} a_{kpq} \tag{2}$$

Where a_{kij} is the output activation of the kth feature map at *(i,j)*, a_{kpq} is the input activation at *(p,q)* within R_{ij}, and $|R_{ij}|$ is the size of the pooling region. An illustration of both of these pooling methods is expressed in Figure 1 (Williams & Li, 2016):

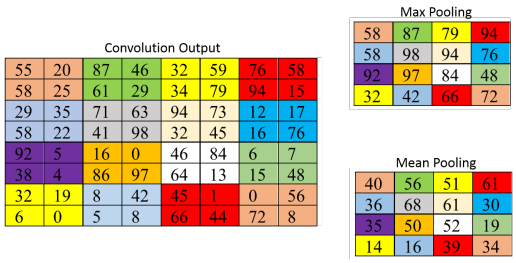


Figure 1: Example of Max & Average Pooling with Stride of 2

While max and average pooling both are effective, simple methods, they also have shortcomings. Max pooling, depending on the data, can erase details from an image (Yu et al., 2014; Zeiler & Fergus, 2013). This happens if the main details have less intensity than the insignificant details. In addition, max pooling commonly overfits training data (Yu et al., 2014; Zeiler & Fergus, 2013). Average pooling, depending on the data, can dilute pertinent details from an image. The averaging of data with values much lower than significant details causes this action (Yu et al., 2014; Zeiler & Fergus, 2013). Figure 2 illustrates these shortcomings using the toy image example:

To combat these issues, researchers have created probabilistic pooling methods. Mixed pooling combines max and average pooling by randomly selecting one method over the other during training (Yu et al., 2014). There is no set way to perform mixed pooling. This method is applied arbitrarily in three different ways (1) for all features within a layer, (2) mixed between features within a layer, or (3) mixed between regions for different features within a layer (Lee et al., 2016; Yu et al., 2014).

Figure 2: Shortcomings of Max & Average Pooling using Toy Image

Mixed pooling is shown in the following equation:

$$a_{kij} = \lambda \cdot \max_{(p,q)\in R_{ij}} (a_{kpq}) + (1-\lambda) \cdot \frac{1}{|R_{ij}|} \sum_{(p,q)\in R_{ij}} a_{kpq} \tag{3}$$

where λ is a random value 0 or 1, indicating max or average pooling for a particular region/feature/layer.

Another probabilistic pooling method, called stochastic pooling, improves upon max pooling by randomly sampling from neighborhood regions based on the probability values of each activation (Zeiler & Fergus, 2013). These probabilities p for each region are calculated by normalizing the activations within the region:

$$p_{pq} = \frac{a_{pq}}{\sum_{(p,q)\in R_{ij}} a_{pq}} \tag{4}$$

The pooled activation is sampled from a multinomial distribution based on p to pick a location l within the region (Zeiler & Fergus, 2013). The process is captured in the following equation (Zeiler & Fergus, 2013):

$$a_{kij} = a_l \quad where \quad l \sim P(p_1, ..., p_{|R_{ij}|}) \tag{5}$$

Figure 3 displays a visual example of stochastic pooling on a 3x3 region:

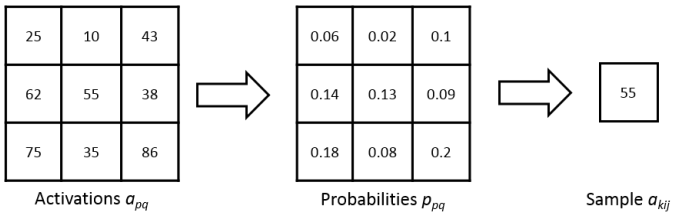


Figure 3: Stochastic Pooling Example

In Figure 3, a region of activations are shown, and in the adjacent region, their corresponding probabilities based on Equation 4. In any given region, the activations with the highest probabilities have the higher chance of selection. However, any activation can be chosen. In this example, the stochastic pooling method selects the midrange activation with a probability of 13%. By being based off probability, and not deterministic, stochastic pooling avoids the shortcomings of max and average pooling, while enjoying some of the advantages of max pooling (Zeiler & Fergus, 2013).

3 Proposed Method

The previously highlighted pooling methods use neighborhoods to subsample, almost identical to nearest neighbor interpolation. Previous studies explore the possibilities of wavelets in image interpolation versus traditional methods (Dumic et al., 2007). Our proposed pooling method uses

wavelets to reduce the dimensions of the feature maps. We propose using the wavelet transform to minimize artifacts resulting from neighborhood reduction (Parker et al., 1983). We postulate that our approach, which discards the first-order subbands, more organically captures the data compression. This organic reduction therefore lessens the creation of jagged edges and other artifacts that may impede correct image classification.

3.1 Forward Propagation

The proposed wavelet pooling scheme pools features by performing a 2nd order decomposition in the wavelet domain according to the two-dimensional discrete wavelet transform (DWT) as follows (Chui, 1992; Strang & Strela, 1995; Rieder et al., 1994):

$$W_\phi(j_0, k, l) = \frac{1}{\sqrt{M \cdot N}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\phi_{j_0,k,l}(m,n) \tag{6}$$

$$W_\psi(j, k, l) = \frac{1}{\sqrt{M \cdot N}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n)\psi_{j,k,l}^{i=H,V,D}(m,n) \tag{7}$$

where ϕ is the approximation function, and ψ is the detail function, W_ϕ, W_ψ are called approximation and detail coefficients. *f(m,n)* is the image feature, *m* and *n* represent the position of the image, while *k* and *l* represent the position of the wavelet. *M* and *N* represent the feature dimensions, while *H*, *V*, and *D* are the horizontal, vertical, and diagonal detail coefficients. In our algorithm, *j = 1,2* and *j0 = 2*.

After performing the 2nd order decomposition, we reconstruct the image features, but only using the 2nd order wavelet subbands. This method pools the image features by a factor of 2 using the inverse DWT (IDWT) (Chui, 1992; Strang & Strela, 1995; Rieder et al., 1994):

$$\begin{aligned} f(m,n) = \frac{1}{\sqrt{M \cdot N}} \sum_m \sum_n W_\phi(j_0,k,l)\phi_{j_0,k,l}(m,n) \quad + \\ \frac{1}{\sqrt{M \cdot N}} \sum_{j=j_0}^{\infty} \sum_m \sum_n W_\psi^i(j,k,l)\psi_{j,k,l}^i(m,n), \quad j = j_0 = 2 \end{aligned} \tag{8}$$

Figure 4 gives an illustration of the algorithm for the forward propagation of wavelet pooling:

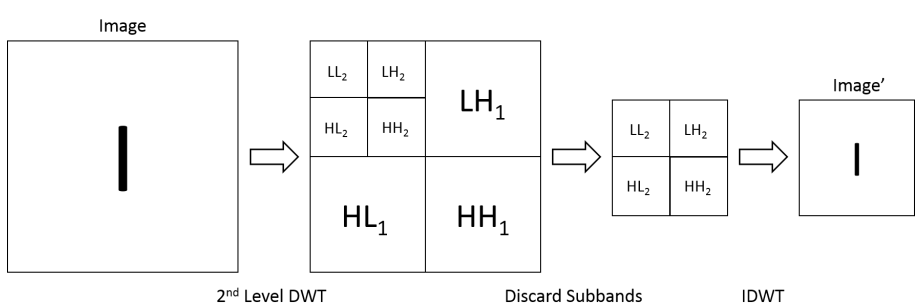


Figure 4: Wavelet Pooling Forward Propagation Algorithm

3.2 Backpropagation

The proposed wavelet pooling algorithm performs backpropagation by reversing the process of its forward propagation. First, the image feature being back propagated undergoes 1st order wavelet decomposition. After decomposition, the detail coefficient subbands upsample by a factor of 2

to create a new 1st level decomposition. The initial decomposition then becomes the 2nd level decomposition. Finally, this new 2nd order wavelet decomposition reconstructs the image feature for further backpropagation using the IDWT. Figure 5 details the backpropagation algorithm of wavelet pooling:

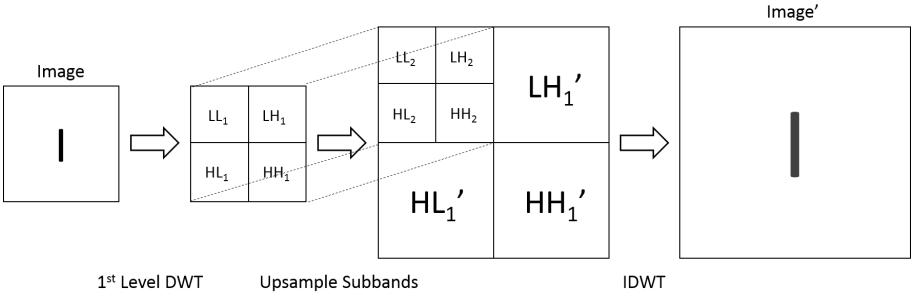


Figure 5: Wavelet Pooling Backpropagation Algorithm

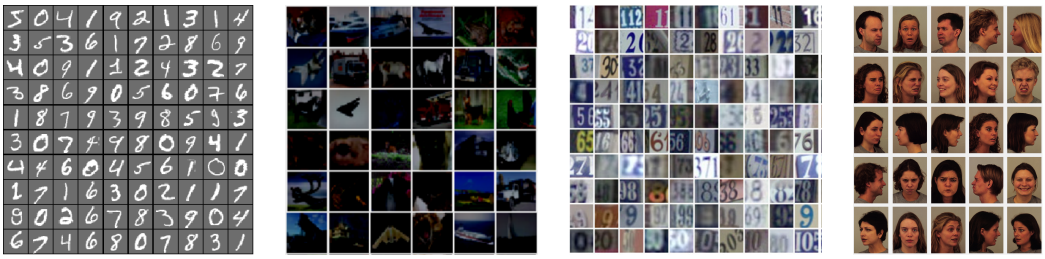
4 Results and Discussion

All CNN experiments use MatConvNet (Vedaldi & Lenc, 2015). All training uses stochastic gradient descent (Bottou, 2010). For our proposed method, the wavelet basis is the Haar wavelet, mainly for its even, square subbands. All experiments are run on a 64-bit operating system, with an Intel Core i7-6800k CPU @ 3.40 GHz processor, with 64.0 GB of RAM. We utilize two GeForce Titan X Pascal GPUs with 12 GB of video memory for all training. All CNN structures except for MNIST use a network loosely based on Zeilers network (Zeiler & Fergus, 2013). We repeat the experiments with Dropout (Srivastava, 2013) and replace Local Response Normalization (Krizhevsky, 2009) with Batch Normalization (Ioffe & Szegedy, 2015) for CIFAR-10 and SHVN (Dropout only) to examine how these regularization techniques change the pooling results. To test the effectiveness of each pooling method on each dataset, we solely pool with that method for all pooling layers in that network. All pooling methods use a 2x2 window for an even comparison to the proposed method. Figure 6 gives a selection of each of the datasets:

4.1 MNIST

The network architecture is based on the example MNIST structure from MatConvNet, with batch normalization inserted. All other parameters are the same. Figure 7 shows our network structure for the MNIST experiments:

The input training data and test data come from the MNIST database of handwritten digits. The full training set of 60,000 images is used, as well as the full testing set of 10,000 images. Our proposed method outperforms all methods. Given the small number of epochs, max pooling is the only method to start to overfit the data during training. Mixed and stochastic pooling show a rocky



(a) MNIST (b) CIFAR-10 (c) SHVN (d) KDEF

Figure 6: Selection of Image Datasets

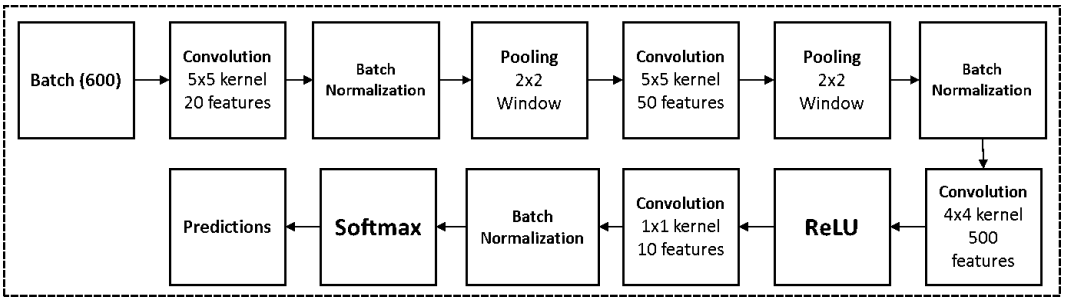


Figure 7: CNN MNIST Structure Block Diagram

trajectory, but do not overfit. Average and wavelet pooling show a smoother descent in learning and error reduction. Figure 8 shows the energy of each method per epoch.

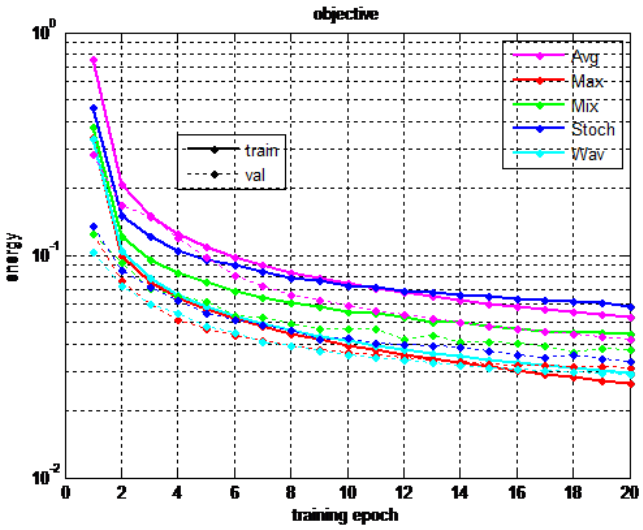


Figure 8: MNIST Pooling Method Energy Performance

Table 1 shows the accuracy of each method:

	Average	Max	Mixed	Stochastic	Wavelet
Accuracy (%)	98.72	98.80	98.86	98.90	**99.01**

Table 1: MNIST Performance of Pooling Methods

4.2 CIFAR-10

We run two sets of experiments with the pooling methods. The first is a regular network structure with no dropout layers. We use this network to observe each pooling method without extra regularization. The second uses dropout and batch normalization, and performs over 30 more epochs to observe the effects of these changes. Figure 9 shows our network structure for the CIFAR-10 experiments:

The input training and test data come from the CIFAR-10 dataset. The full training set of 50,000 images is used, as well as the full testing set of 10,000 images. For the case with no dropout, our proposed method has the second highest accuracy. Max pooling overfits fairly quickly, while wavelet pooling resists overfitting. The change in learning rate prevents our method from overfitting,

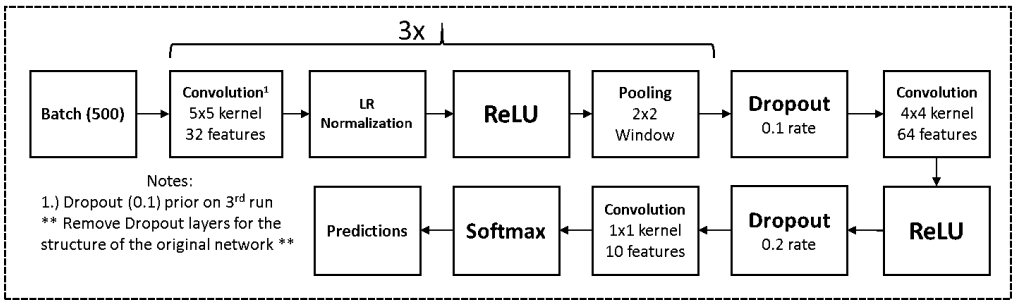


Figure 9: CNN CIFAR-10 Structure Block Diagram

and it continues to show a slower propensity for learning. Mixed and stochastic pooling maintain a consistent progression of learning, and their validation sets trend at a similar, but better rate than our proposed method. Average pooling shows the smoothest descent in learning and error reduction, especially in the validation set. Figure 10 shows the energy of each method per epoch.

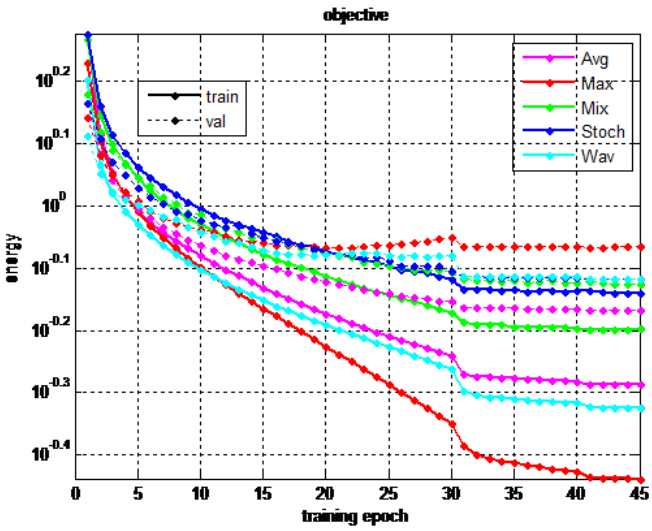


Figure 10: CIFAR-10 Pooling Method Energy Performance

Table 2 shows the accuracy of each method:

	Average	Max	Mixed	Stochastic	Wavelet
Accuracy (%)	**76.51**	71.42	73.77	73.03	74.42

Table 2: CIFAR-10 Performance of Pooling Methods

Table 3 shows the accuracy of each method plus dropout:

	Average	Max	Mixed	Stochastic	Wavelet
Accuracy (%)	**81.15**	80.30	79.21	80.09	80.28

Table 3: CIFAR-10 Performance of Pooling Methods + Dropout

4.3 SHVN

We run two sets of experiments with the pooling methods. The first is a regular network structure with no dropout layers. We use this network to observe each pooling method without extra regularization. The second uses dropout to observe the effects of this change. Figure 11 shows our network structure for the SHVN experiments:

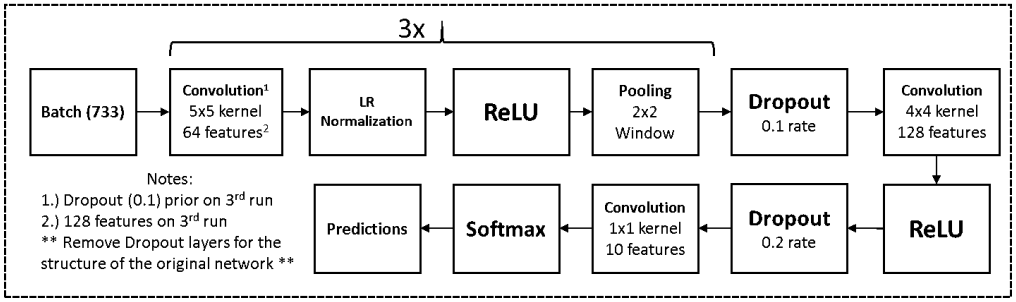


Figure 11: CNN SHVN Structure Block Diagram

The input training and test data come from the SHVN dataset. For the case with no dropout, we use 55,000 images from the training set. For the case with dropout, we use the full training set of 73,257 images, a validation set of 30,000 images we extract from the extra training set of 531,131 images, as well as the full testing set of 26,032 images. For the case with no dropout, our proposed method has the second lowest accuracy. Max and wavelet pooling both slightly overfit the data. Our method follows the path of max pooling, but performs slightly better in maintaining some stability. Mixed, stochastic, and average pooling maintain a slow progression of learning, and their validation sets trend at near identical rates. Figure 12 shows the energy of each method per epoch.

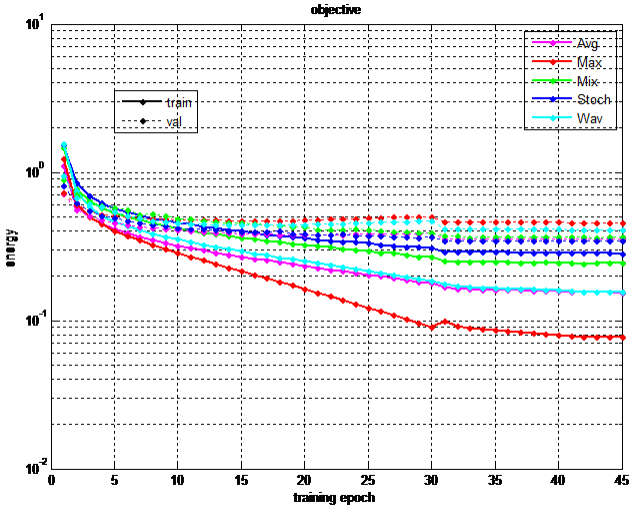


Figure 12: SHVN Pooling Method Energy Performance

Table 4 shows the accuracy of each method:

	Average	Max	Mixed	Stochastic	Wavelet
Accuracy (%)	89.83	88.09	89.25	**89.97**	88.51

Table 4: SHVN Performance of Pooling Methods

Table 5 shows the accuracy of each method plus dropout:

	Average	Max	Mixed	Stochastic	Wavelet
Accuracy (%)	**92.80**	92.18	92.13	91.04	91.10

Table 5: SHVN Performance of Pooling Methods + Dropout

4.4 KDEF

We run one set of experiments with the pooling methods that includes dropout. Figure 13 shows our network structure for the KDEF experiments:

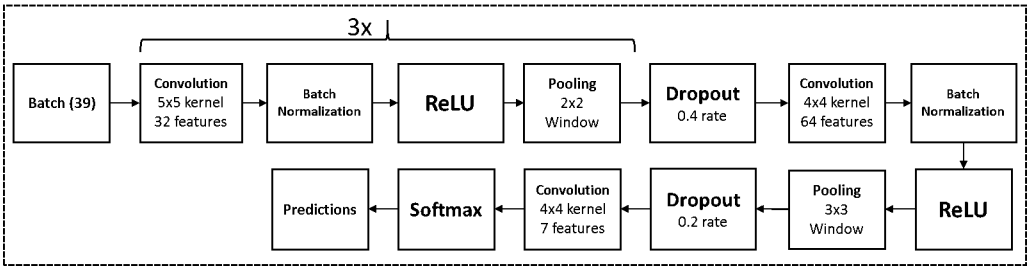


Figure 13: CNN KDEF Structure Block Diagram

The input training and test data come from the KDEF dataset. This dataset contains 4,900 images of 35 people displaying seven basic emotions (afraid, angry, disgusted, happy, neutral, sad, and surprised) using facial expressions. They display emotions at five poses (full left and right profiles, half left and right profiles, and straight).

This dataset contains a few errors that we fix (missing or corrupted images, uncropped images, etc.). All of the missing images are at angles of -90, -45, 45, or 90 degrees. We fix the missing and corrupt images by mirroring their counterparts in MATLAB and adding them back to the dataset. We manually crop the images that need to match the dimensions set by the creators (762 x 562). KDEF does not designate a training or test data set. We shuffle the data and separate 3,900 images as training data, and 1,000 images as test data. We resize the images to 128x128 because of memory and time constraints.

The dropout layers regulate the network and maintain stability in spite of some pooling methods known to overfit. Max pooling eventually overfits, while wavelet pooling resists overfitting. Average and mixed pooling resist overfitting, but are unstable for most of the learning. Stochastic pooling maintains a consistent progression of learning. Wavelet pooling also follows a smoother, consistent progression of learning. Figure 14 shows the energy of each method per epoch.

Table 6 shows the accuracy of each method:

	Average	Max	Mixed	Stochastic	Wavelet
Accuracy (%)	**76.5**	75.6	72.6	72.7	75.9

Table 6: KDEF Performance of Pooling Methods

5 Conclusion

We prove wavelet pooling has potential to equal or eclipse some of the traditional methods currently utilized in CNNs. Our proposed method outperforms all others in the MNIST dataset, outperforms all but one in the CIFAR-10 and KDEF datasets, and performs within respectable ranges of the pooling methods that outdo it in the SHVN dataset. The addition of dropout and batch normalization show our proposed methods response to network regularization. Like the non-dropout cases, it outperforms all but one in both the CIFAR-10 & KDEF datasets, and performs within respectable ranges of the pooling methods that outdo it in the SHVN dataset. Our results confirm previous studies proving that no one pooling method is superior, but some perform better than others depending

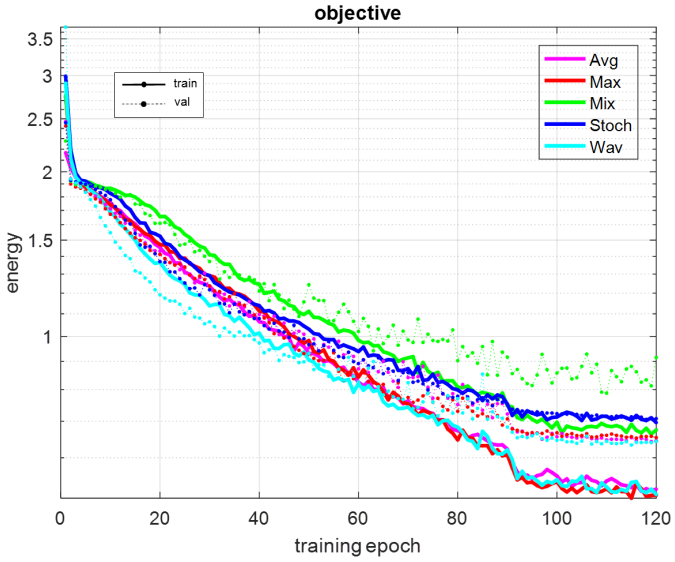


Figure 14: KDEF Pooling Method Energy Performance

on the dataset and network structure Boureau et al. (2010); Lee et al. (2016). Furthermore, many networks alternate between different pooling methods to maximize the effectiveness of each method.

Future work and improvements in this area could be to vary the wavelet basis to explore which basis performs best for the pooling. Altering the upsampling and downsampling factors in the decomposition and reconstruction can lead to better image feature reductions outside of the 2x2 scale. Retention of the subbands we discard for the backpropagation could lead to higher accuracies and fewer errors. Finally, analyzing the structural similarity (SSIM) of wavelet pooling versus other methods could further prove the vitality of using our approach.

ACKNOWLEDGMENTS

This research is supported by the Title III HBGI PhD Fellowship grant from the U.S. Department of Education.

REFERENCES

Léon Bottou. Large–scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT 2010*, pp. 177–186. Springer, 2010.

Y-Lan Boureau, Jean Ponce, and Yann Lecun. A theoretical analysis of feature pooling in visual recognition. In *27TH INTERNATIONAL CONFERENCE ON MACHINE LEARNING, HAIFA, ISRAEL*, 2010.

C. K. Chui. *An Introduction to Wavelets*. New York: Academic Press, 1992.

Emil Dumic, Sonja Grgic, and Mislav Grgic. The use of wavelets in image interpolation: possibilities and limitations. *RADIOENGINEERING-PRAGUE-*, 16(4):101, 2007.

K Fukushima. Neural network model for a mechanism of pattern recognition unaffected by shift in position Neocognitron. *Trans. IECE*, J62-A(10):658–665, 1979.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pp. 2278–2324, 1998.

Chen-Yu Lee, Patrick W. Gallagher, and Zhuowen Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 464–472, 2016.

Daniel Lundqvist, Anders Flykt, and Arne Öhman. The karolinska directed emotional faces (kdef). *CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet*, 1998.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

J. Anthony Parker, Robert V. Kenyon, and Donald E. Troxel. Comparison of interpolating methods for image resampling. *IEEE Transactions on Medical Imaging*, 2(1):31–39, 1983.

Peter Rieder, Jiirgen Gotze, and Josef A Nossek. Multiwavelet transforms based on several scaling functions. In *Proceedings of the IEEE–SP International Symposium on Time–Frequency and Time–Scale Analysis*, pp. 393–396. IEEE, 1994.

Nitish Srivastava. Improving neural networks with dropout. Master's thesis, University of Toronto, 2013.

Gilbert Strang and Vasily Strela. Short wavelets and matrix dilation equations. *IEEE Transactions on Signal Processing*, 43(1):108–115, 1995.

Andrea Vedaldi and Karel Lenc. Matconvnet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 689–692. ACM, 2015.

Juyang Weng, Narendra Ahuja, and Thomas S Huang. Cresceptron: a self-organizing neural network which grows adaptively. In *International Joint Conference on Neural Networks (IJCNN)*, volume 1, pp. 576–581. IEEE, 1992.

Travis Williams and Robert Li. Advanced image classification using wavelets and convolutional neural networks. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 233–239, 2016.

Dingjun Yu, Hanli Wang, Peiqiu Chen, and Zhihua Wei. *Mixed Pooling for Convolutional Neural Networks*, pp. 364–375. Springer International Publishing, 2014.

Matthew Zeiler and Robert Fergus. Stochastic pooling for regularization of deep convolutional neural networks. In *Proceedings of the International Conference on Learning Representation (ICLR)*, 2013.