Understanding and Improving CNNs via Concatenated Rectified Linear Units

Wendy Shang, Kihyuk Sohn, Diogo Almeida and Honglak Lee

Oculus VR, NEC Labs, Enlitics, University of Michigan

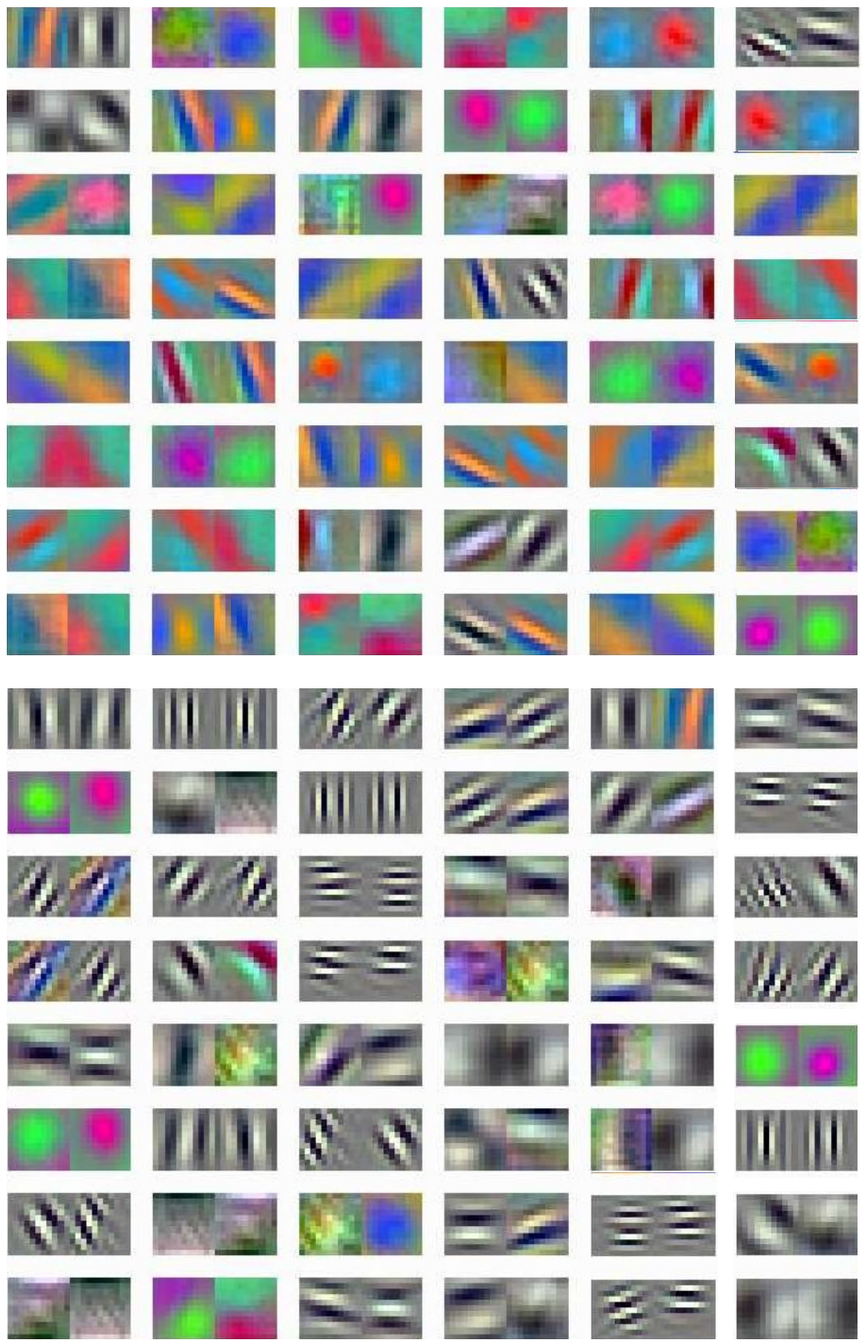
Speaker: Tiberio Uricchio

MICC
Media Integration and Communication Center

Outline

- Motivation of the work
- Concatenated Rectified Linear Units (CReLU)
- Experiments
- Discussion of CReLU properties
- Conclusion

Motivation

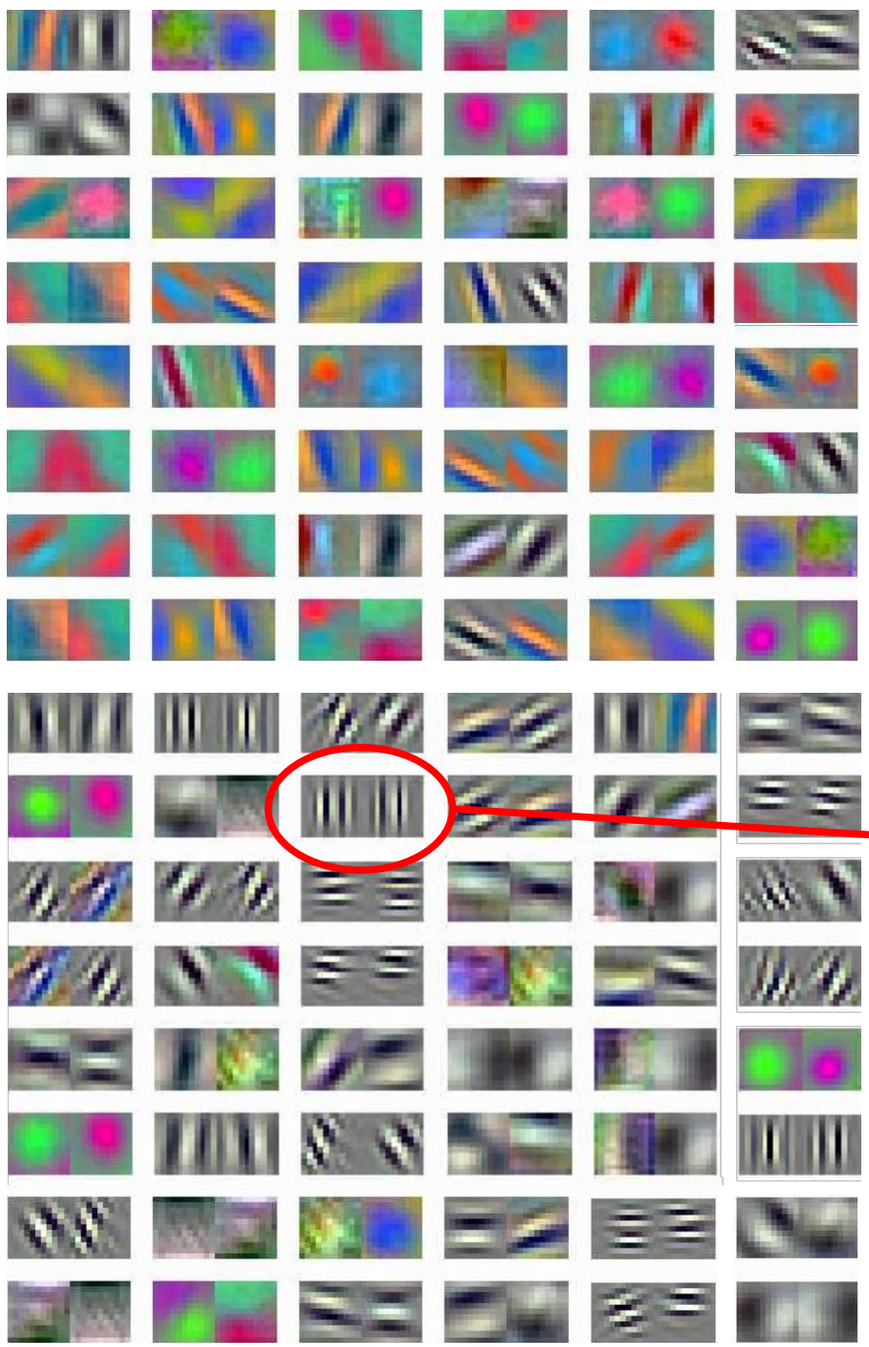


Conv1 filters of AlexNet

- **Observation:** a trained AlexNet have pairs of filters that have opposite phase to each other

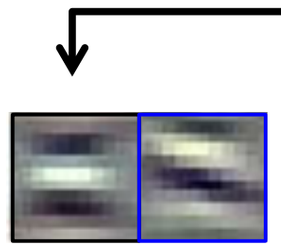
- Natural consequence of using ReLU nonlinearity.

Motivation



Conv1 filters of AlexNet

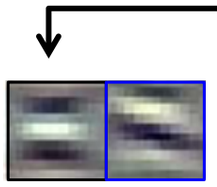
- **Observation:** a trained AlexNet have pairs of filters that have opposite phase to each other.
- Natural consequence of using ReLU nonlinearity.



reference filter: $\phi_i = \dfrac{w_i}{\|w_i\|_2}$

pairing filter: $\bar{\phi}_i = \arg\min_{j \neq i} \langle \phi_i, \phi_j \rangle$

Quantitative study



reference filter: ϕ_i

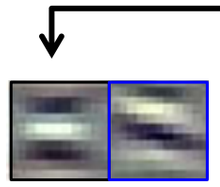
pairing filter: $\bar{\phi}_i = \arg\min_{j \neq i} \langle \phi_i, \phi_j \rangle$

$$\phi_i = \frac{w_i}{\|w_i\|_2}$$

$$\mu_i^{\{\phi\}} = \langle \phi_i, \bar{\phi}_i \rangle$$

measure correlation
[-1, 1]

Quantitative study

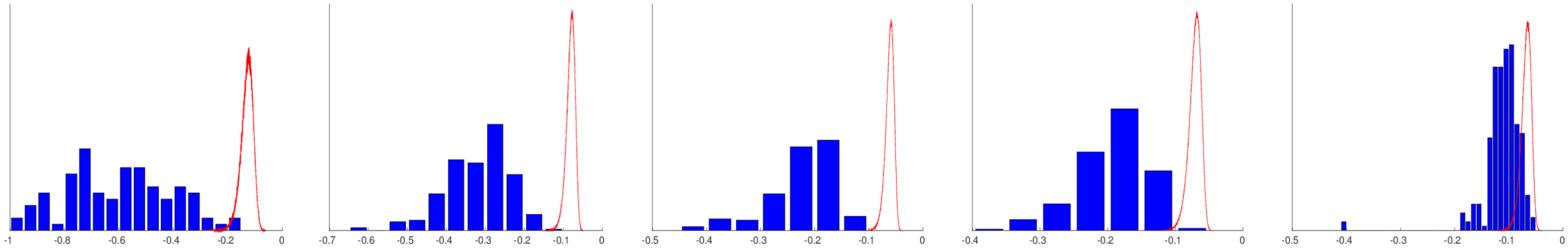


reference filter: ϕ_i

pairing filter: $\bar{\phi}_i = \arg\min_{j \neq i} \langle \phi_i, \phi_j \rangle$

$$\mu_i^{\{\phi\}} = \langle \phi_i, \bar{\phi}_i \rangle$$

measure correlation
[-1, 1]



(a) conv1 (b) conv2 (c) conv3 (d) conv4 (e) conv5

- Blue: histogram of AlexNet filters, Red: histogram of random Gaussian filters
- Pairing filters of AlexNet are highly negatively-correlated.
- Gradually shifts towards 0 when going deeper.

CReLU

- **Hypothesis:** lower convolution layers learn redundant filters to extract *both positive and negative phase information* of an input signal.

CReLU

- **Hypothesis:** lower convolution layers learn redundant filters to extract *both positive and negative phase information* of an input signal.

- **Proposal**: use a different activation scheme that accounts for both phases.

CReLU activation, denoted by $\rho_c : \mathbb{R} \to \mathbb{R}^2$, is defined as follows:

$$\rho_c(x) \triangleq (\max(0, x), \max(0, -x))$$

CReLU

- Absolute Value Rectification (AVR) is the most similar activation function previously proposed in the literature.

AVR: $$\rho_{\text{AVR}}(x) \triangleq \max(0, x) + \max(0, -x)$$

CReLU: $$\rho_c(x) \triangleq (\max(0, x), \max(0, -x))$$

- Differently to ReLU or AVR, they are explicitly crafted to respond to phase and modulus of responses.

	phase	positive activation	negative activation
ReLU	✓	✓	
AVR		✓	✓
CReLU	✓	✓	✓

Properties of CReLU

- CReLU naturally reduce redundancy of learning separate filters that differ in phase only.

- CReLU is an activation scheme. The element-wise ReLU non- linearity after concatenation can be substituted by other activation functions (e.g., Leaky ReLU).

Experiments

- Perform experiments on object recognition tasks on **CIFAR-10/100** and **ImageNet**.

- Experimentation protocol:

  - Employ **existing network architectures** with ReLU.

  - **Replace** ReLU into CReLU.

  - Hyperparameters (e.g., learning rate) are re-used from baseline ReLU models.

CIFAR-10 and 100

- ConvPool-CNN-C (Springenberg et al. 2014)

	Baseline		Baseline (double)	
Layer	kernel, stride, padding	activation	kernel, stride, padding	activation
conv1	3×3×3×96, 1, 1	Relu	3×3×3×192, 1, 1	Relu
conv2	3×3×96×96, 1, 1	Relu	3×3×192×192, 1, 1	Relu
pool1	3×3, 2, 0	max	3×3, 2, 0	max
conv3	3×3×96×192, 1, 1	Relu	3×3×192×384, 1, 1	Relu
conv4	3×3×192×192, 1, 1	Relu	3×3×384×384, 1, 1	Relu
conv5	3×3×192×192, 1, 1	Relu	3×3×384×384, 1, 1	Relu
pool2	3×3, 2, 0	max	3×3, 2, 0	max
conv6	3×3×192×192, 1, 1	Relu	3×3×384×384, 1, 1	Relu
conv7	1×1×192×192, 1, 1	Relu	1×1×384×384, 1, 1	Relu
conv8	1×1×192×10/100, 1, 0	Relu	1×1×384×10/100, 1, 0	Relu
pool3	10×10 (100 for CIFAR-100)	avg	10×10 (100 for CIFAR-100)	avg

	CRelu		CRelu (half)	
Layer	kernel, stride, padding	activation	kernel, stride, padding	activation
conv1	3×3×3×96, 1, 1	CRelu	3×3×3×48, 1, 1	CRelu
conv2	3×3×192×96, 1, 1	CRelu	3×3×96×48, 1, 1	CRelu
pool1	3×3, 2, 0	max	3×3, 2, 0	max
conv3	3×3×192×192, 1, 1	CRelu	3×3×96×48, 1, 1	CRelu
conv4	3×3×384×192, 1, 1	CRelu	3×3×96×96, 1, 1	CRelu
conv5	3×3×384×192, 1, 1	CRelu	3×3×192×96, 1, 1	CRelu
pool2	3×3, 2, 0	max	3×3, 2, 0	max
conv6	3×3×384×192, 1, 1	CRelu	3×3×192×96, 1, 1	CRelu
conv7	1×1×384×192, 1, 1	CRelu	1×1×192×96, 1, 1	CRelu
conv8	1×1×384×10/100, 1, 0	Relu	1×1×192×10/100, 1, 0	Relu
pool3	10×10 (100 for CIFAR-100)	avg	10×10 (100 for CIFAR-100)	avg

CIFAR-10 and 100

- ConvPool-CNN-C (Springenberg et al. 2014)

	Baseline		Baseline (double)	
Layer	kernel, stride, padding	activation	kernel, stride, padding	activation
conv1	3×3×3×96, 1, 1	Relu	3×3×3×192, 1, 1	Relu
conv2	3×3×96×96, 1, 1	Relu	3×3×192×192, 1, 1	Relu
pool1	3×3, 2, 0	max	3×3, 2, 0	max
conv3	3×3×96×192, 1, 1	Relu	3×3×192×384, 1, 1	Relu
conv4	3×3×192×192, 1, 1	Relu	3×3×384×384, 1, 1	Relu
conv5	3×3×192×192, 1, 1	Relu	3×3×384×384, 1, 1	Relu
pool2	3×3, 2, 0	max	3×3, 2, 0	max
conv6	3×3×192×192, 1, 1	Relu	3×3×384×384, 1, 1	Relu
conv7	1×1×192×192, 1, 1	Relu	1×1×384×384, 1, 1	Relu
conv8	1×1×192×10/100, 1, 0	Relu	1×1×384×10/100, 1, 0	Relu
pool3	10×10 (100 for CIFAR-100)	avg	10×10 (100 for CIFAR-100)	avg

	CRelu		CRelu (half)	
Layer	kernel, stride, padding	activation	kernel, stride, padding	activation
conv1	3×3×3×96, 1, 1	CRelu	3×3×3×48, 1, 1	CRelu
conv2	3×3×192×96, 1, 1	CRelu	3×3×96×48, 1, 1	CRelu
pool1	3×3, 2, 0	max	3×3, 2, 0	max
conv3	3×3×192×192, 1, 1	CRelu	3×3×96×48, 1, 1	CRelu
conv4	3×3×384×192, 1, 1	CRelu	3×3×96×96, 1, 1	CRelu
conv5	3×3×384×192, 1, 1	CRelu	3×3×192×96, 1, 1	CRelu
pool2	3×3, 2, 0	max	3×3, 2, 0	max
conv6	3×3×384×192, 1, 1	CRelu	3×3×192×96, 1, 1	CRelu
conv7	1×1×384×192, 1, 1	CRelu	1×1×192×96, 1, 1	CRelu
conv8	1×1×384×10/100, 1, 0	Relu	1×1×192×10/100, 1, 0	Relu
pool3	10×10 (100 for CIFAR-100)	avg	10×10 (100 for CIFAR-100)	avg

Results on CIFAR 10/100

Model	CIFAR-10		CIFAR-100		params.
	Average	Vote	Average	Vote	
ReLU	10.20±0.09	7.55	38.52±0.12	31.26	1.4M
+(double)	9.87±0.09	7.28	36.73±0.15	28.34	5.6M
AVR	10.26±0.10	7.76	37.24±0.20	29.77	1.4M
CReLU	**9.39**±0.11	**7.09**	**33.76**±0.12	**27.60**	2.8M
+(half)	**9.44**±0.09	**7.09**	36.20±0.18	29.93	0.7M

Test set recognition error with baseline network architecture.

- CReLU models outperform the ReLU and AVR models.
- CReLU with reduced model capacity still outperforms the baseline ReLU model.

Results on CIFAR 10/100

Model		CIFAR-10		CIFAR-100	
		Average	Vote	Average	Vote
ReLU		6.90±0.03	5.43	30.27±0.09	26.85
CReLU	conv1	**6.45**±0.05	5.22	28.43±0.11	24.67
CReLU	conv1,3	**6.45**±0.05	**5.09**	27.79±0.08	23.93
CReLU	conv1,3,5	**6.45**±0.05	5.16	**27.67**±0.07	**23.66**

Test set recognition error with deeper network architecture.

- VGG-style network architecture involves 13 conv. + 2 fully-connected layers.
- Replacing ReLU into CReLU gradually from lower layers to higher layers improves the performance.

Results on CIFAR 10/100

Model	CIFAR-10	CIFAR-100
Rippel et al., 2015	8.60	31.60
Snoek et al., 2015	6.37	27.40
Liang and Hu, 2015	7.09	31.75
Lee et al., 2016	6.05	32.37
Srivastava et al., 2015	7.60	32.24
VGG	5.43	26.85
VGG with CReLU	**5.09**	**23.66**

Test set recognition error with deeper network architecture (VGG).

- Achieved lower error rates to other state-of-the-art.

ImageNet

- All-CNN-B model [2]

Layer	kernel	activation
conv1	11×11×3×96	ReLU/AVR
conv2	1×1×96×96	ReLU/AVR
conv3	3×3×96×96	ReLU/AVR
conv4	5×5×96×256	ReLU/AVR
conv5	1×1×256×256	ReLU/AVR
conv6	3×3×256×256	ReLU/AVR
conv7	3×3×256×384	ReLU/AVR
conv8	1×1×384×384	ReLU/AVR
conv9	3×3×384×384	ReLU/AVR
conv10	3×3×384×1024	ReLU/AVR
conv11	1×1×1024×1024	ReLU/AVR
conv12	1×1×1024×1000	Linear
pool	6×6	average

Baseline CNN (ReLU, AVR)
# of parameters: 9.37M

Layer	kernel	activation
conv1	11×11×3×96	CReLU
conv2	1×1×192×96	CReLU
conv3	3×3×192×96	CReLU
conv4	5×5×192×256	CReLU
conv5	1×1×512×256	ReLU
conv6	...	

CReLU (conv1-4): 10.14M

Layer	kernel	activation
conv1	11×11×3×96	CReLU
conv2	1×1×192×96	CReLU
conv3	3×3×192×96	CReLU
conv4	5×5×192×256	CReLU
conv5	1×1×512×256	CReLU
conv6	3×3×512×256	CReLU
conv7	3×3×512×384	CReLU
conv8	1×1×768×384	ReLU
conv9	...	

CReLU (conv1-7): 11.62M

- [2] Springenberg et al. Striving for simplicity: The all convolutional net. In ICLR Workshop, 2014.

Results on ImageNet

Model		top-1	top-5	top-1†	top-5†	params.
ReLU		41.81	19.74	38.03	17.17	9.37M
AVR	(conv1–4)	41.12	19.25	37.32	16.49	
AVR	(conv1–7)	42.36	20.05	38.21	17.42	
CReLU	(all, half)	40.93	19.39	37.28	16.72	4.68M
CReLU	(conv1–4)	**39.82**	**18.28**	36.20	15.72	10.14M
CReLU	(conv1–7)	39.97	18.33	36.53	16.01	11.62M
CReLU	(conv1,4,7, half)	40.45	18.58	**35.70**	**15.32**	8.60M

- Test set recognition error. † are obtained by averaging scores from 10 patches.
- CReLU improves the classification accuracy.
- Going deeper with CReLU isn’t necessarily beneficial.

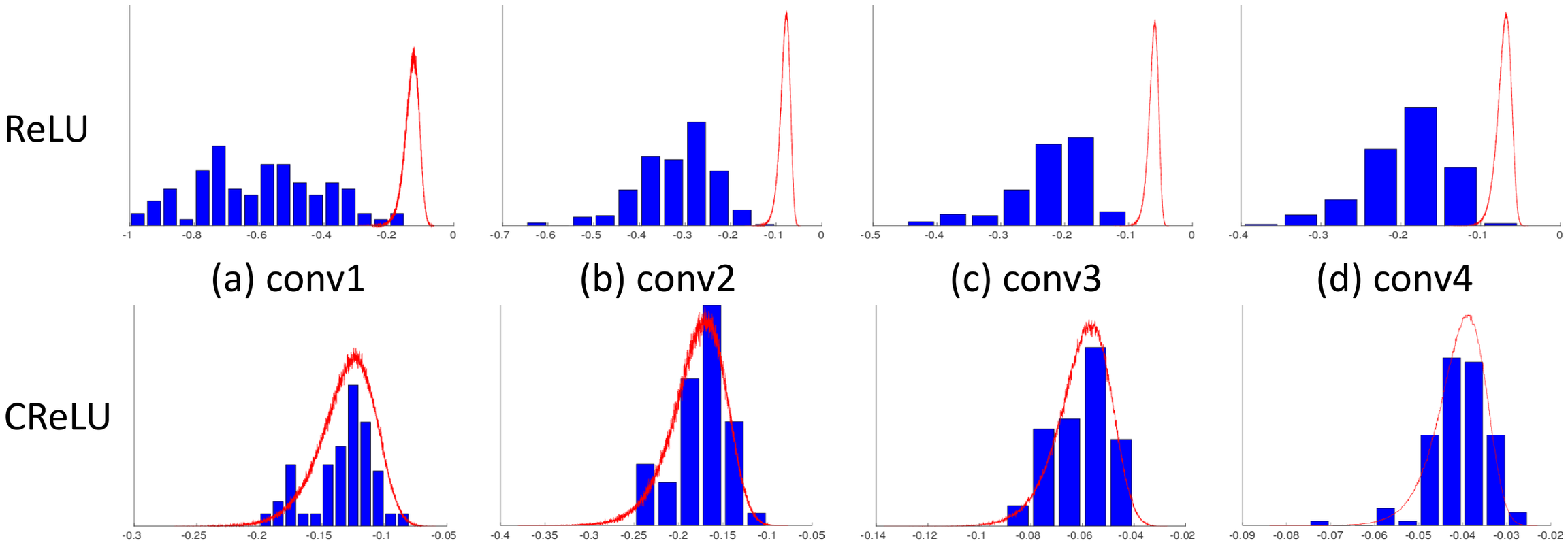
Results on ImageNet

Model	top-1	top-5	top-1†	top-5†	params.
AlexNet	42.6	19.6	40.7	18.2	61M
FriedNet [4]	41.93	–	–	–	32.8M
PrunedNet [5]	42.77	19.67	–	–	6.7M
AllConvB	41.81	19.74	38.03	17.17	9.37M
CReLU (all, half)	40.93	19.39	37.28	16.72	**4.68M**
CReLU (conv1,4,7, half)	**40.45**	**18.58**	**35.70**	**15.32**	8.60M

- CReLU model achieves highly competitive results while leveraging a small number of model parameters.

Pairing Disappears with CReLU

- Negative correlations of pairing filters



Blue: histogram of CReLU filters, Red: histogram of random Gaussian filters

- Histograms of CReLU model well align with the distributions of random Gaussian filters.

Discussion: regularization

- Empirical evidence: train vs. test errors

Model	CIFAR-10		CIFAR-100		params.
	train	test	train	test	
ReLU	1.09	9.17	13.68	36.30	1.4M
CReLU	4.23	8.43	14.25	31.48	2.8M

  - CReLU model shows higher train error even though it contains more trainable parameters.

Discussion: regularization

- Empirical evidence: train vs. test errors

Model	CIFAR-10		CIFAR-100		params.
	train	test	train	test	
ReLU	1.09	9.17	13.68	36.30	1.4M
CReLU	4.23	8.43	14.25	31.48	2.8M

  - CReLU model shows higher train error even though it contains more trainable parameters.

- Rademacher Complexity Bound

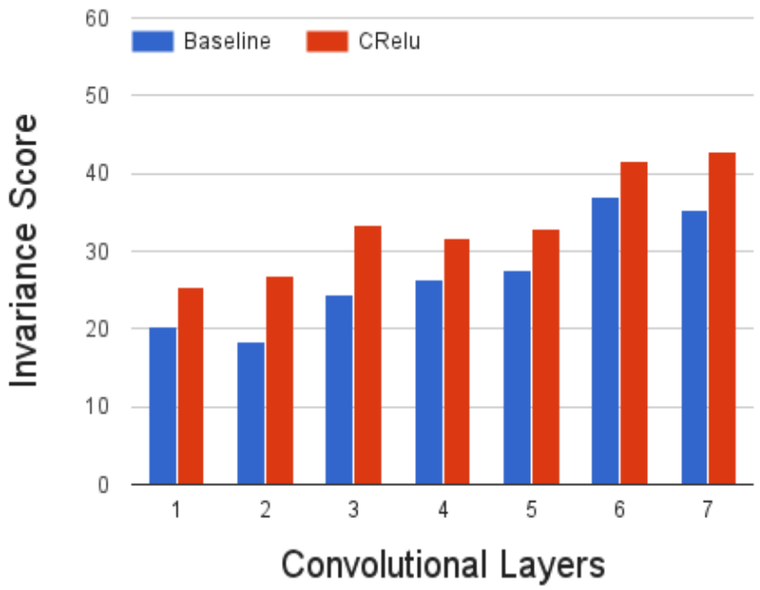
Theorem. Let $\mathcal{G}$ be the class of real functions $\mathbb{R}^{d_{in}} \to \mathbb{R}$ with input dimension $\mathcal{F}$, that is, $\mathcal{G} = [\mathcal{F}]_{j=1}^{d_{in}}$. Let $\mathcal{H}$ be a linear transformation function $\mathbb{R}^{2d_{in}} \to \mathbb{R}$, parametrized by W, where $\|W\|_2 \leq B$. Then,

$$\hat{R}_L(\mathcal{H} \circ \rho_c \circ \mathcal{G}) \leq \sqrt{d_{in}} B \hat{R}_L(\mathcal{F})$$

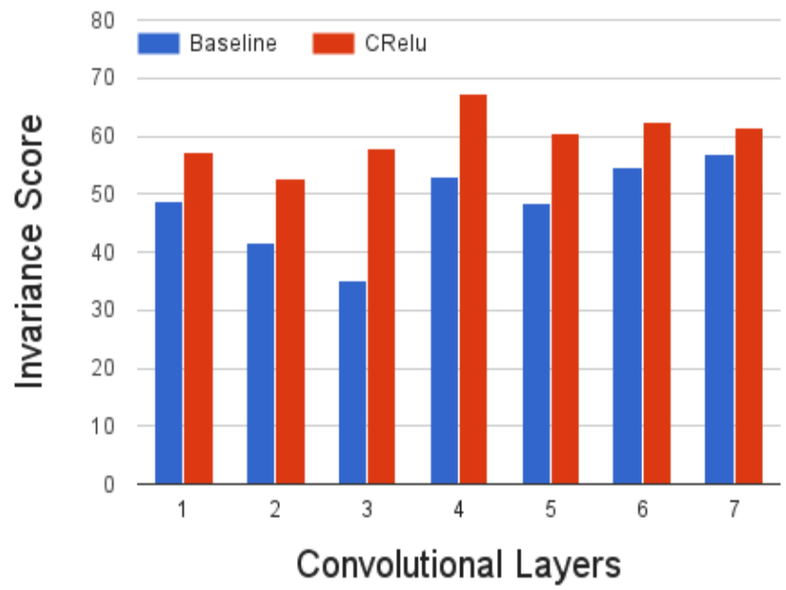
  - RCB of CReLU model is the same as that of ReLU model [6].

[6] Wan et al., Regularization of neural networks using drop-connect. In ICML, 2013.

Discussion: invariance



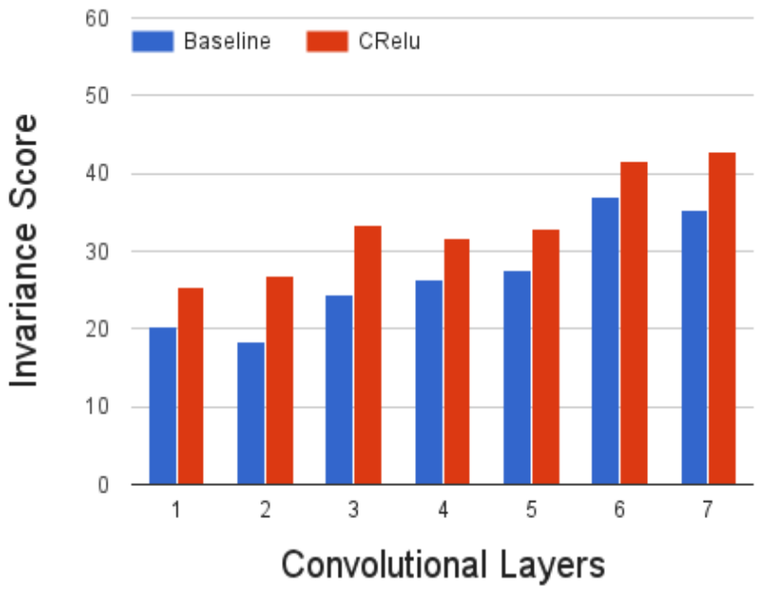
(a) CIFAR-10



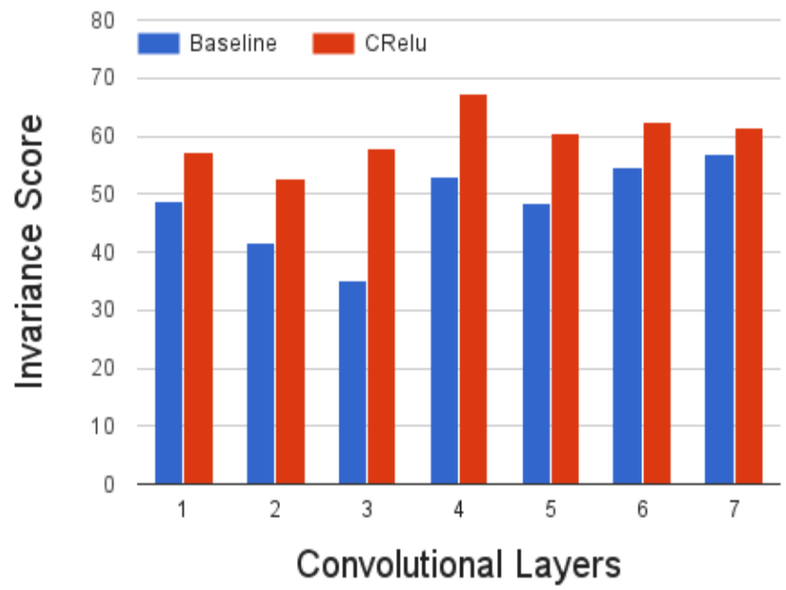
(b) ImageNet

- Invariant representation is desirable for high-level vision tasks.
- CReLU models show higher invariance score than ReLU models.

Discussion: invariance



(a) CIFAR-10



(b) ImageNet

- Invariant representation is desirable for high-level vision tasks.
- CReLU models show higher invariance score than ReLU models.

- Invariance score gives an idea where to place CReLU.

Model		top-1	top-5	top-1†	top-5†
CReLU	(all, half)	40.93	19.39	37.28	16.72
CReLU	(conv1,4,7, half)	40.45	18.58	**35.70**	**15.32**

Discussion: reconstruction property

- CReLU model can be *inverted* for reconstruction

1: $f_{\text{cnn}}(x) \leftarrow$ convolution features, $W \leftarrow$ weight matrix.

2: Obtain the linear responses after convolution by reverting CReLU: $z = \rho_c^{-1}(f_{\text{cnn}}(x))$.

3: Compute the Moore-Penrose pseudoinverse $(W^\top)^+$ of $W^\top$.

4: Obtain reconstruction: $x' = (W^\top)^+ z$.

Reconstruction algorithm of CReLU model (without max-pooling).

(a) input (b) conv1 (c) conv2 (d) conv3 (e) conv4

Reconstruction of CReLU model using different convolution layers.

Discussion: reconstruction property

(a) input (b) conv1 (c) conv2 (d) conv3 (e) conv4

Reconstruction of CReLU model using different convolution layers.

Conclusions

- A new activation scheme, CReLU, which preserves both positive and negative linear responses after convolution is proposed so that each filter can efficiently represent its unique direction.

- CReLU improves deep networks with classification objective.

- CReLU is effective from several viewpoints, such as regularization, invariance, as well as reconstruction property.