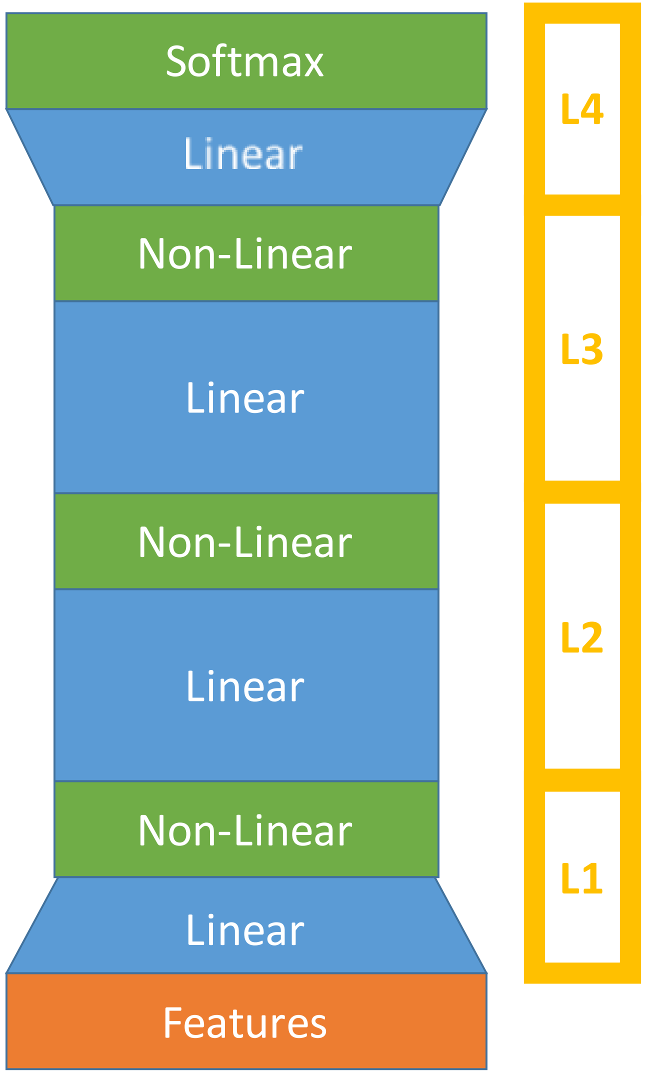
Linearly Augmented Deep Neural Network

Pegah Ghahremani, Johns Hopkins University,

Jasha Droppo, Michael L. Seltzer, Microsoft Research

Typical DNN Architecture

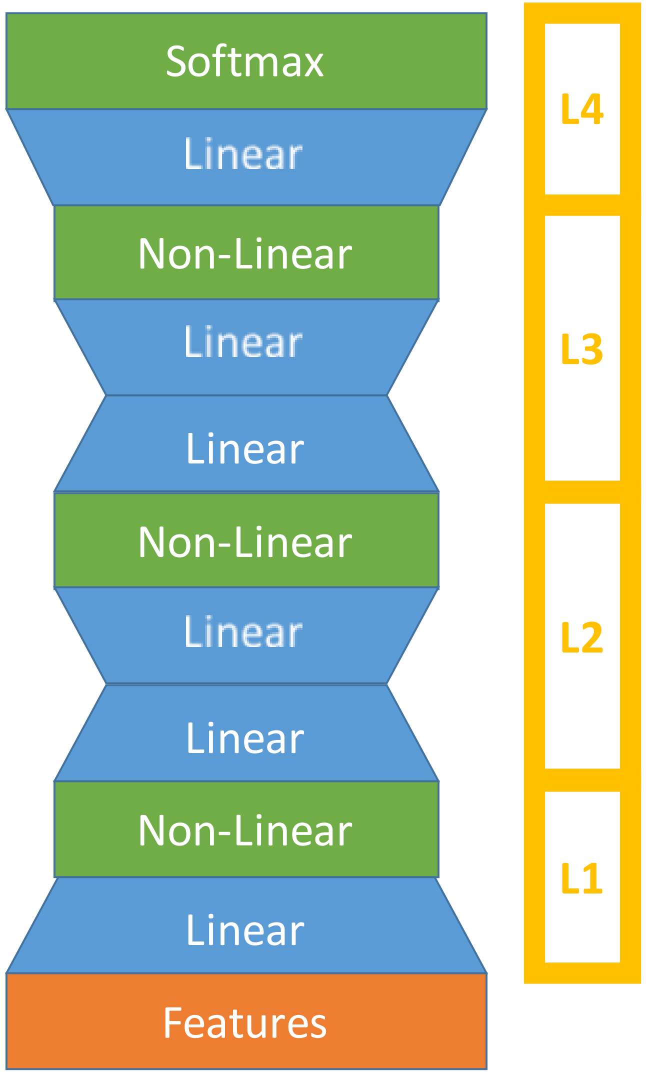


- Layers are a composition of an affine and a non-linear function.

$$f_i(x) = \phi(W_i x + b_i)$$

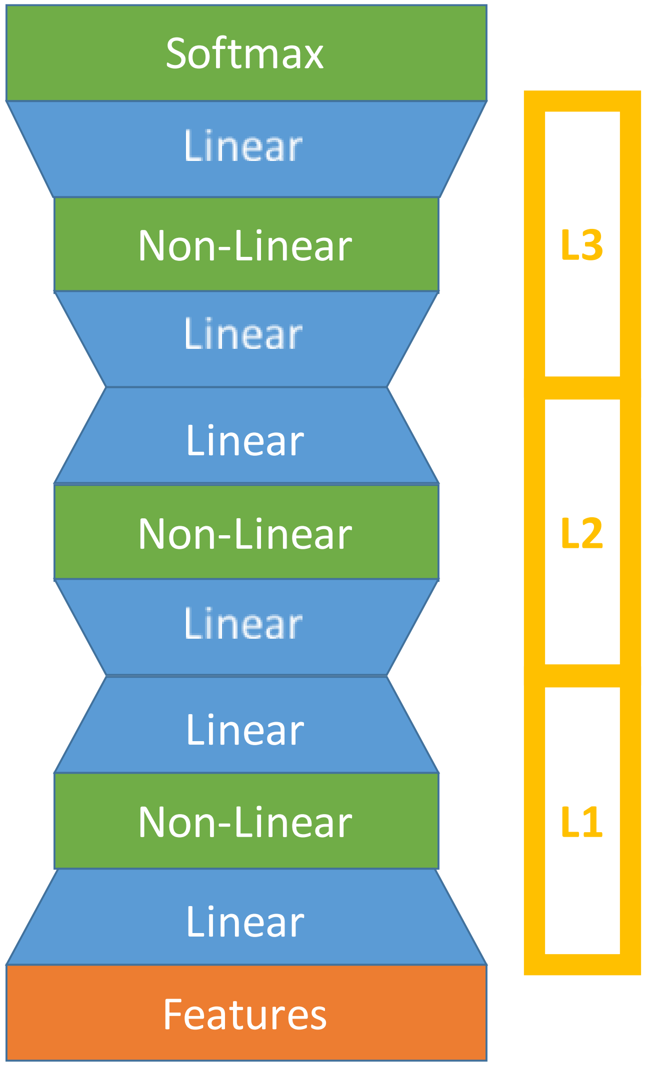
- Typical DNN is a composition of several similar functions.
- Can be trained from random initialization, but pre-training can help.
- Does DNN use all its capacity? Can we reduce the model size?
  - Small amount of neurons are active
  - High memory usage

SVD-DNN Architecture



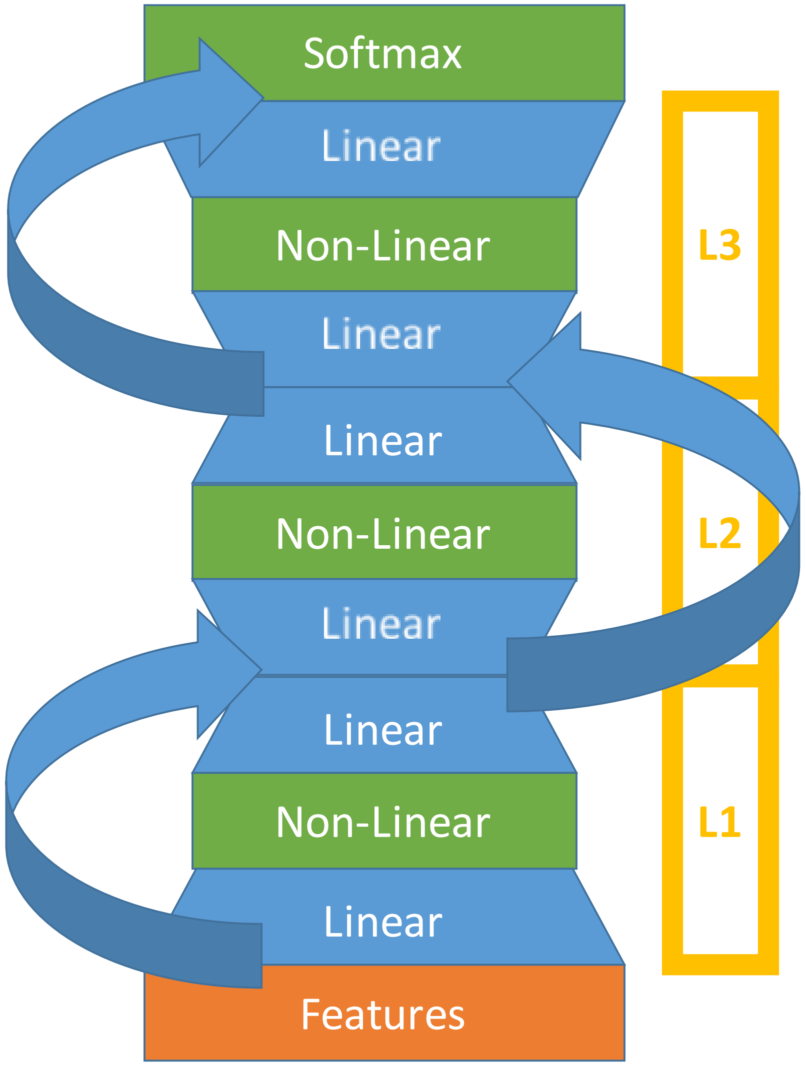
- Alternating hourglass-linear and nonlinear blocks.
- Layer concept is the same as typical DNN.
- Training
  - Train typical DNN.
  - Compress linear transformations with SVD.
  - Fine tune the model to regain the lost accuracy.
- Can not be trained from random initialization.

SVD-DNN Architecture (alternate view)



- Layers are composition of an affine, non-linear, and linear operation.
  $$f_i(x) = V_i \phi(U_i x + b_i)$$
- The SVD-DNN is a composition of several of these layers.
- Each layer:
  - Maps the from one continuous vector space embedding to another.
  - Is general function approximator.
  - Is very inefficient at representing a linear transformation.

LA-DNN Architecture



- Similar to SVD-DNN
- Augment each layer with a linear term.
$$f_i(x) = V_i \phi(U_i x + b_i) + T_i x$$
- These layers:
  - Use T_i to model any linear component of the desired layer transformation.
  - Use $\{V_i, U_i, b_i\}$ to model the non-linear residual.
  - Posses greater modeling power, with a similar parameter count.

Network Type Comparison

	Train from Random	Pre-training	Compressed	Notes
Typical DNN	Yes	Available	No	Vanishing gradients Over-parameterized Large Model Unused Capacity
SVD-DNN	No	Required	Yes	DNN approximation Smaller Model Difficult to train
LA-DNN	Yes	Un-necessary	Yes	

LA-DNN Linear Component Parameters

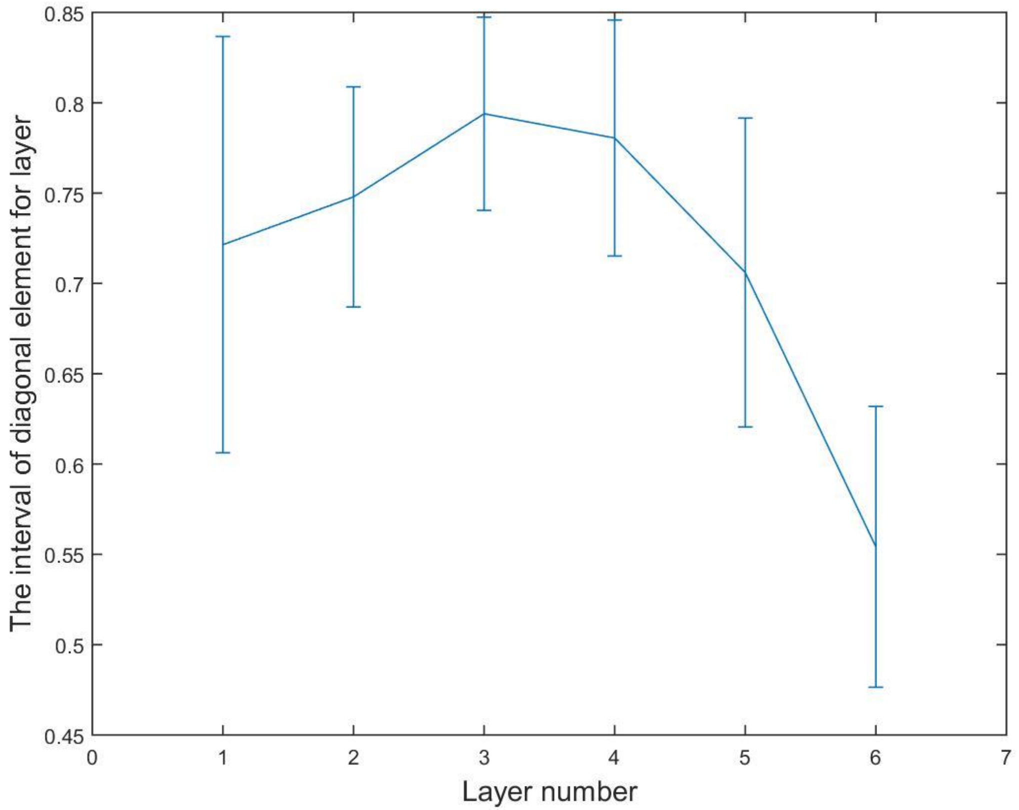
- Recall the formula for the LA-DNN layer:

$$f_i(x) = V_i \phi(U_i x + b_i) + T_i x$$

- The matrix T_i can be Identity matrix.
  - Fewest number of parameters, least flexible.
- It can be a full matrix.
  - Most flexible, but increases parameter count considerably.
- It can be a diagonal matrix.
  - Balance between flexibility and parameter count.
  - Best configuration in our experiments.

LA-DNN Linear Component Values

- Q: How does the network weight the linear component of its transform?
- A: Lower transition weight for higher layers



TIMIT Results (baseline)

- DNN-Sigmoid system size has been tuned to minimize TIMIT PER.
- LA-DNN variants easily beat the tuned DNN system.
  - Better - Improvements in all metrics.
  - Faster - Drastically fewer parameters speeds training and evaluation.
  - Deeper - LA layers are able to benefit from deeper networks structure.

Model	Num of H.Layers	Layers Size	# Params	Training CE	Training Frame Err %	Validation CE	Validation Frame Err	PER %
DNN + Sigmoid	2	2048X2048	10.9M	0.66	21.39	1.23	37.67	23.63
LA-DNN + Sigmoid	6	1024X512	8M	0.61	20.5	1.18	35.8	22.28
LA-DNN+ReLU	6	1024X256	4.5M	0.54	18.6	1.22	35.5	22.08

TIMIT Results (Going Deeper)

- Keeping parameter count well under the baseline (10.9M)
- All metrics continue to improve – to at least forty-eight layers deep.

LA-DNN with ReLU Units							
Num of H.Layers	Layers Size	# Params	Training		Validation		PER %
			Training CE	Training Frame Err %	Validation CE	Validation Frame Err	
3	1024X256	2.9M	0.61	20.7	1.2	35.77	**22.39**
6	1024X256	4.5M	0.54	18.6	1.22	35.5	22.08
12	512X256	3.8M	0.55	19.2	1.21	35.5	21.8
24	256X256	3.5M	0.55	19.31	1.21	35.3	22.06
48	256X128	3.4M	0.56	19.5	1.21	35.4	**21.7**

AMI-HMI Results

- DNN+Sigmoid WER increases if model size is reduced.
- LA-DNN+Sigmoid beats DNN+Sigmoid with fewer parameters.
- LA-DNN+ReLU beats DNN+ReLU with fewer parameters.

Model	Num of H.Layers	Layers Size	# Params	Training		Validation		WER %
				Training CE	Training Frame Err %	Validation CE	Validation Frame Err	
DNN+Sigmoid	6	2048X2048	**37.6M**	1.46	37.83	2.11	49.3	**31.67**
DNN+Sigmoid	6	1024X1024	12.5M	1.59	40.75	2.13	50.0	32.43
DNN+ReLU	6	1024X1024	12.5M	1.45	40.47	2.00	47.5	31.54
LA-DNN+Sigmoid	6	2048X512	18.4M	1.35	35.3			31.88
LA-DNN+ReLU	6	1024X512	**10.5M**	1.34	35.7	2.02	47.3	**30.68**

AMI-HMI Results (Going Deeper)

- Deeper network, with fewer parameters, improves all validation metrics - To at least forty eight layers!
- Larger 48 layer system is slightly better.

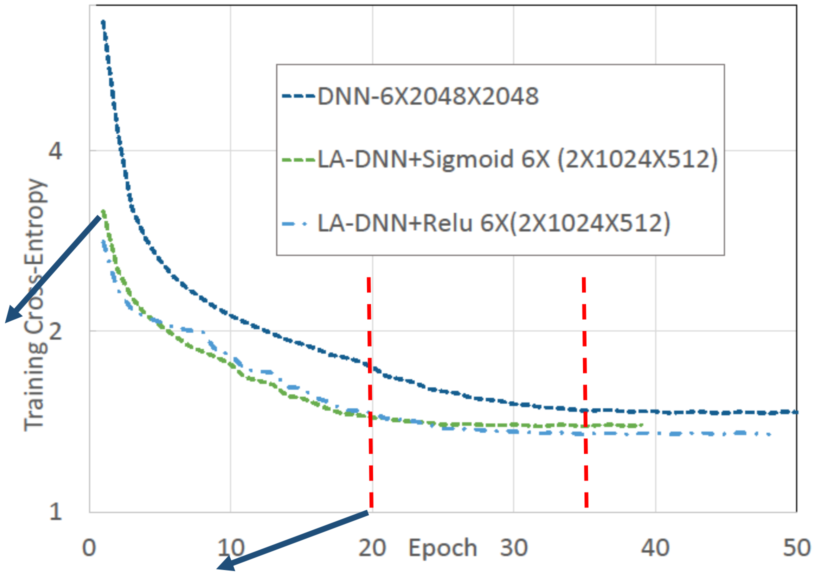
LA-DNN with ReLU Units							
Num of H.Layers	Layers Size	# Params	Training		Validation		WER %
			Training CE	Training Frame Err %	Validation CE	Validation Frame Err	
3	2048X512	12.1M	1.34	35.6	2.03	47.8	31.5
6	1024X512	10.5M	1.34	35.7	2.00	47.3	30.7
12	1024X256	8.9M	1.31	35.2	2.01	47.2	30.4
24	512X256	8.2M	1.34	35.7	1.99	47.2	30.2
48	256X256	7.9M	1.35	35.9	1.97	47.0	29.9
48	512X256	14M	1.25	33.9	2.00	46.7	29.7

Relation of LA-DNN to Pre-training

- Do we really need deep architecture?
- Complicated functions with high level abstraction (Bengio and Lecun 2007)
    - more complex functions given the same number of parameters
    - hierarchical representations
- How to train a deep network using normal DNN?
    - Problems with training deeper network
    - Gradient vanishing
- DNN solution => Unsupervised Pre-training
- LA-DNN solution => Bypass connection

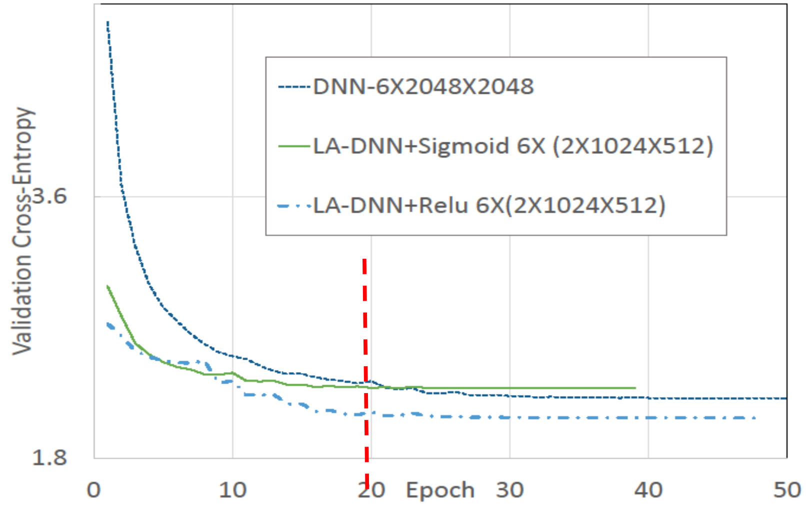
Relation of LA-DNN to Pre-training

- What problem does pre-training really tackle??
- Pre-training initializes the network in a region of the parameter space that is:
  - A better starting point for the non-convex optimization
  - Easier for optimization
  - Near better local optima
- LA-DNN is naturally initialized to a good information-preserving, gradient-passing starting point.



smaller CE after 1^{st} epoch

LA-DNN Converge after 20 epochs



Conclusion

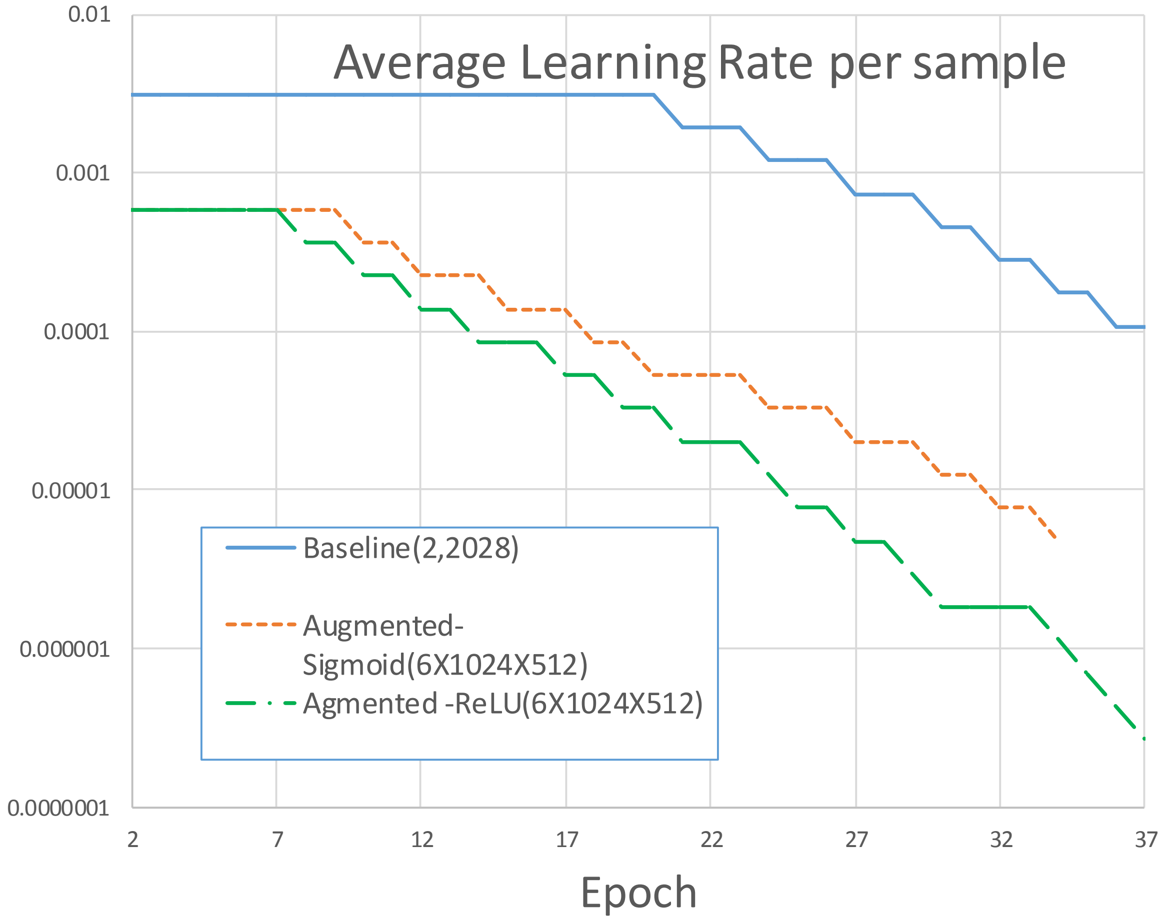
- Proposed a new layer structure for DNN.
- Including “linear augmentation”:
  - Tackles gradient vanishing problem
  - Improve initial gradient computation and results in faster convergence.
  - Higher modeling capacity with fewer parameters.
  - Enables training truly deep networks.
- Faster convergence, smaller network, better results.

BONUS SLIDES

DNN vs. LA-DNN

- In a Basin of attraction of gradient descent corresponding to better generalization performance

- Smaller initial learning rate(LR)
  - DNN initial LR is 0.8:3.2
  - LA-DNN LR is 0.1:0.4
  - Closer to the final solution in the region of parameter space and needs smaller step size.

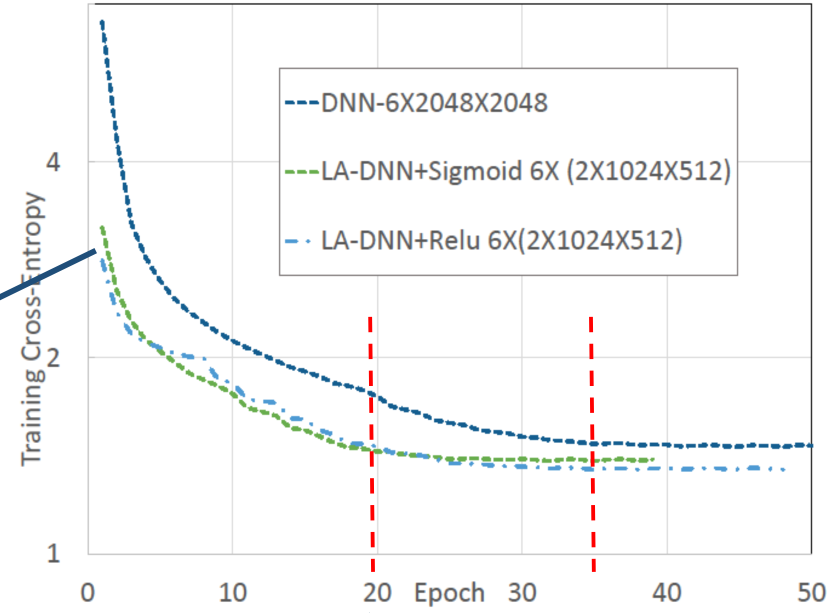


Linear Augmented Model

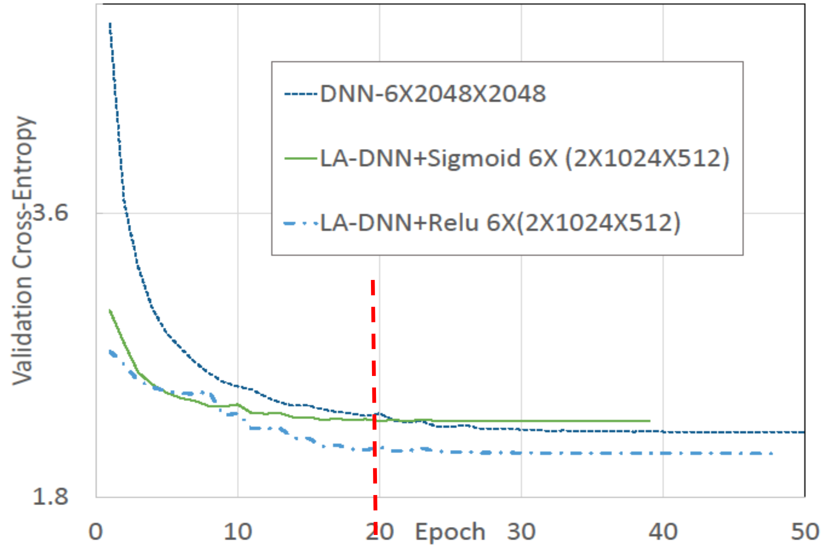
- Better gradient for initial steps

- Faster convergence rate
  - Better error backpropagation
  - Better initial model

smaller CE after 1st epoch



LA-DNN Converge after 20 epochs



SDNN versus Deep stacking network (DSN)

- In DSN, the input of layer l is the outputs of all previous layers stacked together.

- In DSN, we increase layer dimension, specially in a very deep networks.