YEAR 2014

Do Deep Nets Really Need to be Deep?

Lei Jimmy Ba (Toronto)
Rich Caruana (MS Research)
@ NIPS 2014

article overview
by Ilya Kuzovkin
@ Coputational Neuroscience Seminar
@ University of Tartu

Training set with 1M labeled points

Training set with 1M labeled points

Shallow net: 86% accuracy

Deep net: 91% accuracy

Training set with 1M labeled points

Shallow net: 86% accuracy

Deep net: 91% accuracy

What is the source of improvement?

a) deep net has more parameters?

a) deep net has more parameters?

b) deep architecture allows to learn
more complex functions?

a) deep net has more parameters?

b) deep architecture allows to learn more complex functions?

c) deep nets have better inductive bias? (e.g. hierarchy is good)

Inductive bias

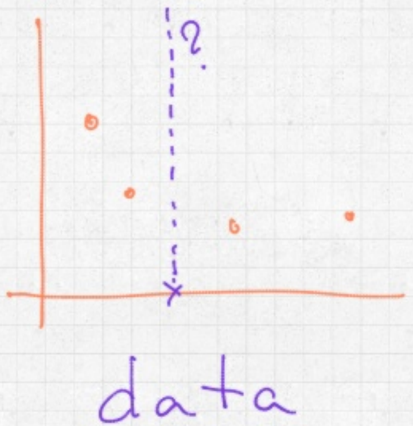
assumption about the unseen data samples

example of inductive bias: Occam's razor in ML

Inductive bias

assumption about the unseen data samples

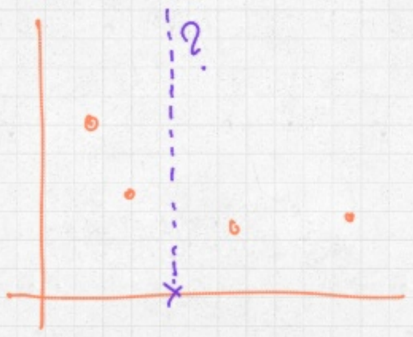
example of inductive bias: Occam's razor in ML



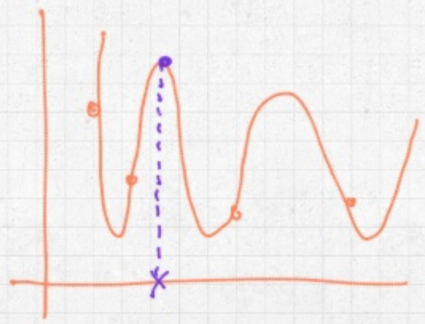
Inductive bias

assumption about the unseen data samples

example of inductive bias: Occam's razor in ML



data

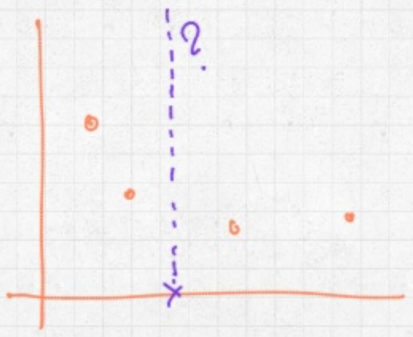


possible model

Inductive bias

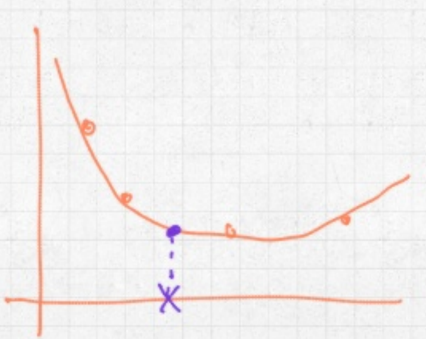
assumption about the unseen data samples

example of inductive bias: Occam's razor in ML



data

possible model

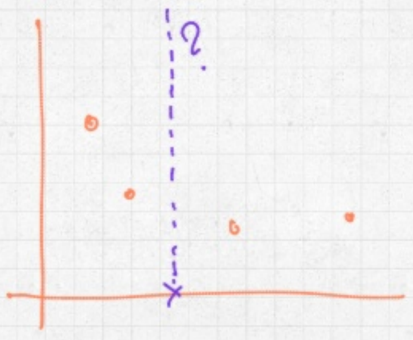


preferred model

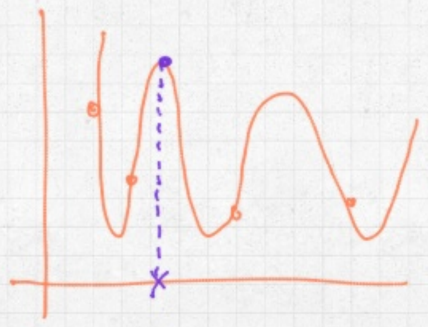
Inductive bias

assumption about the unseen data samples

example of inductive bias: Occam's razor in ML



data



possible model

preferred model

- Conditional independence
- Minimal CV error
- Maximal margin in SVM
- Minimum description
- Minimum features
- Nearest neighbors

a) deep net has more parameters?

b) deep architecture allows to learn more complex functions?

c) deep nets have better inductive bias? (e.g. hierarchy is good)

d) convolution gives a lot?

a) deep net has more parameters?

b) deep architecture allows to learn more complex functions?

c) deep nets have better inductive bias? (e.g. hierarchy is good)

d) convolution gives a lot?

e) current learning algorithms work better with deep architecture?

a) deep net has more parameters?

b) deep architecture allows to learn more complex functions?

c) deep nets have better inductive bias? (e.g. hierarchy is good)

d) convolution gives a lot?

e) current learning algorithms work better with deep architecture?

f) all of above

g) none of above

There are articles showing how deep nets excel over shallow nets and imply that it is due to this or that property of DNN.

There are articles showing how deep nets excel over shallow nets and imply that it is due to this or that property of DNN.

Here we show that it is possible to train a shallow net, which will mimic the function of a deep net.

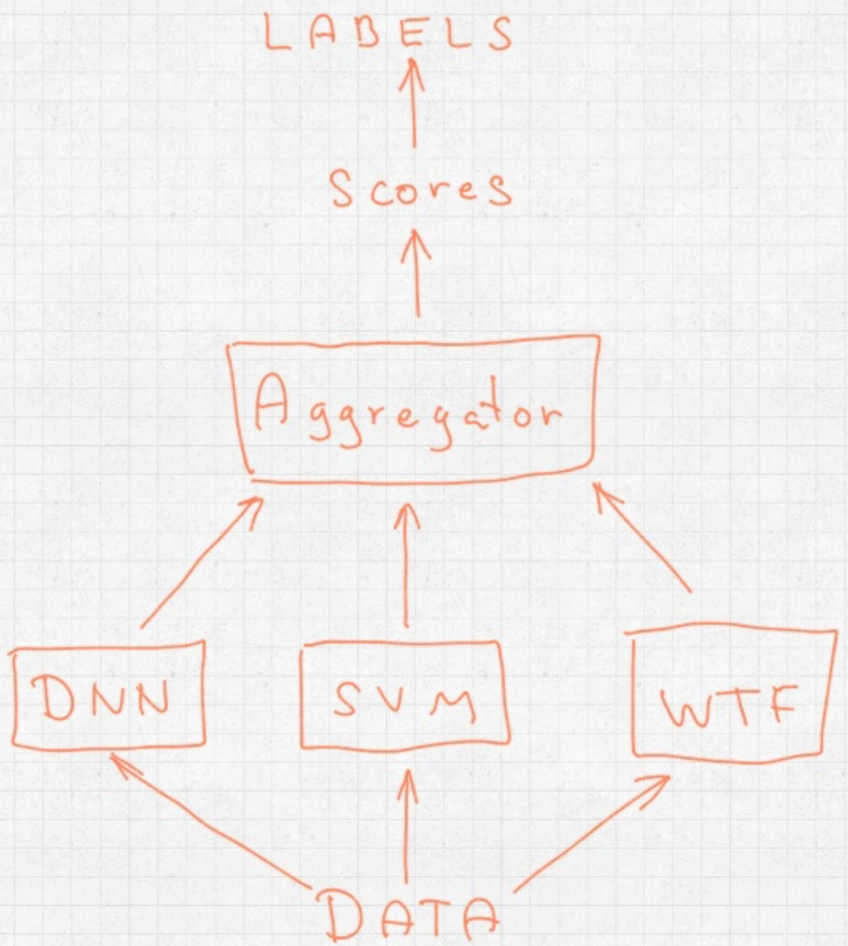
There are articles showing how deep nets excel over shallow nets and imply that it is due to this or that property of DNN.

Here we show that it is possible to train a shallow net, which will mimic the function of a deep net.

Possible to mimic bot not able to train. Deep not a requirement. Success is related to the learning process

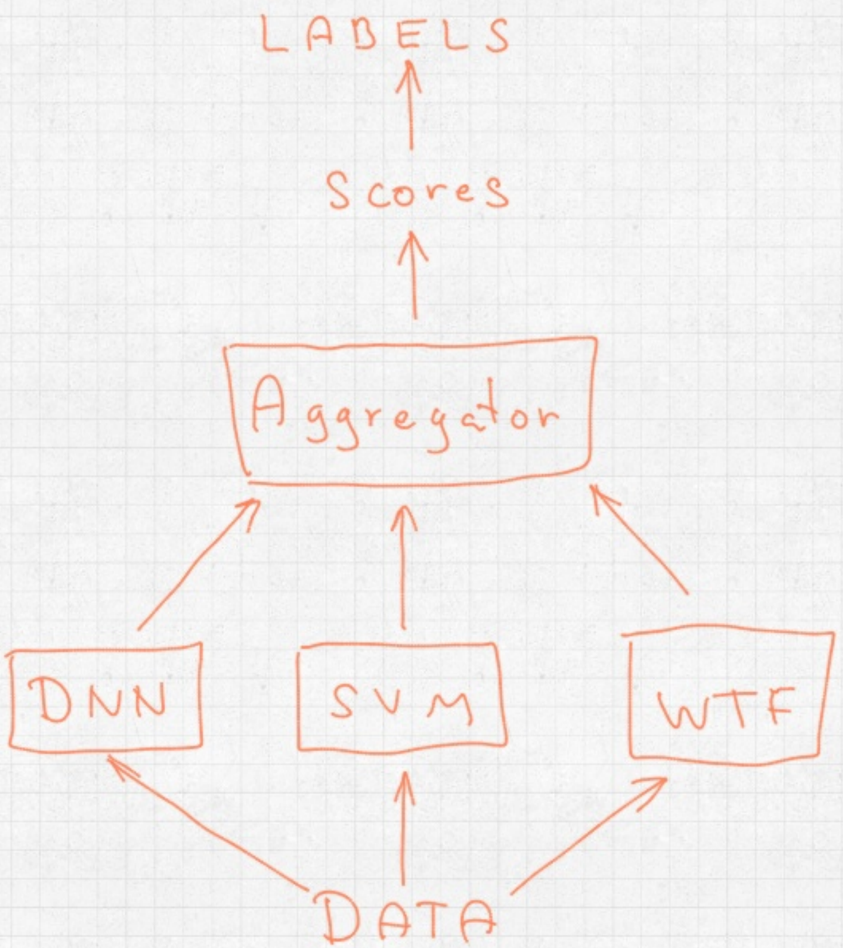
Model Compression (2.1)

1) Build a complex model

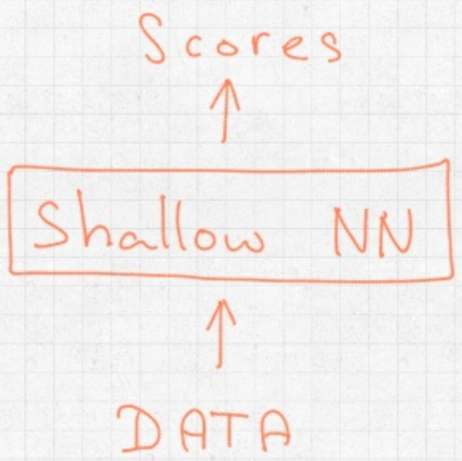


Model Compression (2.1)

1) Build a complex model

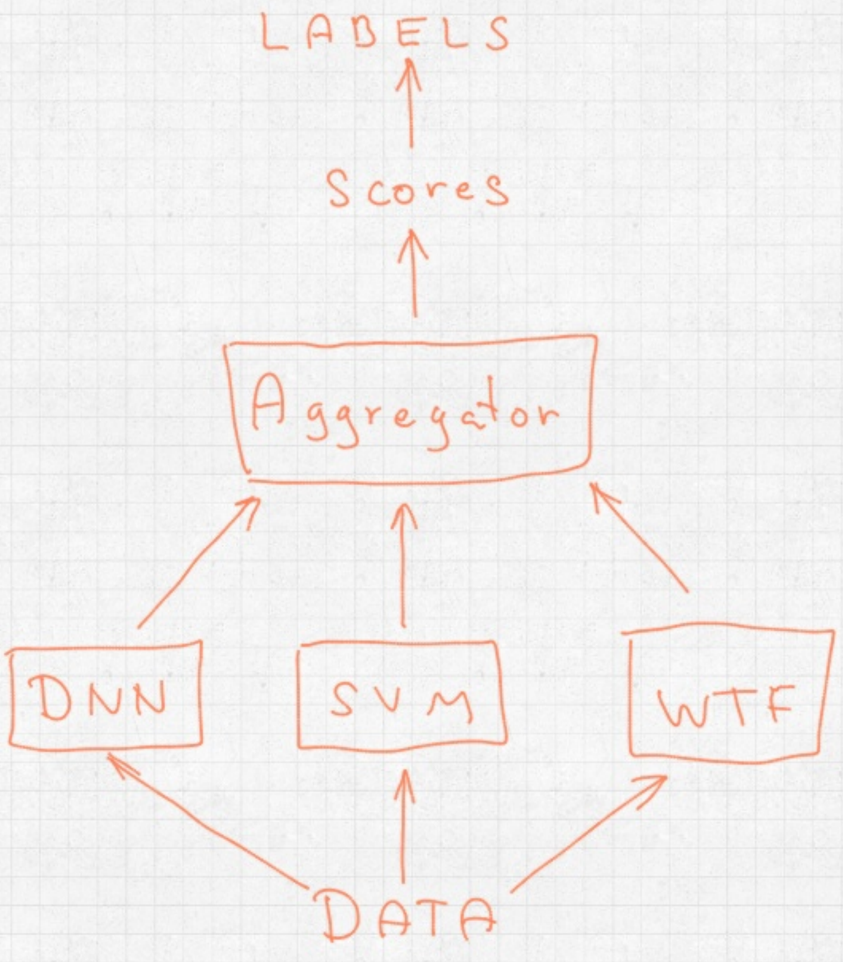


2) Train a simple model to mimic complex function

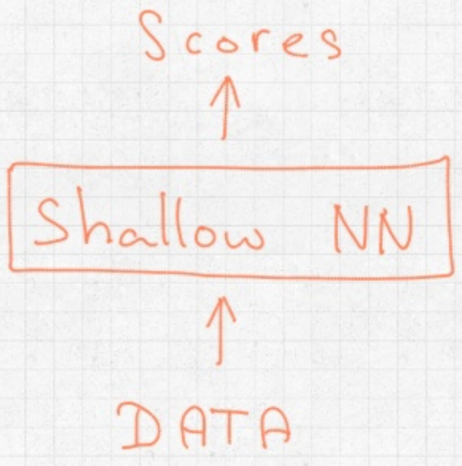


Model Compression (2.1)

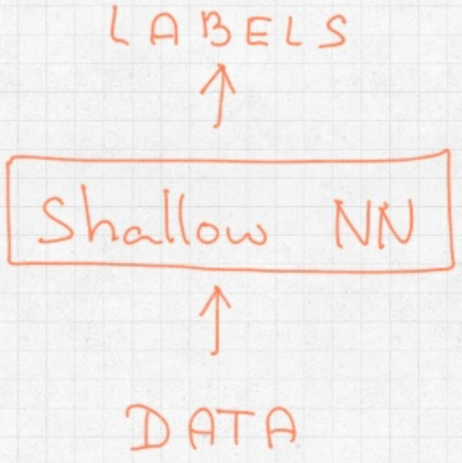
1) Build a complex model



2) Train a simple model to mimic complex function

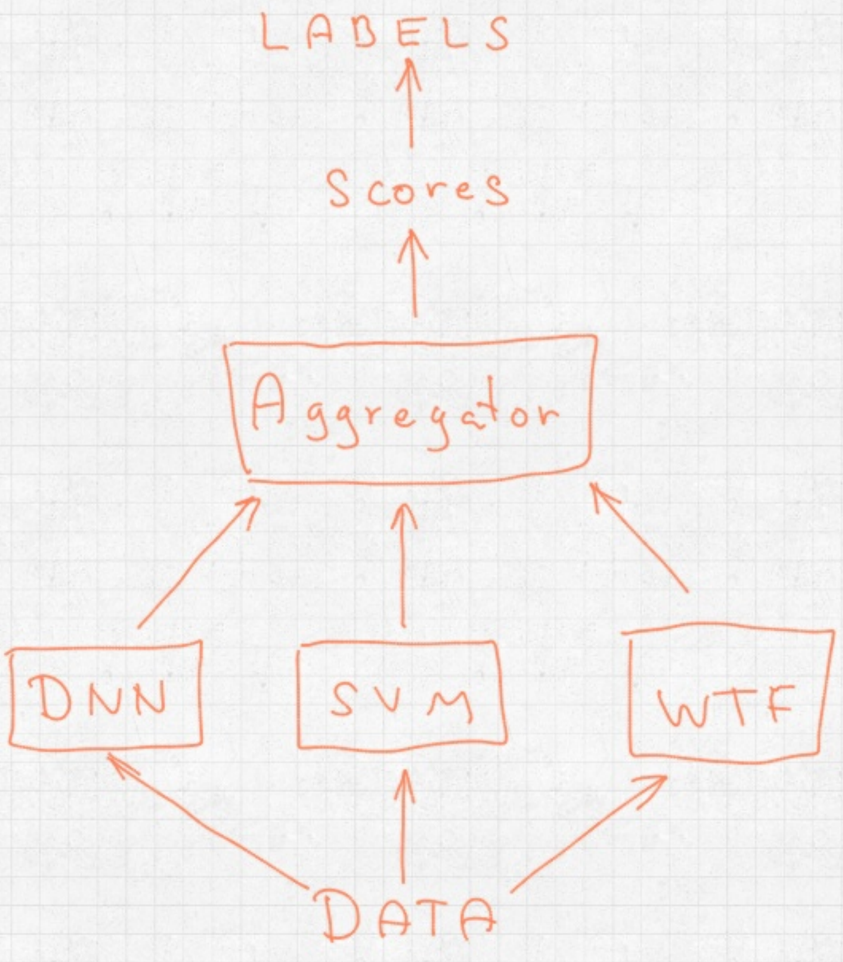


3) Apply it

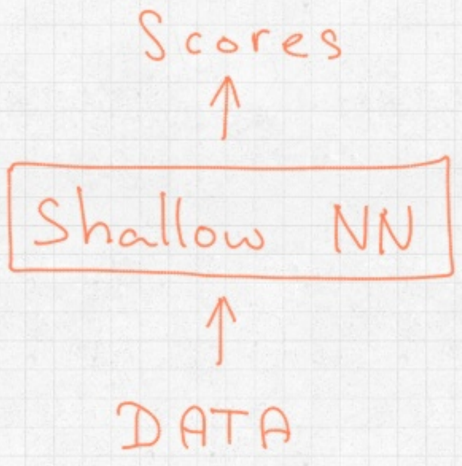


Model Compression (2.1)

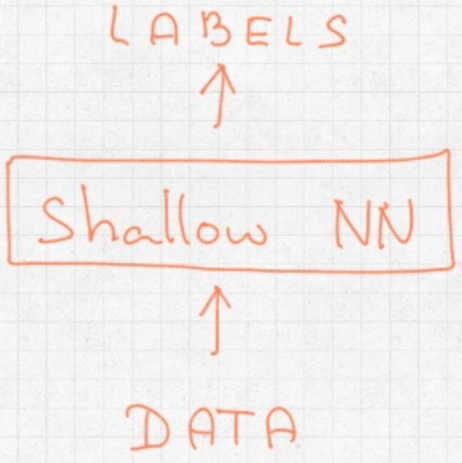
1) Build a complex model



2) Train a simple model to mimic complex function



3) Apply it

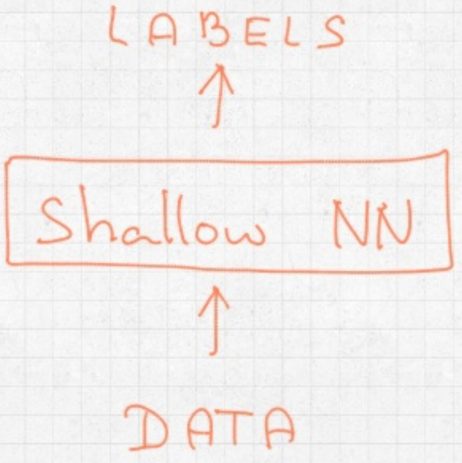


Works!

Model Compression (2.1)

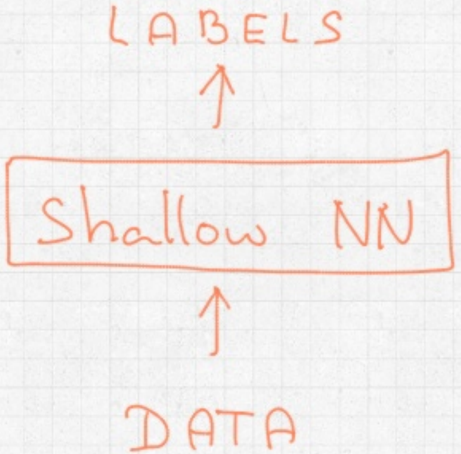
however

1) Train a simple model



Does not work!

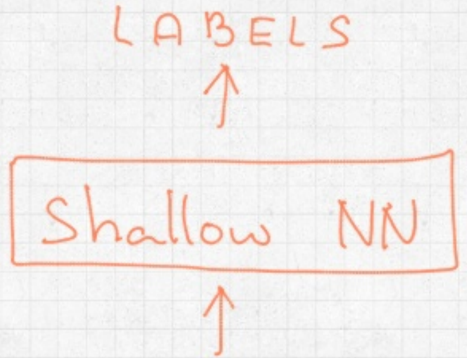
2) and then apply it



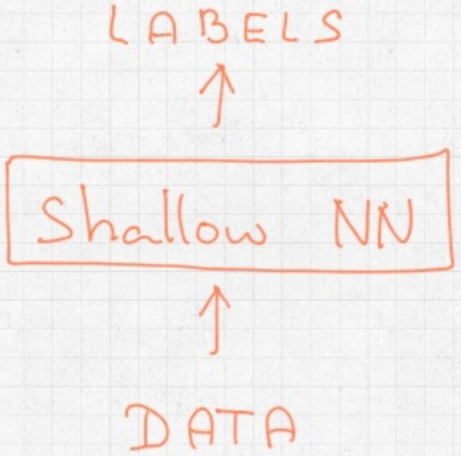
Model Compression (2.1)

however

1) Train a simple model



2) and then apply it

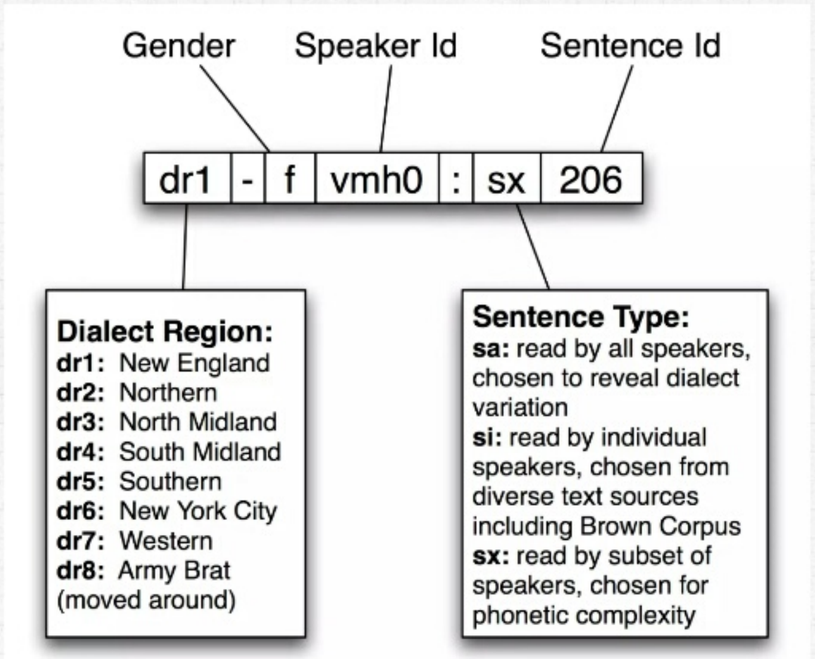


Does not work!

Compression demonstrates that in principle simple NN could learn same function, but we (current learning algorithms) don't know how.

TIMIT

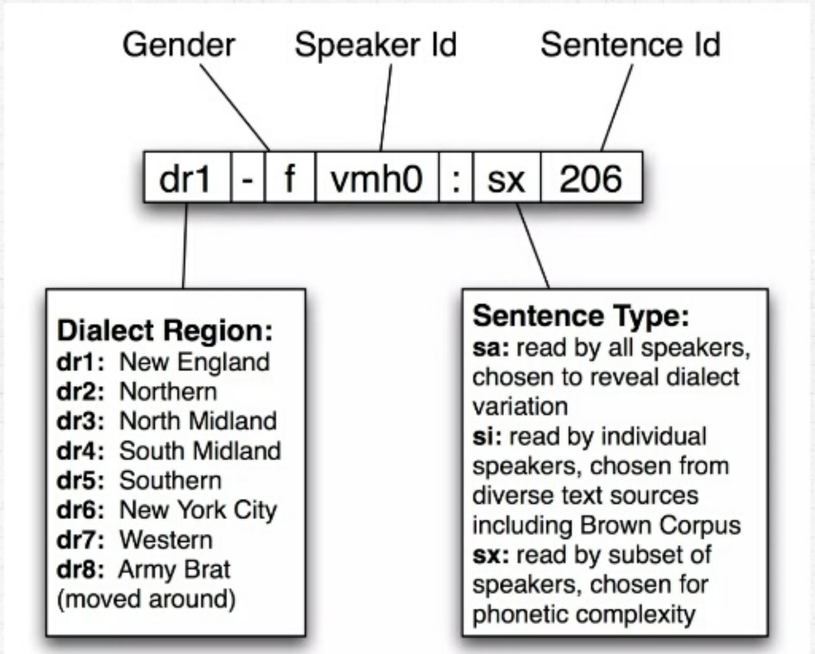
lexically and phonetically
labeled 630 x 10 sentences.



61 phonemes, each represented by 3 classes - 3 most popular triphones

TIMIT

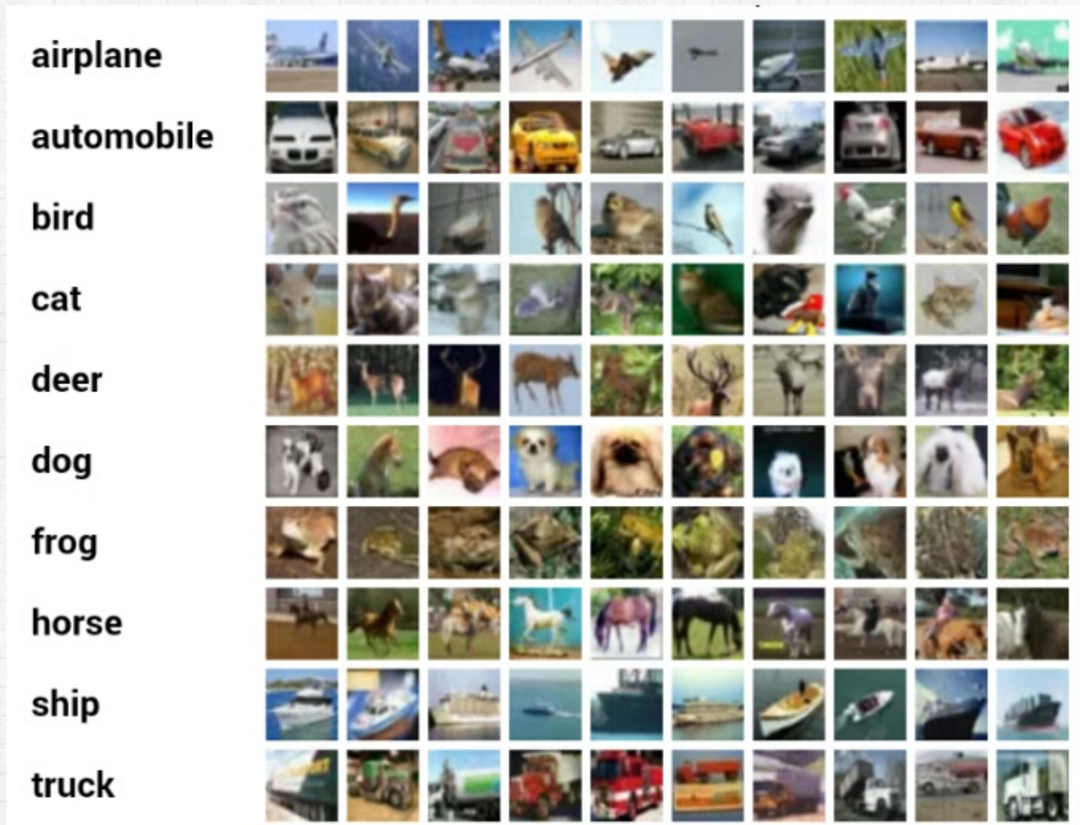
lexically and phonetically labeled 630 x 10 sentences.



61 phonemes, each represented by 3 classes - 3 most popular triphones

CIFAR-10

60,000 32x32 color images
10 classes



2.2 Mimic Learning

original models trained with softmax output

$$p_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$$

z_i is called logit

2.2 Mimic Learning

original models trained with softmax output

$$p_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$$

z_i is called logit

shallow mimic net is trained on 183 logits:

$$[10, 20, 30]$$

is more informative than

$$[10^{-5}, 10^{-4}, 0.999]$$

2.2 Mimic Learning

original models trained with softmax output

$$p_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$$

z_i is called logit

shallow mimic net is trained on 183 logits:

$$[10, 20, 30]$$

is more informative than

$$[10^{-5}, 10^{-4}, 0.999]$$

SNN-MIMIC objective

$$L(W, \beta) = \frac{1}{2T} \sum_t \| \beta f(W x^t) - z^t \|^2$$

W – input-to-hidden weights

β – hidden-to-output weights

$\beta f(W x^t)$ – model's prediction on t'th sample

2.2 Mimic Learning

original models trained with softmax output

$$p_k = \frac{e^{z_k}}{\sum_j e^{z_j}}$$

z_i is called logit

shallow mimic net is trained on 183 logits:

$$[10, 20, 30]$$

is more informative than

$$[10^{-5}, 10^{-4}, 0.999]$$

SNN-MIMIC objective

$$L(W, \beta) = \frac{1}{2T} \sum_t \| \beta f(W x^t) - z^t \|^2$$

W – input-to-hidden weights

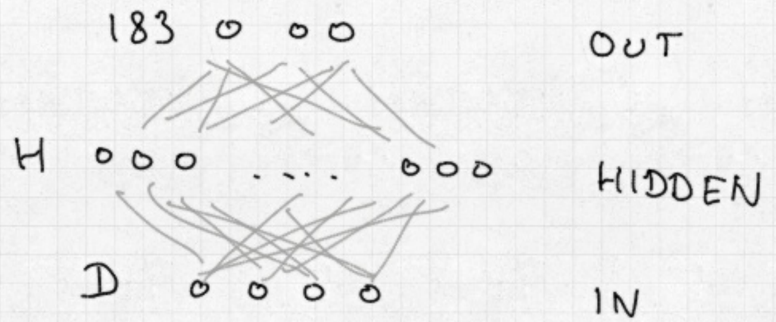
β – hidden-to-output weights

$\beta f(W x^t)$ – model's prediction on t'th sample

* back-propagation
* SGD with momentum

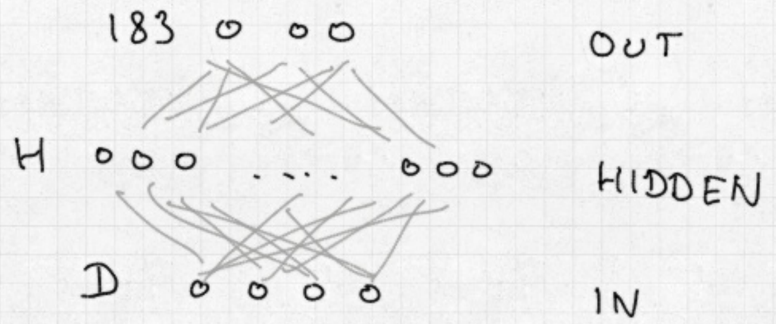
2.3 Speed-up mimic learning

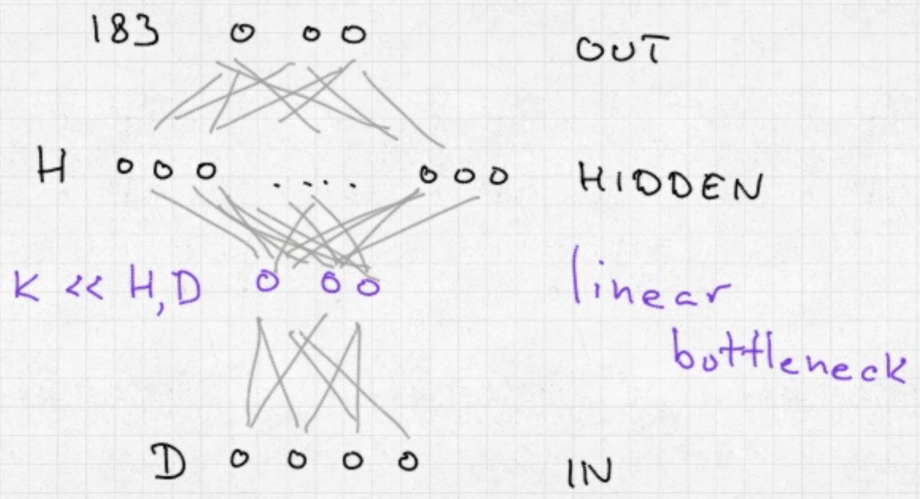
Shallow net has same number of parameters so learning is slow (multiple weeks on GPU)



2.3 Speed-up mimic learning

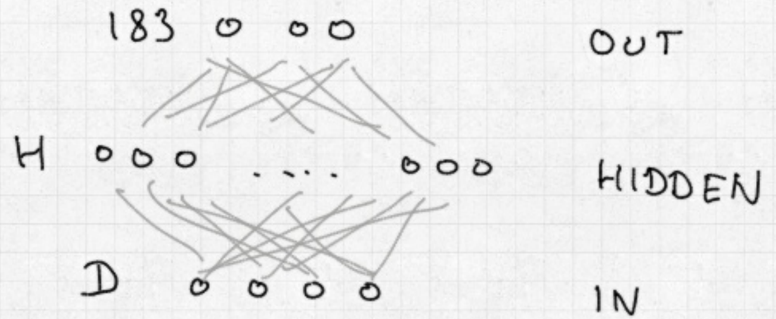
Shallow net has same number of parameters so learning is slow (multiple weeks on GPU)





2.3 Speed-up mimic learning

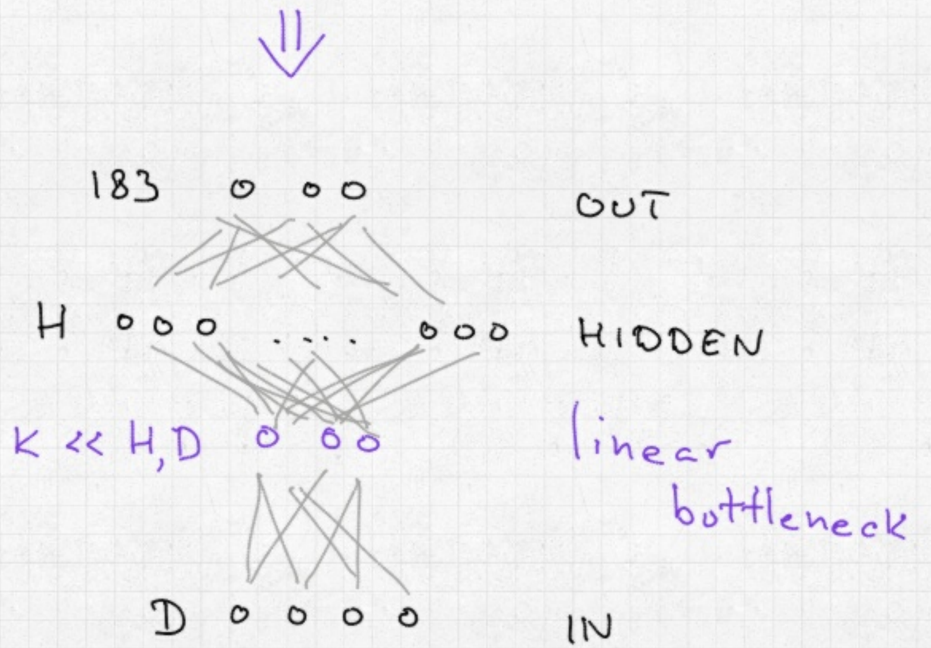
Shallow net has same number of parameters so learning is slow (multiple weeks on GPU)



Instead of $W \in \mathbb{R}^{H \times D}$ (hidden, input)

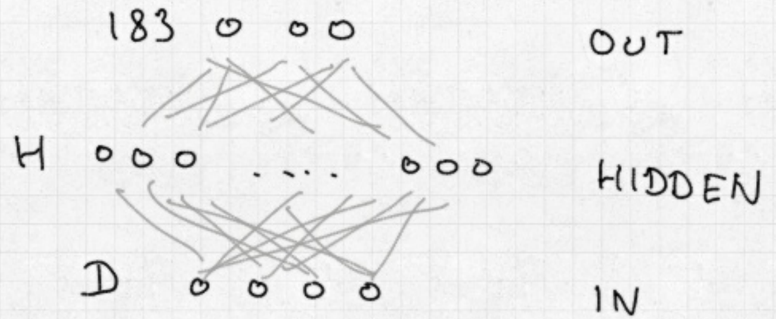
we have $U \in \mathbb{R}^{H \times K}$

and $V \in \mathbb{R}^{K \times D}$



2.3 Speed-up mimic learning

Shallow net has same number of parameters so learning is slow (multiple weeks on GPU)



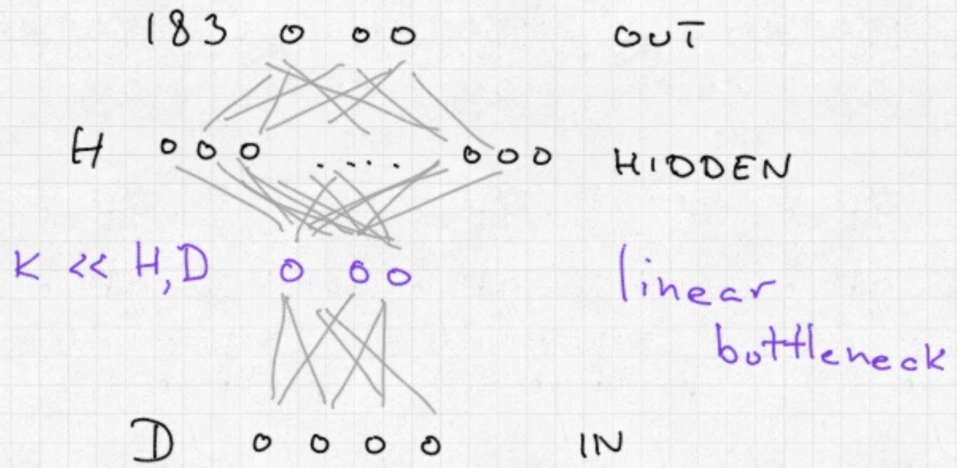
Instead of $W \in \mathbb{R}^{H \times D}$ (hidden, input)

we have $U \in \mathbb{R}^{H \times K}$

and $V \in \mathbb{R}^{K \times D}$

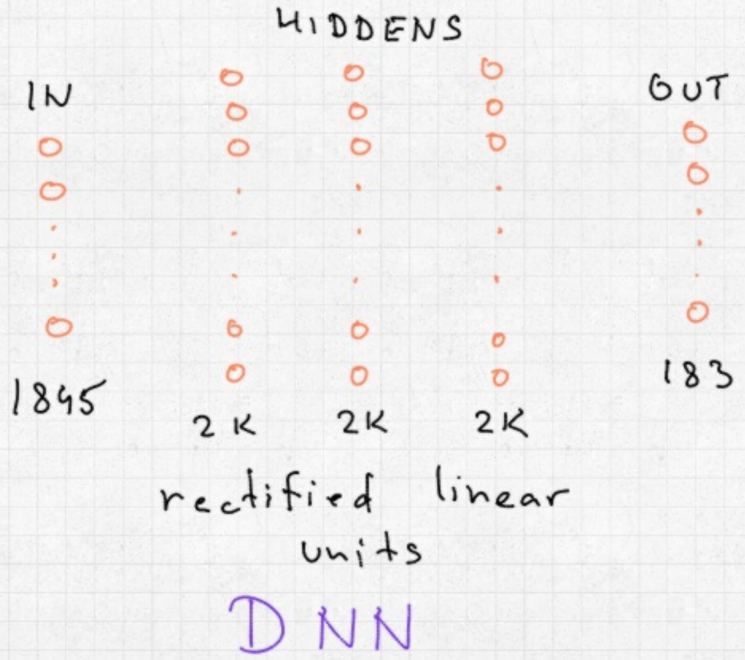
$$L(U,V,\beta) = \frac{1}{2T} \sum_t \| \beta f(UVx^t) - z^t \|^2$$

2.3 Speed-up mimic learning



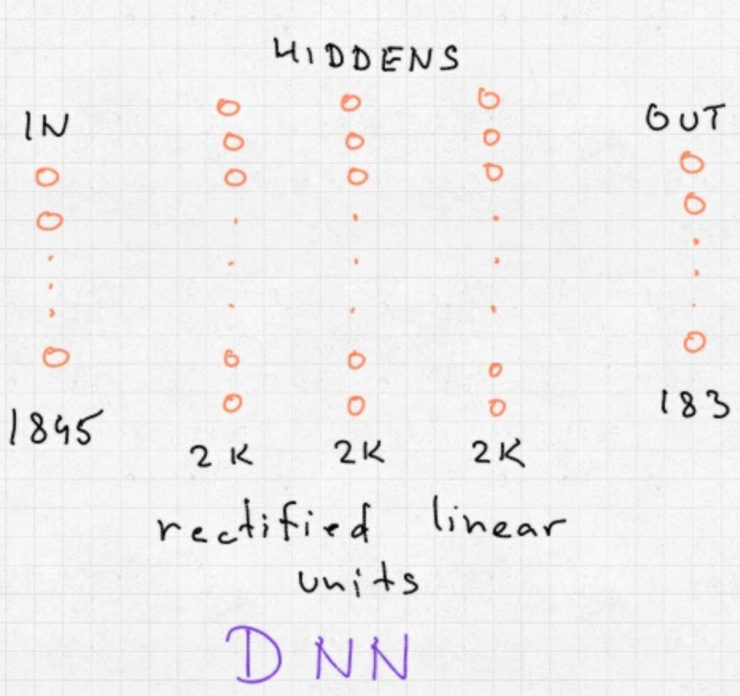
* "Dramatical" speed-up
* $O(K(H+D))$ memory instead of $O(HD)$
* applying matrix factorization between input and hidden is a novel idea
* if anything linear bottleneck only reduces power of network

Experiments on TIMIT (3)

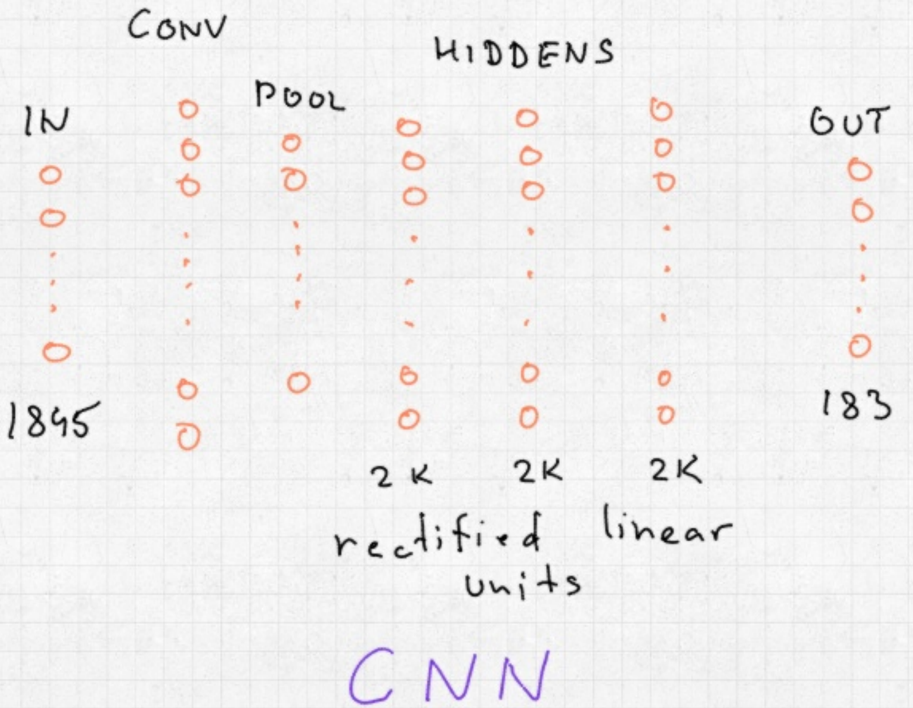


DNN

Experiments on TIMIT (3)

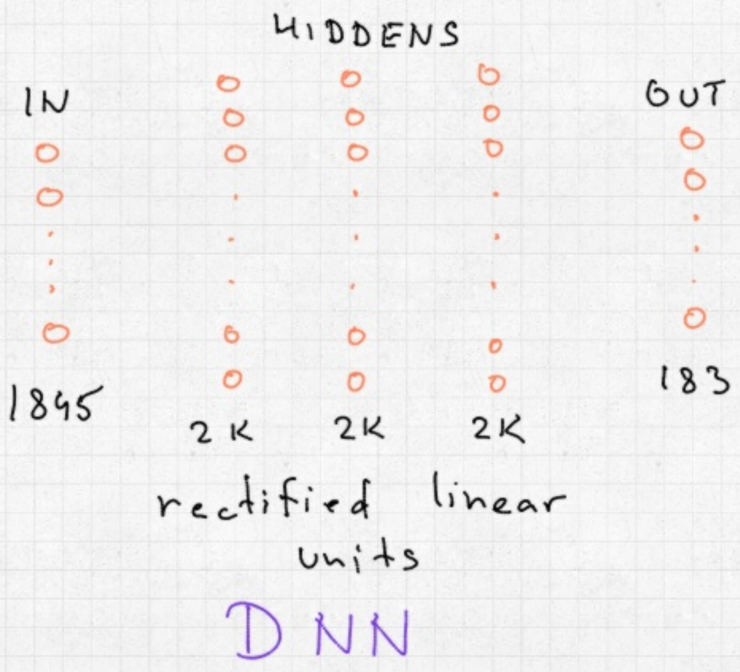


DNN

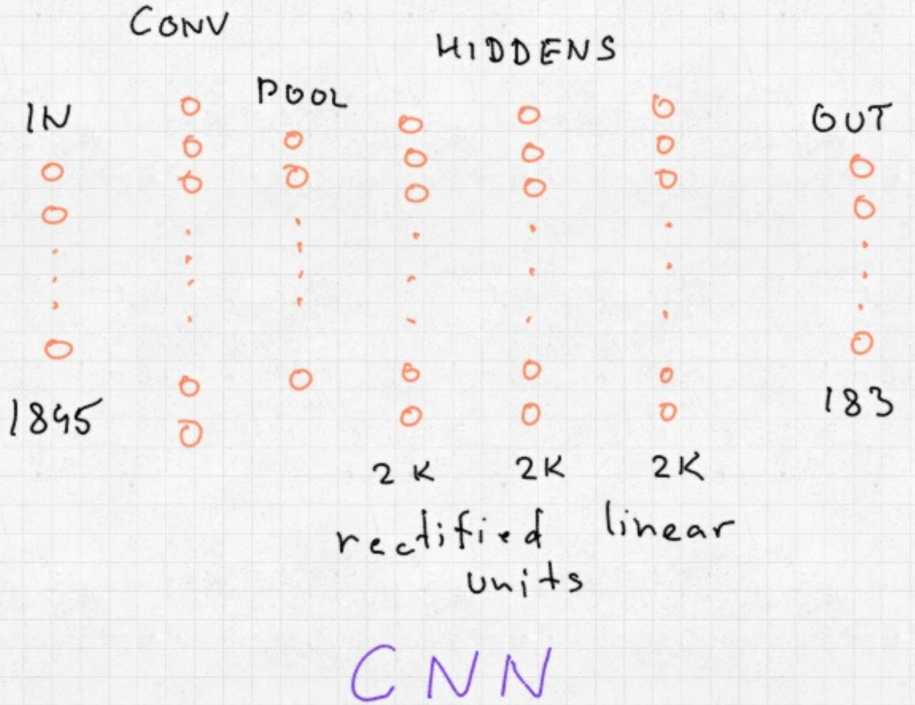


CNN

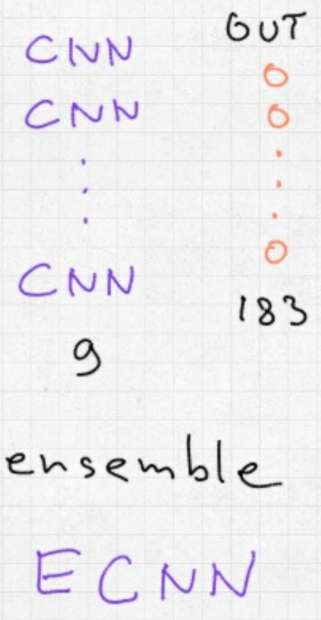
Experiments on TIMIT (3)



DNN

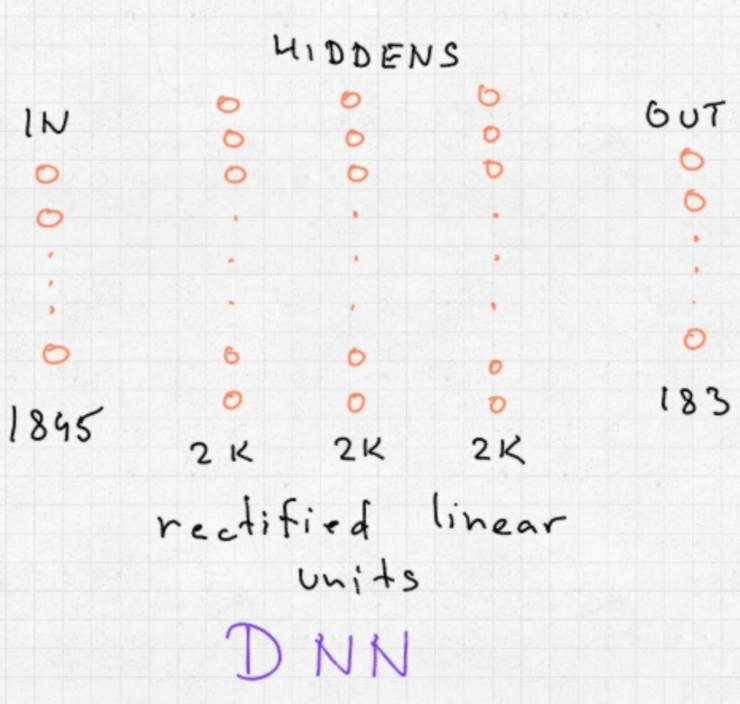


CNN

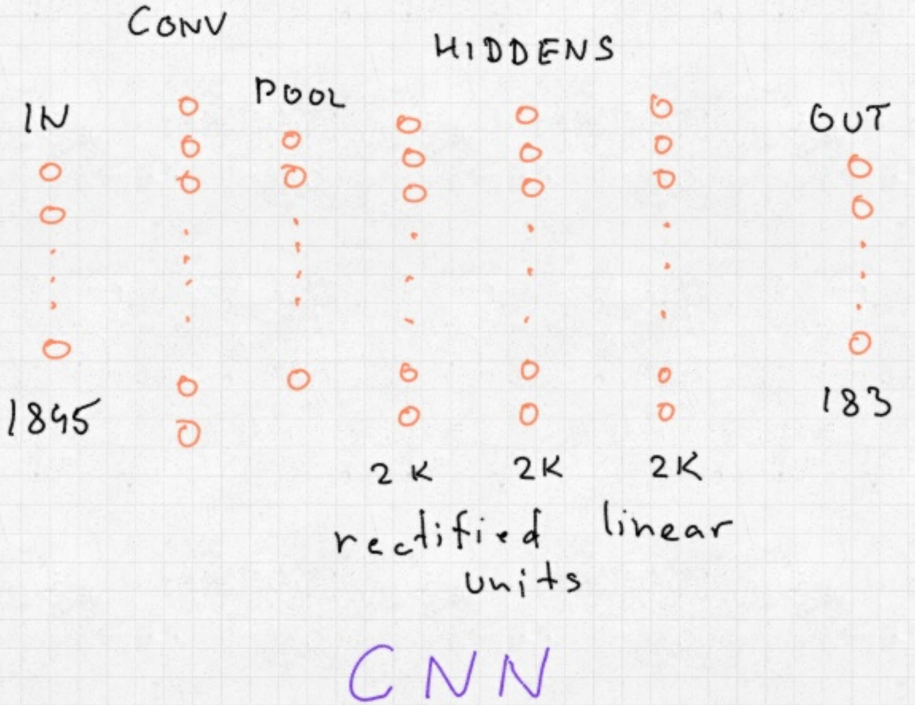


ECNN

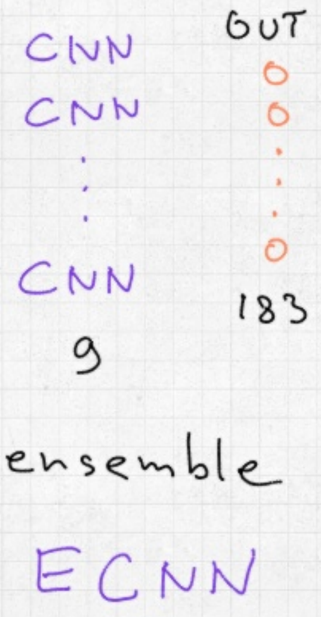
Experiments on TIMIT (3)



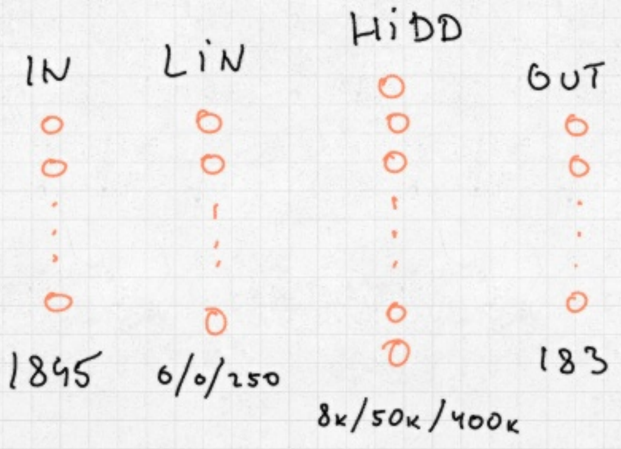
DNN



CNN

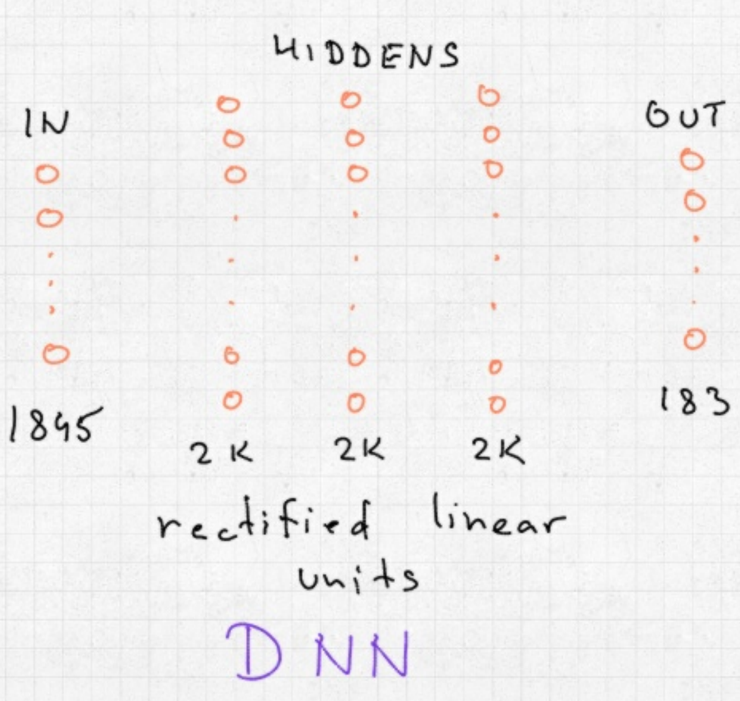


ECNN

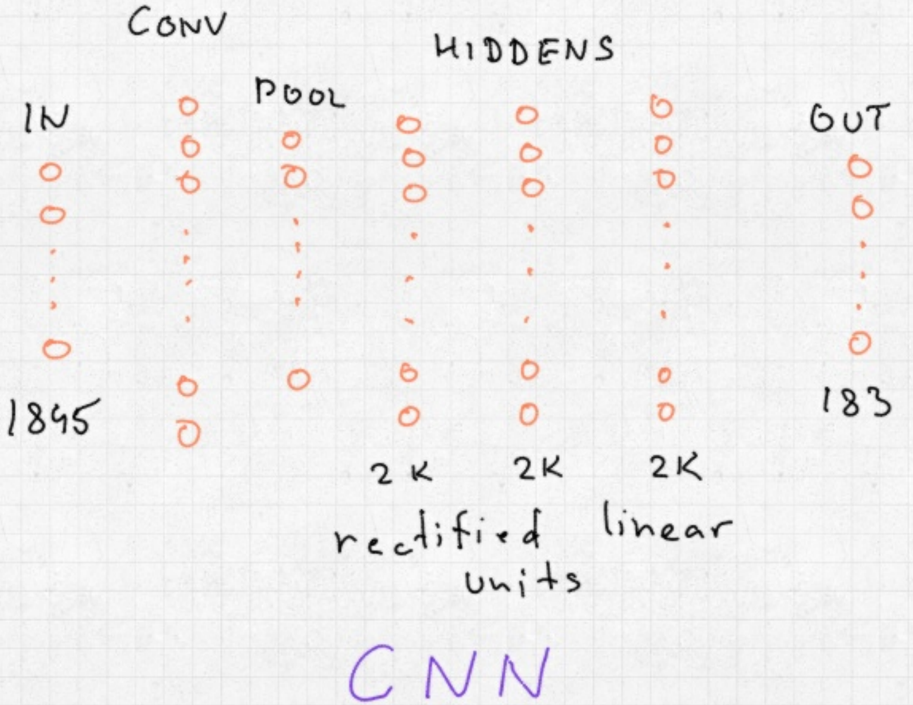


SNN - 8K/50K/400K

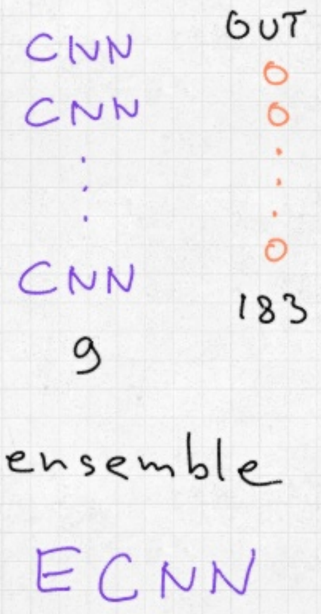
Experiments on TIMIT (3)



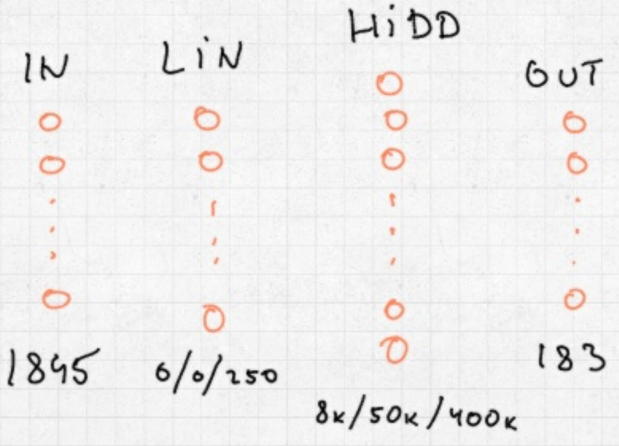
DNN



CNN



ECNN



SNN - 8K/50K/400K

SNN - MIMIC - 8K/400K

	Architecture	# Param.	# Hidden units	PER
DNN	2k-2k-2k + dropout trained on original data	∼12M	∼6k	21.9%
CNN	c-p-2k-2k-2k + dropout trained on original data	∼13M	∼10k	**19.5%**
ECNN	ensemble of 9 CNNs	∼125M	∼90k	**18.5%**

Table 1: Comparison of shallow and deep models: phone error rate (PER) on TIMIT core test set.

	Architecture	# Param.	# Hidden units	PER
SNN-8k	8k + dropout trained on original data	~12M	~8k	23.1%
SNN-50k	50k + dropout trained on original data	~100M	~50k	23.0%
SNN-400k	250L-400k + dropout trained on original data	~180M	~400k	23.6%
DNN	2k-2k-2k + dropout trained on original data	~12M	~6k	21.9%
CNN	c-p-2k-2k-2k + dropout trained on original data	~13M	~10k	**19.5%**
ECNN	ensemble of 9 CNNs	~125M	~90k	**18.5%**

Table 1: Comparison of shallow and deep models: phone error rate (PER) on TIMIT core test set.

	Architecture	# Param.	# Hidden units	PER
SNN-8k	8k + dropout trained on original data	~12M	~8k	23.1%
SNN-50k	50k + dropout trained on original data	~100M	~50k	23.0%
SNN-400k	250L-400k + dropout trained on original data	~180M	~400k	23.6%
DNN	2k-2k-2k + dropout trained on original data	~12M	~6k	21.9%
CNN	c-p-2k-2k-2k + dropout trained on original data	~13M	~10k	**19.5%**
ECNN	ensemble of 9 CNNs	~125M	~90k	**18.5%**
SNN-MIMIC-8k	250L-8k no convolution or pooling layers	~12M	~8k	**21.6%**
SNN-MIMIC-400k	250L-400k no convolution or pooling layers	~180M	~400k	**20.0%**

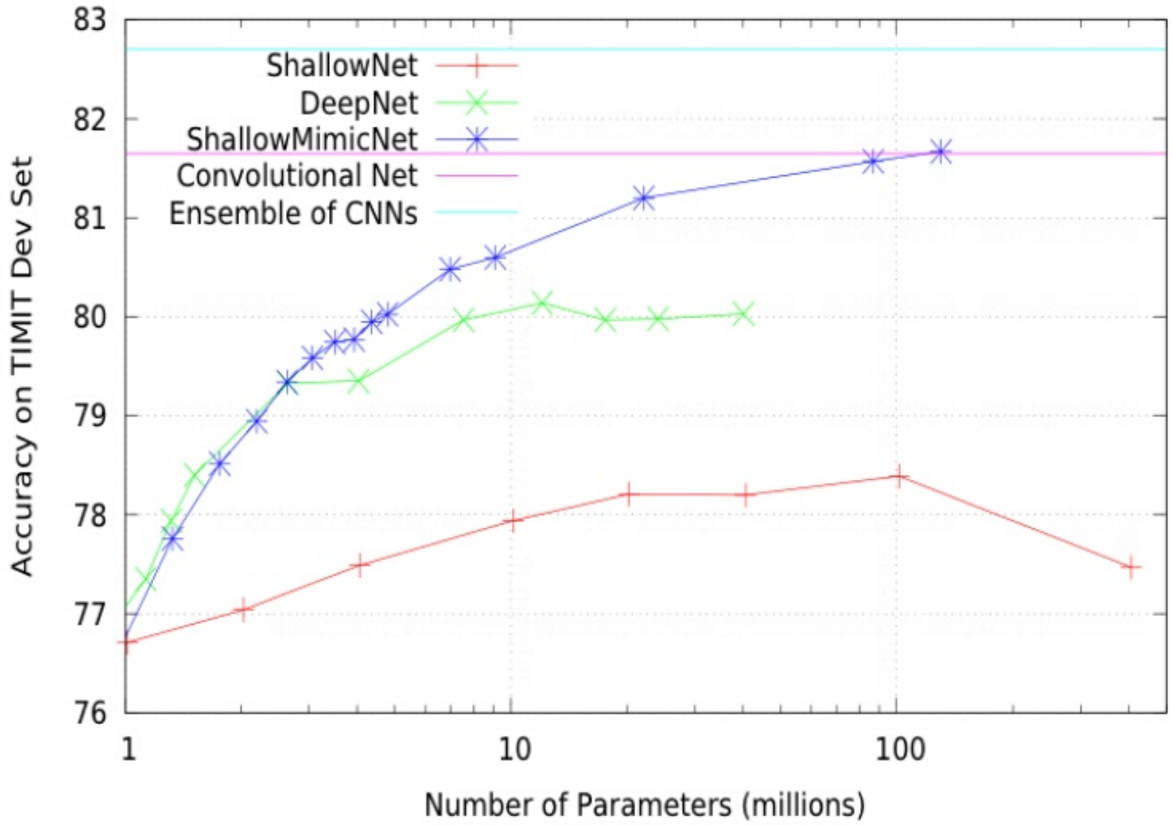
Table 1: Comparison of shallow and deep models: phone error rate (PER) on TIMIT core test set.

	Architecture	# Param.	# Hidden units	PER
SNN-8k	8k + dropout trained on original data	∼12M	∼8k	23.1%
SNN-50k	50k + dropout trained on original data	∼100M	∼50k	23.0%
SNN-400k	250L-400k + dropout trained on original data	∼180M	∼400k	23.6%
DNN	2k-2k-2k + dropout trained on original data	∼12M	∼6k	21.9%
CNN	c-p-2k-2k-2k + dropout trained on original data	∼13M	∼10k	**19.5%**
ECNN	ensemble of 9 CNNs	∼125M	∼90k	**18.5%**
SNN-MIMIC-8k	250L-8k no convolution or pooling layers	∼12M	∼8k	**21.6%**
SNN-MIMIC-400k	250L-400k no convolution or pooling layers	∼180M	∼400k	**20.0%**

Table 1: Comparison of shallow and deep models: phone error rate (PER) on TIMIT core test set.

	Architecture	# Param.	# Hidden units	PER
SNN-8k	8k + dropout trained on original data	∼12M	∼8k	23.1%
SNN-50k	50k + dropout trained on original data	∼100M	∼50k	23.0%
SNN-400k	250L-400k + dropout trained on original data	∼180M	∼400k	23.6%
DNN	2k-2k-2k + dropout trained on original data	∼12M	∼6k	21.9%
CNN	c-p-2k-2k-2k + dropout trained on original data	∼13M	∼10k	**19.5%**
ECNN	ensemble of 9 CNNs	∼125M	∼90k	**18.5%**
SNN-MIMIC-8k	250L-8k no convolution or pooling layers	∼12M	∼8k	**21.6%**
SNN-MIMIC-400k	250L-400k no convolution or pooling layers	∼180M	∼400k	**20.0%**

Table 1: Comparison of shallow and deep models: phone error rate (PER) on TIMIT core test set.



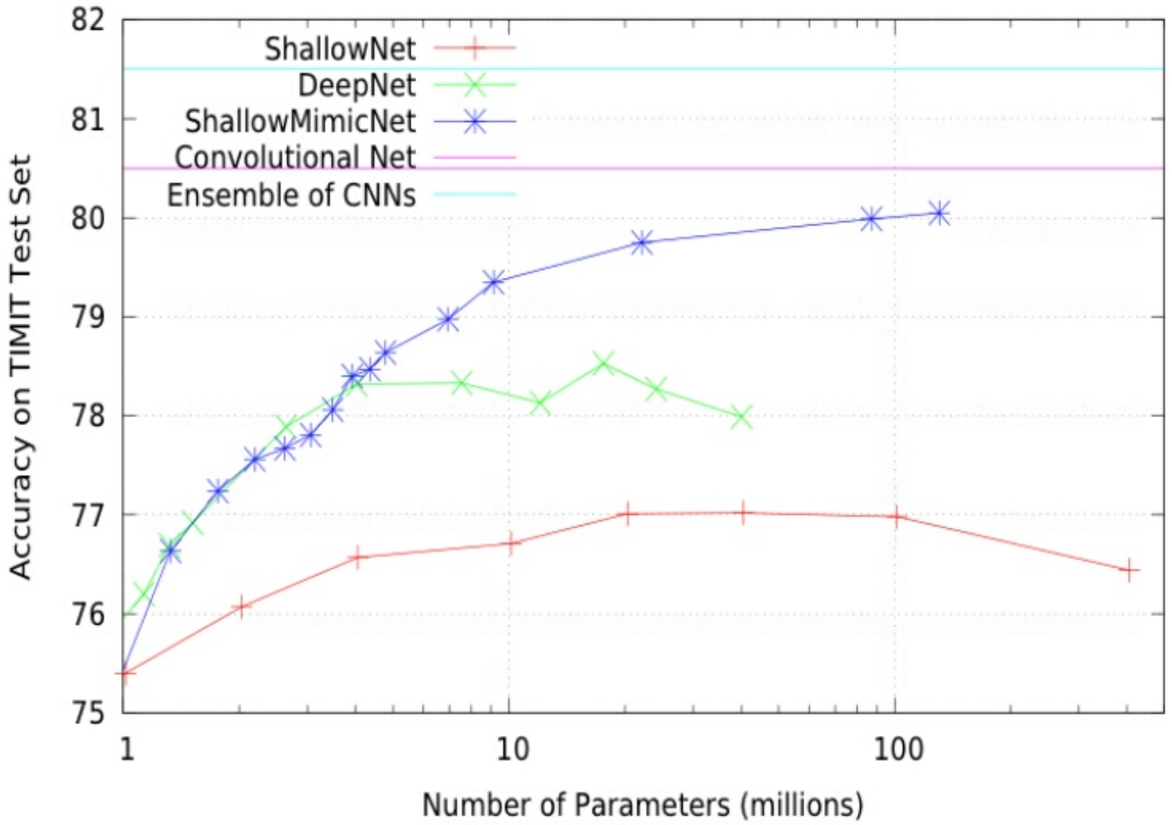


Figure 1: Accuracy of SNNs, DNNs, and Mimic SNNs vs. # of parameters on TIMIT Dev (left) and Test (right) sets. Accuracy of the CNN and target ECNN are shown as horizontal lines for reference.

Experiments on CIFAR-10

INPUT: 3072 32x32 in color

OUT: 10

TRAINING: 50,000

Experiments on CIFAR-10

INPUT: 3072 32x32 in color

OUT : 10

TRAINING: 50,000

Additionaly 1M images From "80M tiny images" for evaluating on trained ECNN

Experiments on CIFAR-10

INPUT: 3072 32x32 in color

OUT : 10

TRAINING: 50,000

Additionaly 1M images From "80M tiny images" for evaluating on trained ECNN

No matter how deep without convolution net performs poorly (on images), thus we add convolution + pooling layers into MIMIC models

	Architecture	# Param.	# Hidden units	Err.
DNN	2000-2000 + dropout	∼10M	4k	57.8%
SNN-30k	128c-p-1200L-30k + dropout input&hidden	∼70M	∼190k	21.8%
single-layer feature extraction	4000c-p followed by SVM	∼125M	∼3.7B	18.4%
CNN[11] (no augmentation)	64c-p-64c-p-64c-p-16lc + dropout on lc	∼10k	∼110k	15.6%
CNN[21] (no augmentation)	64c-p-64c-p-128c-p-fc + dropout on fc and stochastic pooling	∼56k	∼120k	15.13%
teacher CNN (no augmentation)	128c-p-128c-p-128c-p-1kfc + dropout on fc and stochastic pooling	∼35k	∼210k	**12.0%**
ECNN (no augmentation)	ensemble of 4 CNNs	∼140k	∼840k	**11.0%**
SNN-CNN-MIMIC-30k trained on a single CNN	64c-p-1200L-30k with no regularization	∼54M	∼110k	**15.4%**
SNN-CNN-MIMIC-30k trained on a single CNN	128c-p-1200L-30k with no regularization	∼70M	∼190k	**15.1%**
SNN-ECNN-MIMIC-30k trained on ensemble	128c-p-1200L-30k with no regularization	∼70M	∼190k	**14.2%**

Table 2: Comparison of shallow and deep models: classification error rate on CIFAR-10. Key: c, convolution layer; p, pooling layer; lc, locally connected layer; fc, fully connected layer

Why mimic models can be more accurate than training on original labels?

1) If labels have errors teacher may eliminate them making things easier for student

Why mimic models can be more accurate than training on original labels?

1) If labels have errors teacher may eliminate them making things easier for student

2) Teacher might resolve complex regions

Why mimic models can be more accurate than training on original labels?

1) If labels have errors teacher may eliminate them making things easier for student

2) Teacher might resolve complex regions

3) Learning from probabilities is easier

Why mimic models can be more accurate than training on original labels?

1) If labels have errors teacher may eliminate them making things easier for student

2) Teacher might resolve complex regions

3) Learning from probabilities is easier

4) For student all outputs have "reason", teacher might encounter unexplainable things

Why mimic models can be more accurate than training on original labels?

1) If labels have errors teacher may eliminate them making things easier for student

2) Teacher might resolve complex regions

3) Learning from probabilities is easier

4) For student all outputs have "reason", teacher might encounter unexplainable things

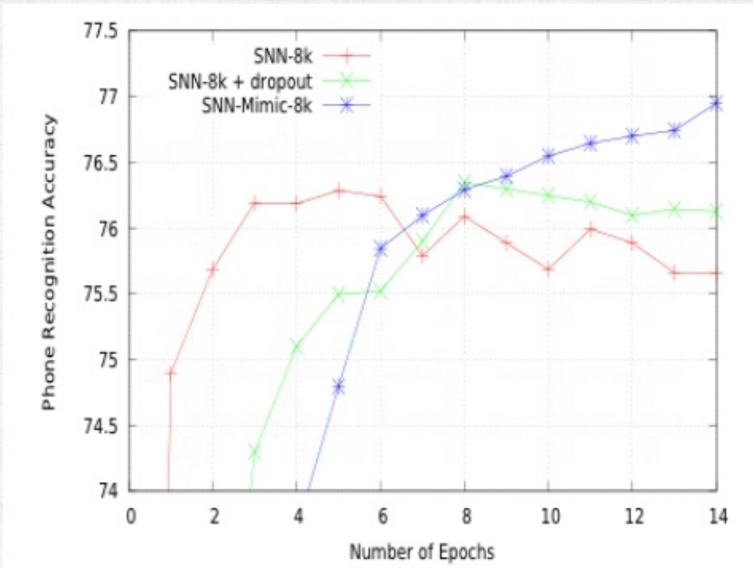


Figure 2: Shallow mimic tends not to overfit.

The capacity and representational power of shallow models.

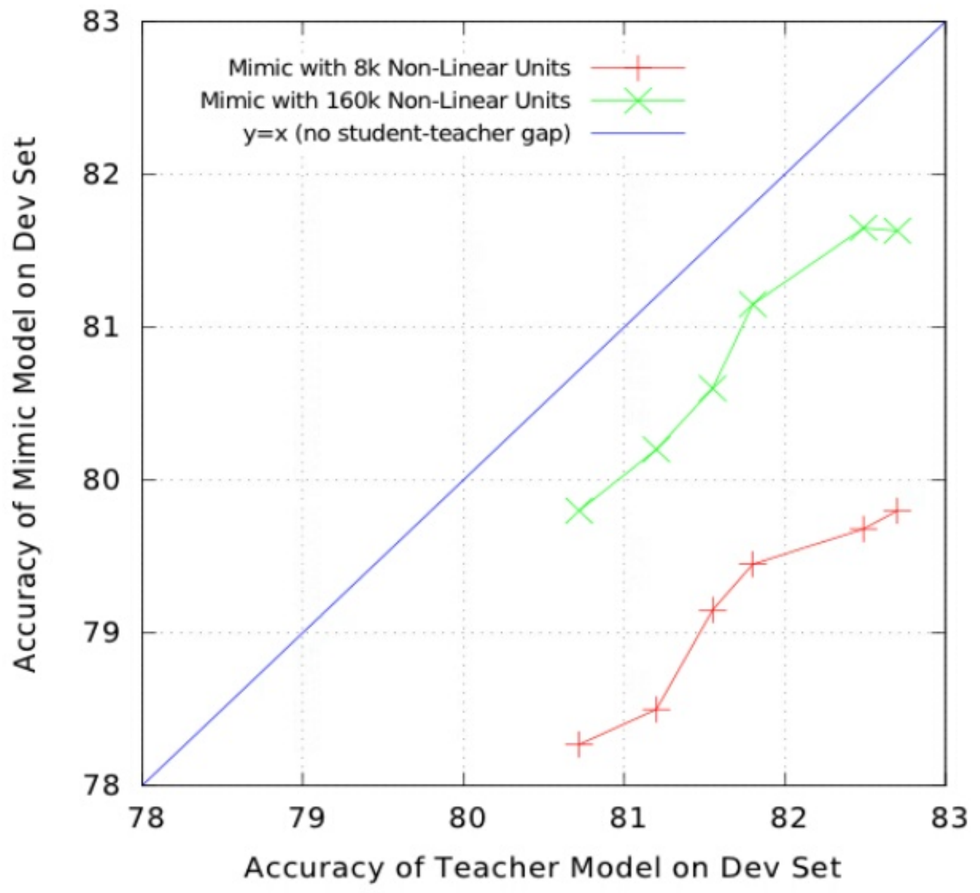


Figure 3: Accuracy of student models continues to improve as accuracy of teacher models improves.

The capacity and representational power of shallow models.

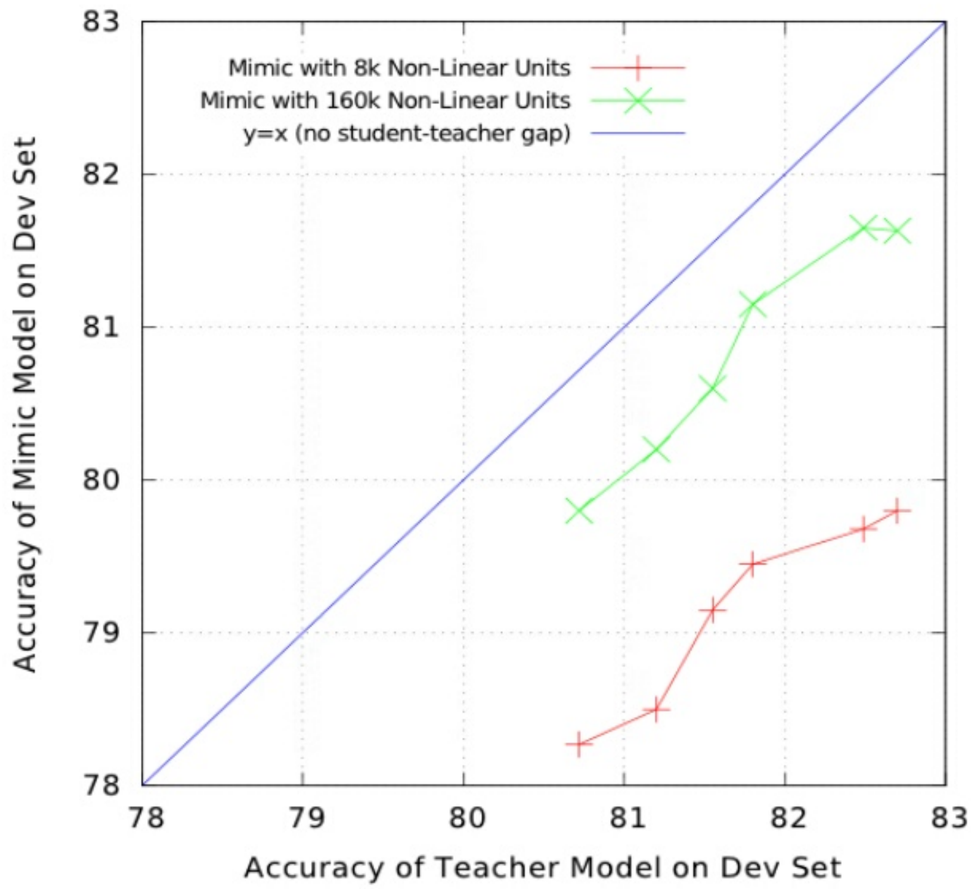


Figure 3: Accuracy of student models continues to improve as accuracy of teacher models improves.

"We see little evidence that shallow models have limited capacity or representational power. Instead, the main limitation appears to be the learning and regularization procedures used to train the shallow models"