Coded ResNeXt

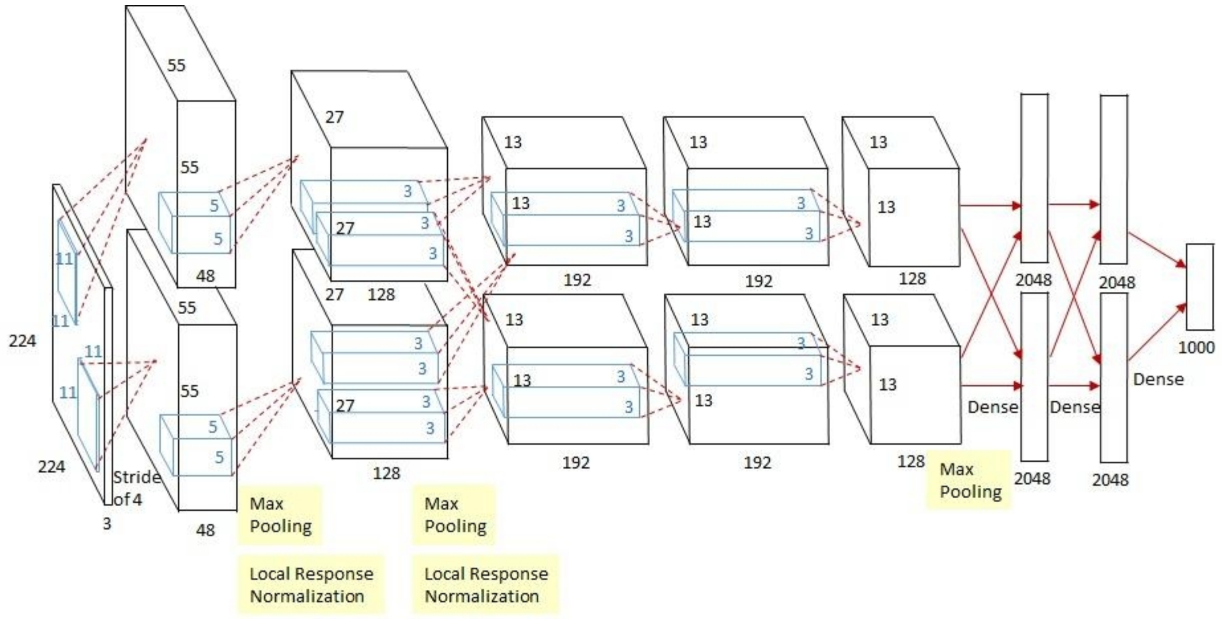
Apostolos Avranas, Marios Kountouris

Main Question

What is the purpose of multi-branch architectures?

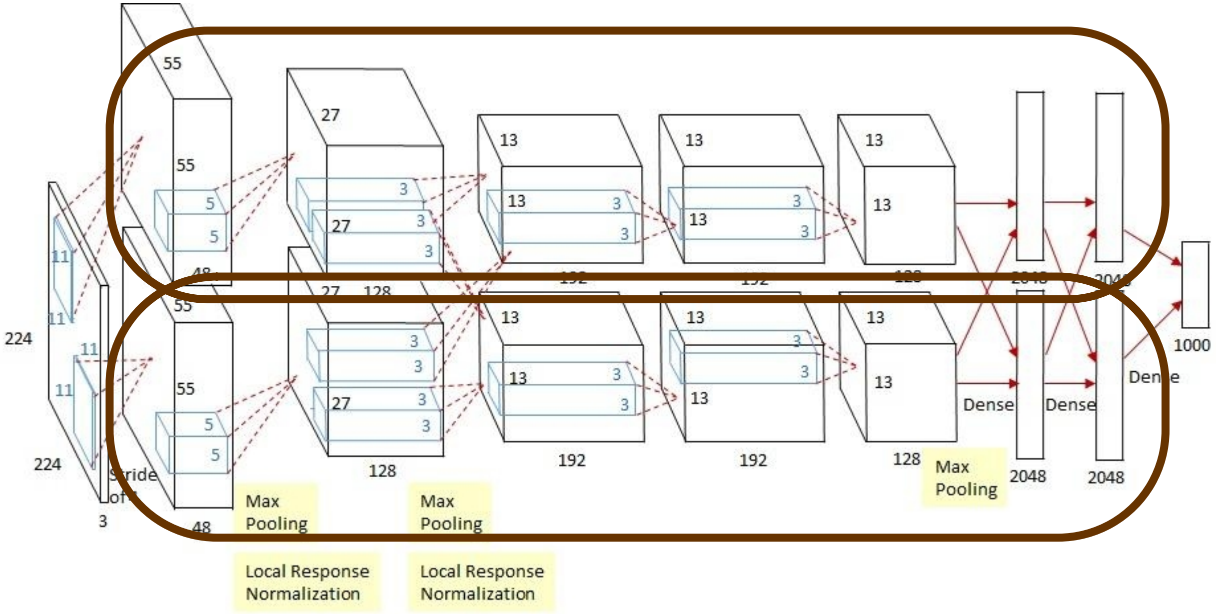
It began with AlexNet

(2012) Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton
Won *ImageNet Large Scale Visual Recognition Challenge* 2012



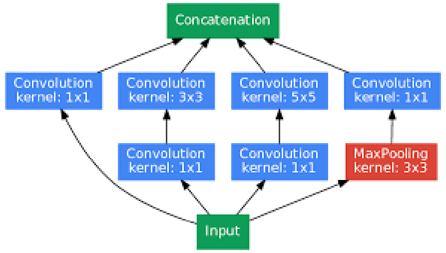
It began with AlexNet

Used two branches.
Primary motivation was to allow the training of the network over two **GPUs with 1.5GB** of **memory** each. (2012…)

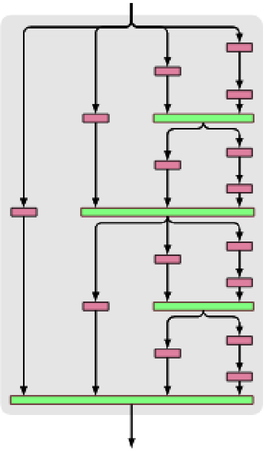


Motivation for using multi-branches for the rest?
Accuracy

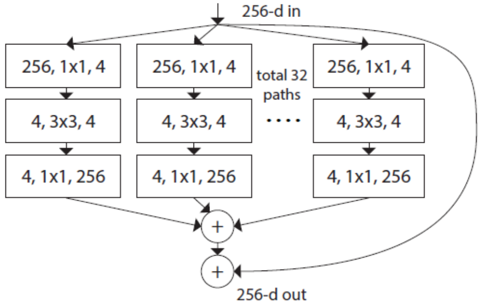
Inception (2015)



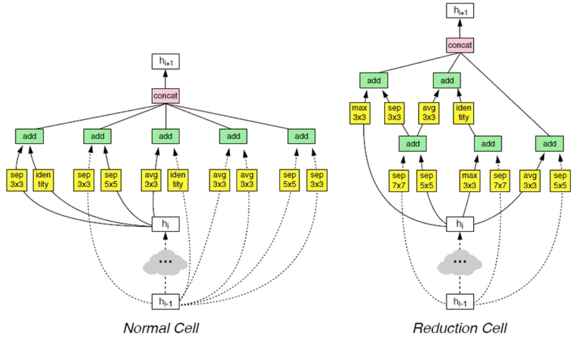
Fractal Net (2017)

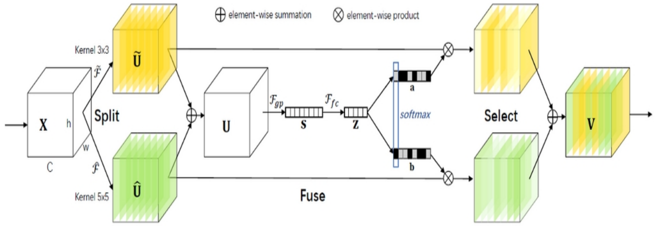


ResNeXt (2017)

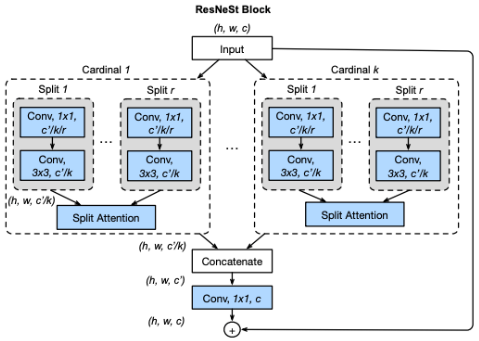


NasNet(2017)

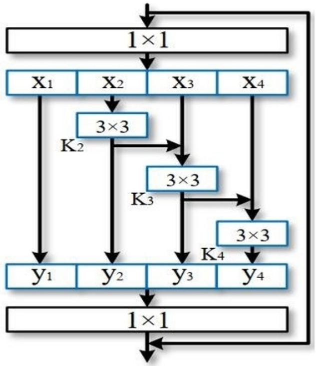




Selective Kernel Networks (2019)



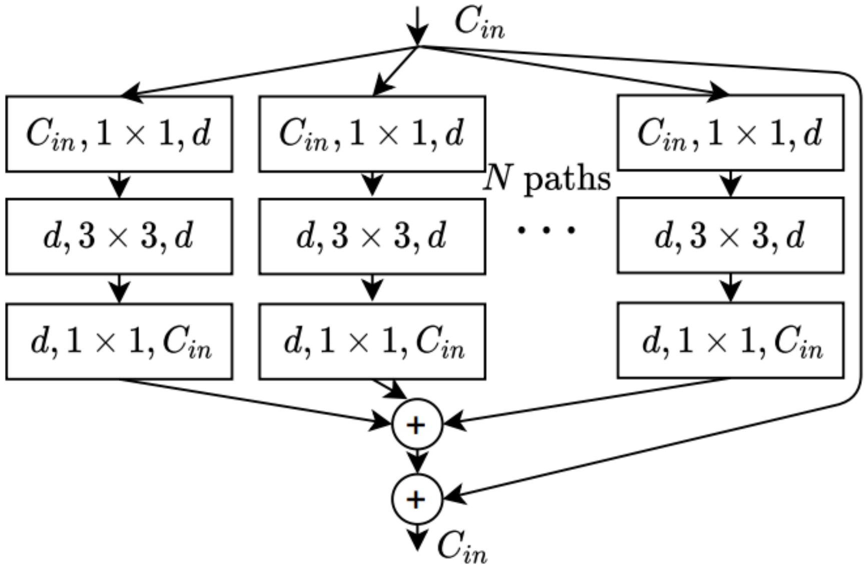
ResNeSt (2020)



Res2Net (2021)

Our Motivation?

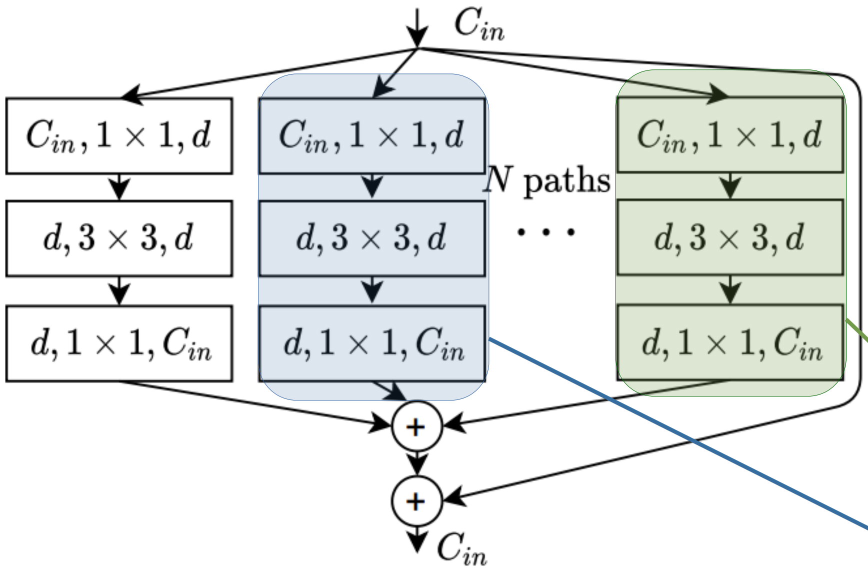
Each branch work for different set of classes.



Build upon the popular *ResNeXt.*

Our Motivation?

Each branch work for different set of classes.

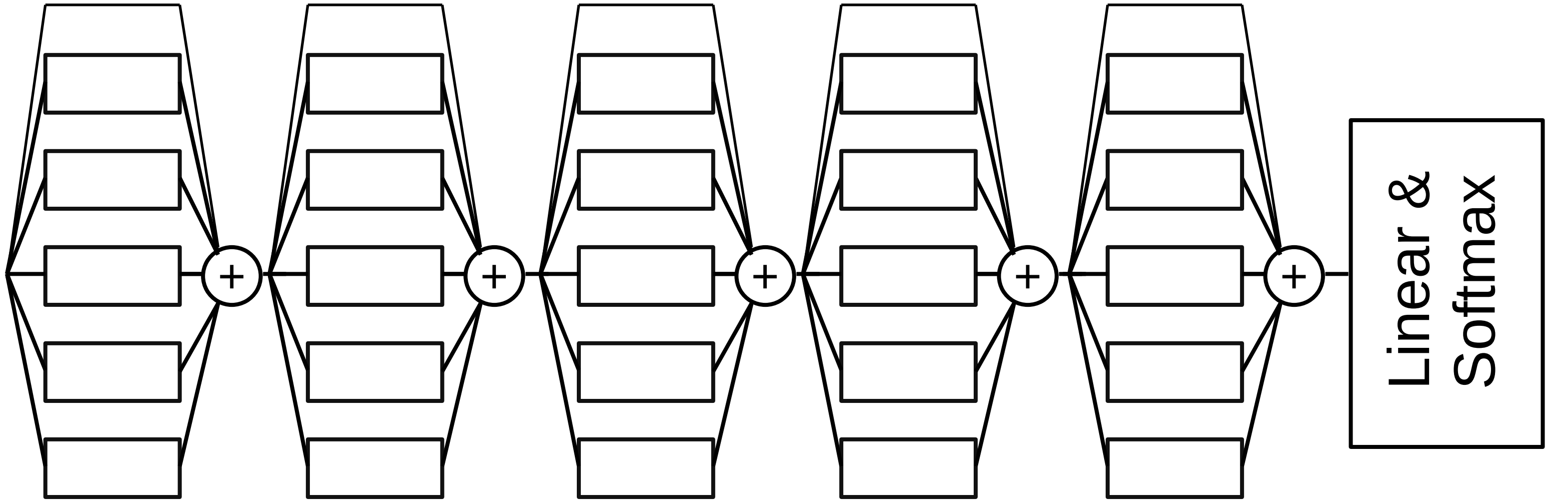


Build upon the popular *ResNeXt.*

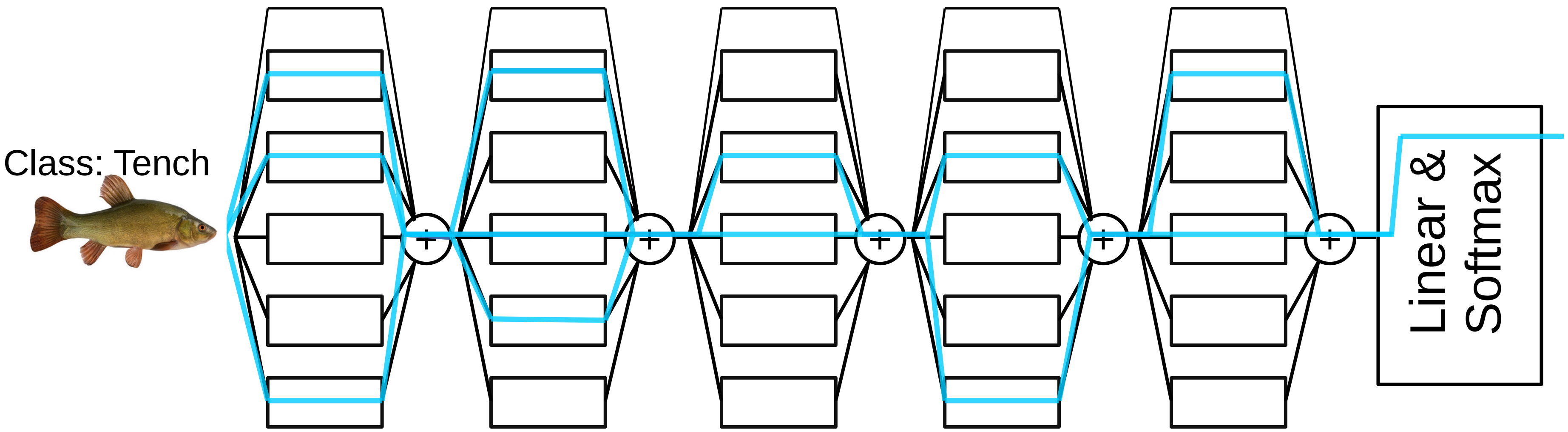
Works for (Bird, Horse, Ship)

Works for (Automobile, Horse, Truck)

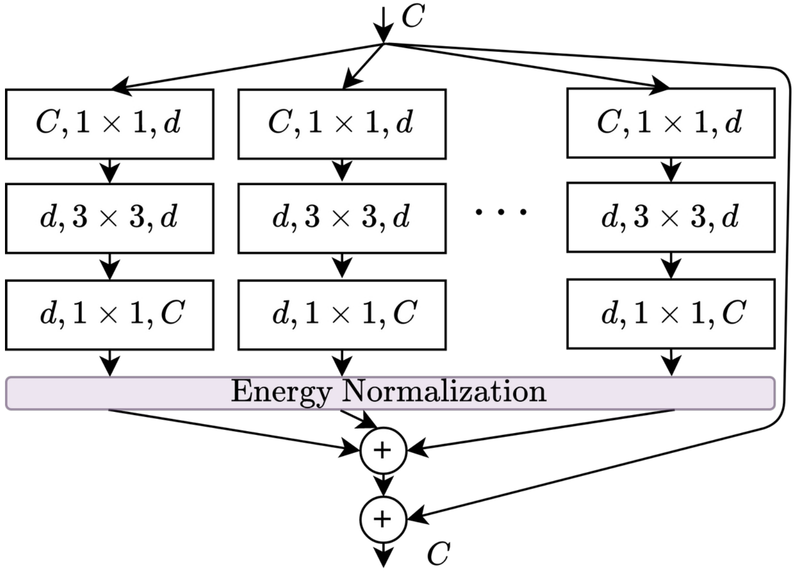
Big Picture: Ability to design *before training* paths through which per class information flows



Big Picture: Ability to design *before training* paths through which per class information flows



Changes
1 Architectural: Energy Normalization



ResNeXt:

$$For\ N\ branches: y = x + \sum_{n=1}^{N} \mathcal{T}_n(x)$$

With Energy Normalization:

$$y = x + \sum_{n=1}^{N} y_n$$

$$with\ y_n = \frac{\sum_{n=1}^{N} \mathcal{T}_n(x)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N} \mathcal{E}(\mathcal{T}_i(x))}}\ the\ output\ of$$

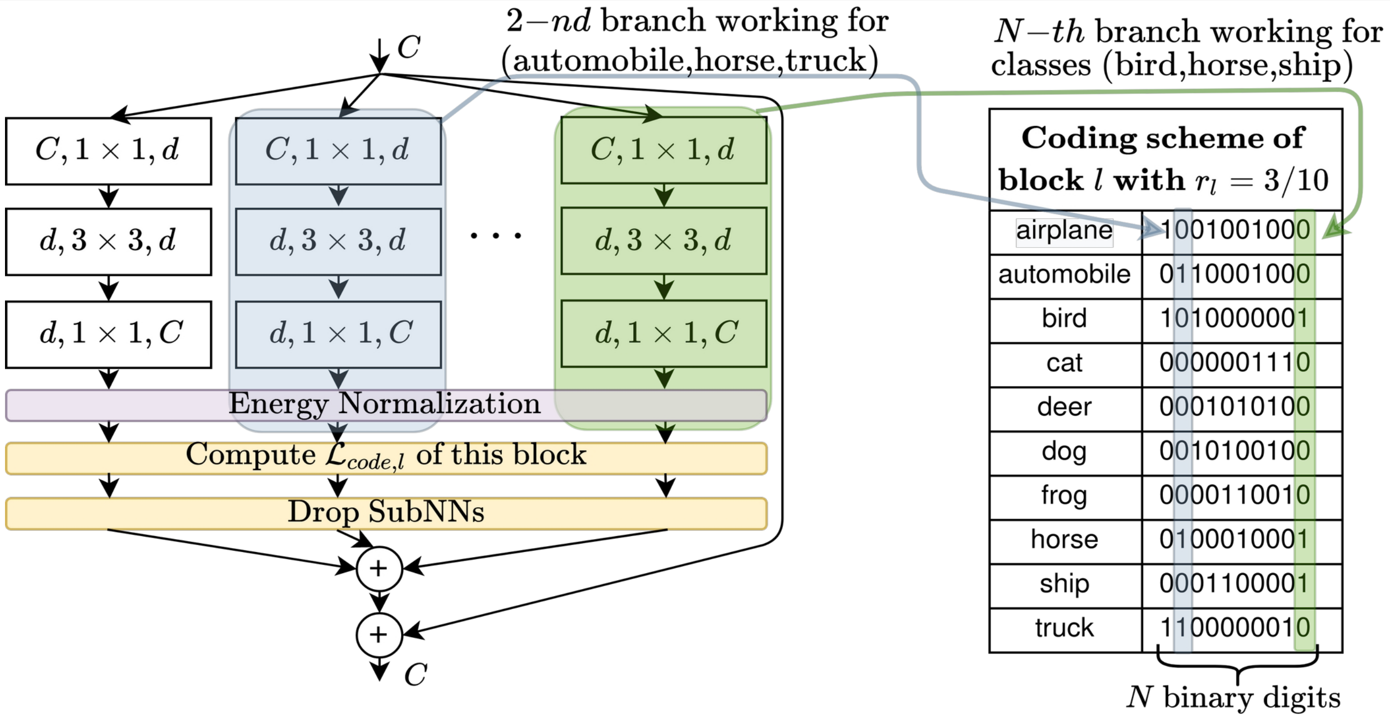
$$n\text{-}th\ branch\ and\text{:}\ \mathcal{E}(\cdot) \triangleq Energy\ of\ signal$$

Purpose: The total energy is constrained since $\sum_{n=1}^{N} \mathcal{E}(y_n) = N$.

Increasing the output energy of one branch will decrease the output energy of the rest. Pushes only few branches to be activated.

Changes

2 Algorithmic: Drop SubNNs, Coding Loss



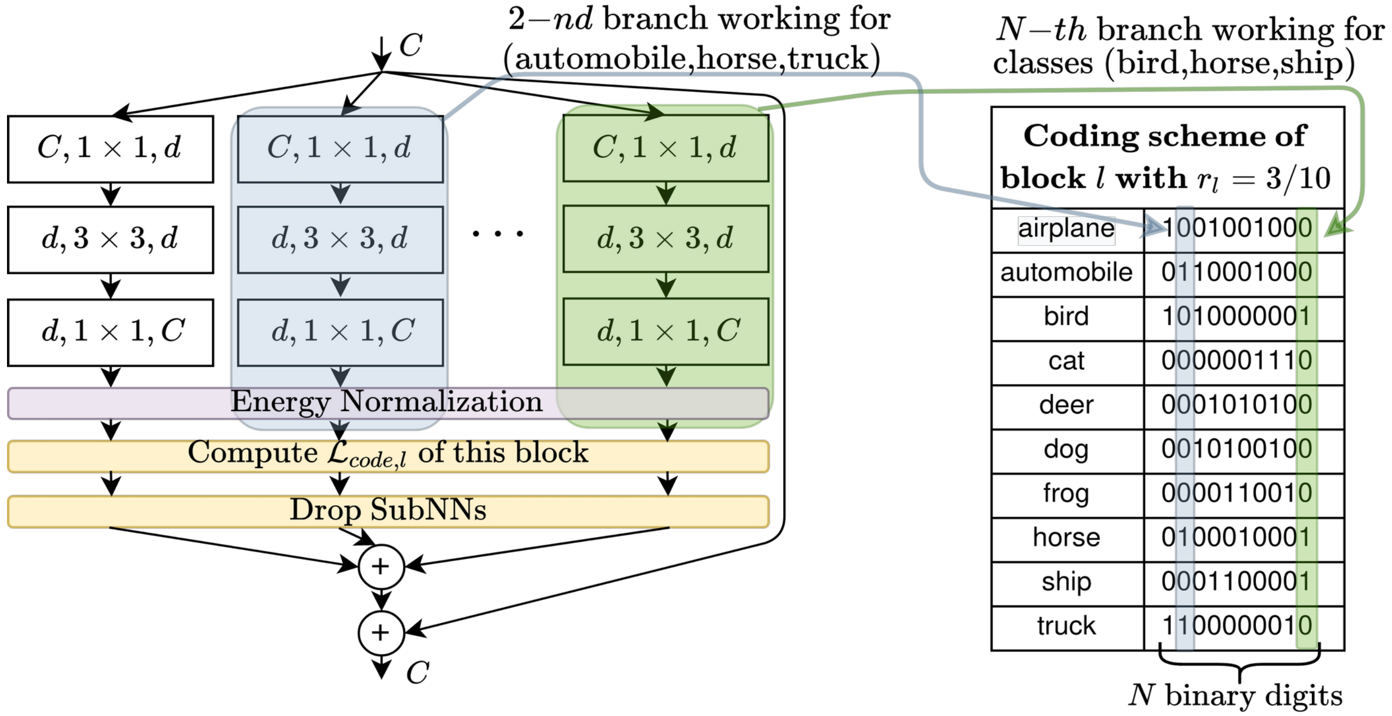
airplane	1001001000
automobile	0110001000
bird	1010000001
cat	0000001110
deer	0001010100
dog	0010100100
frog	0000110010
horse	0100010001
ship	0001100001
truck	1100000010

For the l-th block of the network, if the input sample belongs to class c, then

$$\text{coding loss:}\;\; \mathcal{L}_{code,l} = \begin{cases} \frac{1}{N}\sum_{n=1}^{N}(r_l\mathcal{E}(y_n))^4, & n \text{ branch does not work for class } c \\ \frac{1}{N}\sum_{n=1}^{N}(r_l\mathcal{E}(y_n)-1)^4, & n \text{ branch work for class } c \end{cases}$$

Changes

2 Algorithmic: Drop SubNNs, Coding Loss



Coding scheme of block l with $r_l = 3/10$	
airplane	1001001000
automobile	0110001000
bird	1010000001
cat	0000001110
deer	0001010100
dog	0010100100
frog	0000110010
horse	0100010001
ship	0001100001
truck	1100000010

Coding Loss: Push energy of inactive branches to 0 and active to 1.
Drop SubNNs: With some probability it makes zero the output of a branch.
Prevents co-adaptation of branches.

Leveraging *Coding Theory* *to a*ssign branches to classes

- Assigning branches to classes *before starting the training,* and
- in an *agnostic* to the semantic similarity of the classes way. We show show that branches can specialize to specific sets of classes *even if those sets are not designed based on the semantics of each class.*

Leveraging *Coding Theory* *to a*ssign branches to classes

- Assigning branches to classes *before starting the training,* and
- in an *agnostic* to the semantic similarity of the classes. We show show that branches can specialize to specific sets of classes even if those sets are not designed based on the semantics of each class.

Coding scheme of a block l with $r_l = 3/10$	
airplane	1001001000
automobile	0110001000
bird	1010000001
cat	0000001110
deer	0001010100
dog	0010100100
frog	0000110010
horse	0100010001
ship	0001100001
truck	1100000010

N binary digits

Three rules to assign a class to branches:

1) Equal Number of ‘‘1’’ per codeword. Assigning the same computational resources per class.
2) Equal Number of ‘‘1’’ per column. The same load of work (number of classes) assigned to each branch.
3) The minimum Hamming distance between two codeword to be as high as possible.

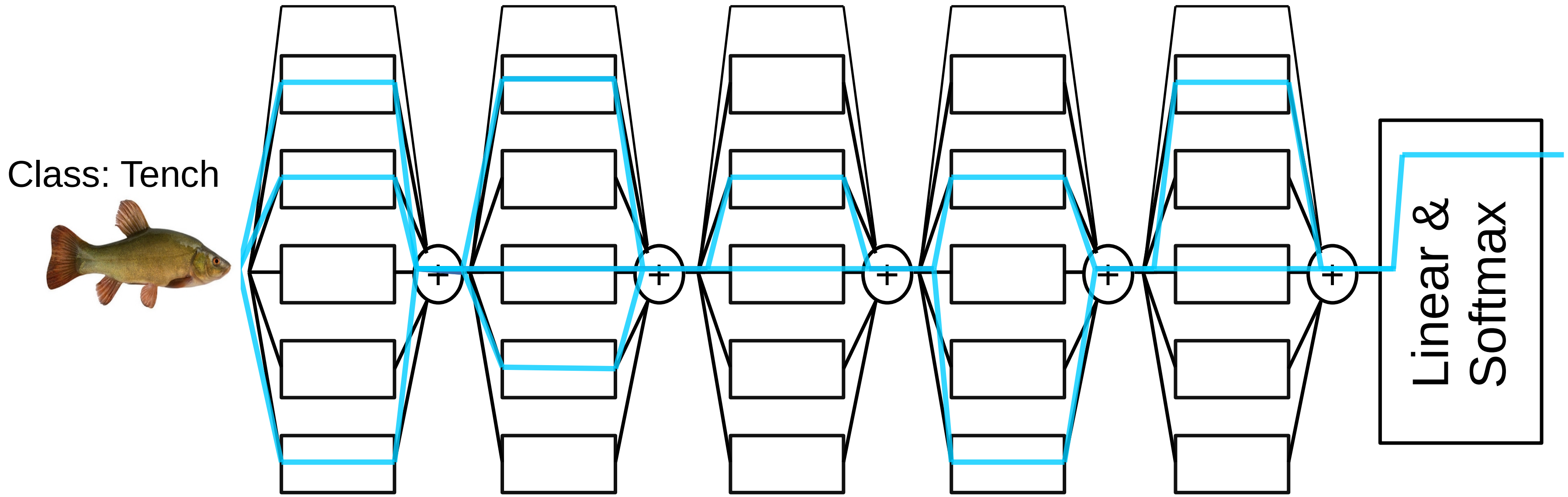
Architectures. The deeper the block the more specialized branches.

stage	**Coded ResNeXt-29 (10×11d)** for CIFAR-10	**Coded ResNeXt-29 (20×6d)** for CIFAR-100	**Coded ResNeXt-50 (32×4d)** for ImageNet
c1	conv $3\times3, 64$	conv $3\times3, 64$	conv $7\times7, 64$, str. 2 3×3 max pool, str. 2
c2	$\begin{bmatrix} 256, & 11, \\ 10/10 \end{bmatrix}\times3$	$\begin{bmatrix} 256, & 6, \\ 20/20 \end{bmatrix}\times3$	$\begin{bmatrix} 256, & 4, \\ 32/32 \end{bmatrix}\times3$
c3	$\begin{bmatrix} 512, & 22, \\ \mathbf{5/10} \end{bmatrix}\times3$	$\begin{bmatrix} 512, & 12, \\ \mathbf{8/20} \end{bmatrix}\times3$	$\begin{bmatrix} 512, & 8, \\ 32/32 \end{bmatrix}\times4$
c4	$\begin{bmatrix} 1024, & 44, \\ \mathbf{3/10} \end{bmatrix}\times3$	$\begin{bmatrix} 1024, & 24, \\ \mathbf{4/20} \end{bmatrix}\times3$	$\begin{bmatrix} 1024, & 16, \\ \mathbf{16/32} \end{bmatrix}\times6$
c5	global avg. pool 10-d fc, softmax	global avg. pool 100-d fc, softmax	$\begin{bmatrix} 2048, & 32, \\ \mathbf{8/32} \end{bmatrix}\times3$
			global avg. pool 1000-d fc, softmax

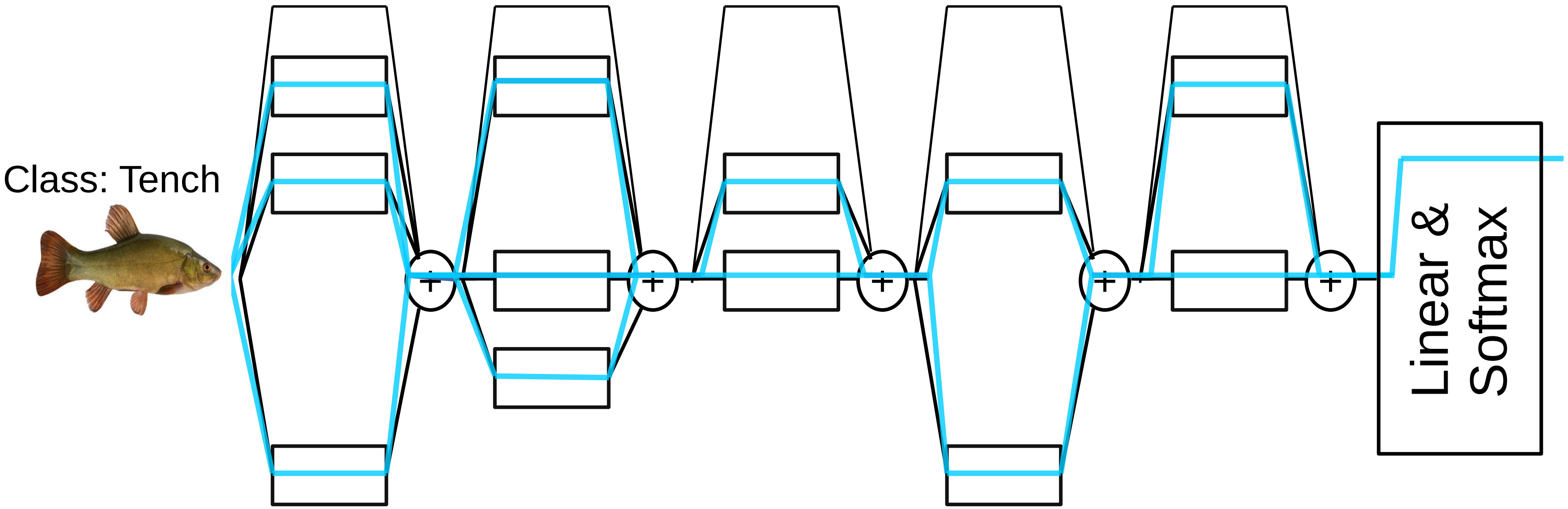
$$r_l = \frac{Number\ working\ branches\ per\ class}{N}$$

With bold is depicted the r_l.
The deeper in the architecture the smaller it gets, so each class has fewer but more specialized branches working for it.

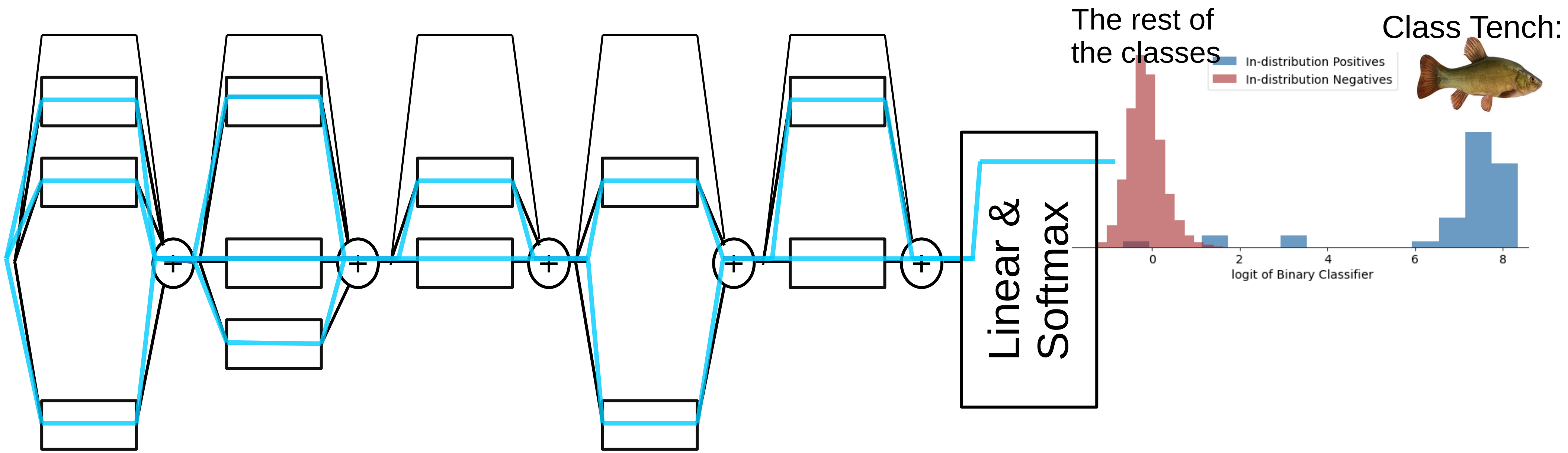
Did we achieve information to flow only through the designated paths?



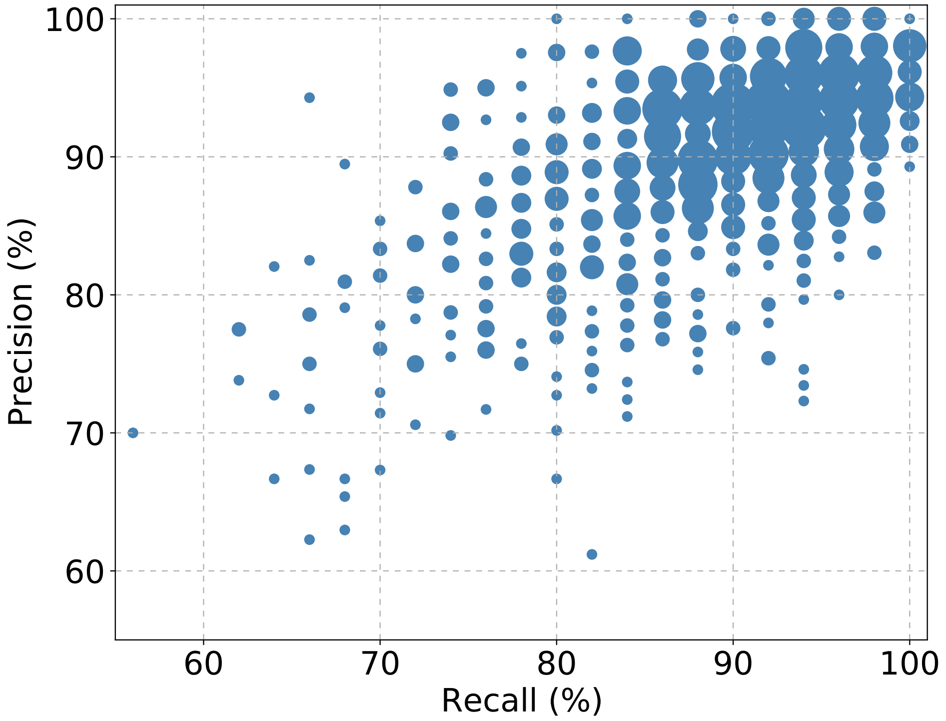
Did we achieve information to flow only through the designated paths?



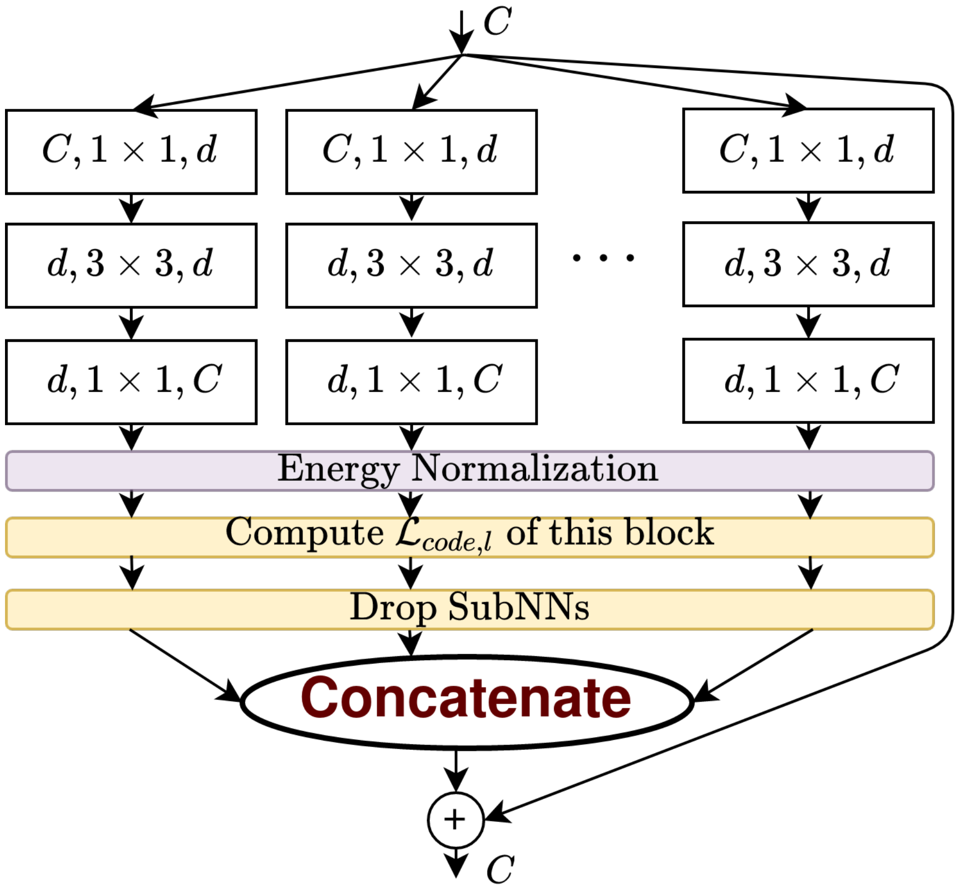
Did we achieve information to flow only through the designated paths?



With a single training 1000 lighter (<40% #params) single purposed classifiers



What if Concatenation instead of Summation?



It does not work…
But why?

What about accuracy?

	Timm ResNeXt-50 (Res: 224)
ImageNet (2012)	79.76%

Hyperparameters:

Epochs 250, Learning Rate=0.6, Batch-size=192 per core (8 cores in total), Cosine Scheduler, 5 epochs warmup and 10 cooling down, RandAugment (N,M)=(2,7) with noise of std 0.5 added in magnitude, Random erasing probability 0.4 and recount 3

What about accuracy?

	Timm ResNeXt-50 (Res: 224)	ResNeXt-50 (Res: 160)
ImageNet (2012)	79.76%	79.50%

What about accuracy?

	Timm ResNeXt-50 (Res: 224)	ResNeXt-50 (Res: 160)	Coded ResNeXt-50 (Res: 160)
ImageNet (2012)	79.76%	79.50%	80.21%

What about accuracy?

	Timm ResNeXt-50 (Res: 224)	ResNeXt-50 (Res: 160)	Coded ResNeXt-50 (Res: 160)
ImageNet (2012)	79.76%	79.50%	**80.21%**

	ResNeXt-29	Coded ResNeXt-29
CIFAR-10	93.66%	**94.41%**
CIFAR-100	76.82%	**78.28%**

What if Coding loss
with power of 2 or absolute value?

$$\mathcal{L}_{code,l} = \begin{cases} \frac{1}{N}\sum_{n=1}^{N}(r_l\mathcal{E}(y_n))^2, & n \text{ branch does not work for class } c \\ \frac{1}{N}\sum_{n=1}^{N}(r_l\mathcal{E}(y_n)-1)^2, & n \text{ branch \quad work for class } c \end{cases}$$

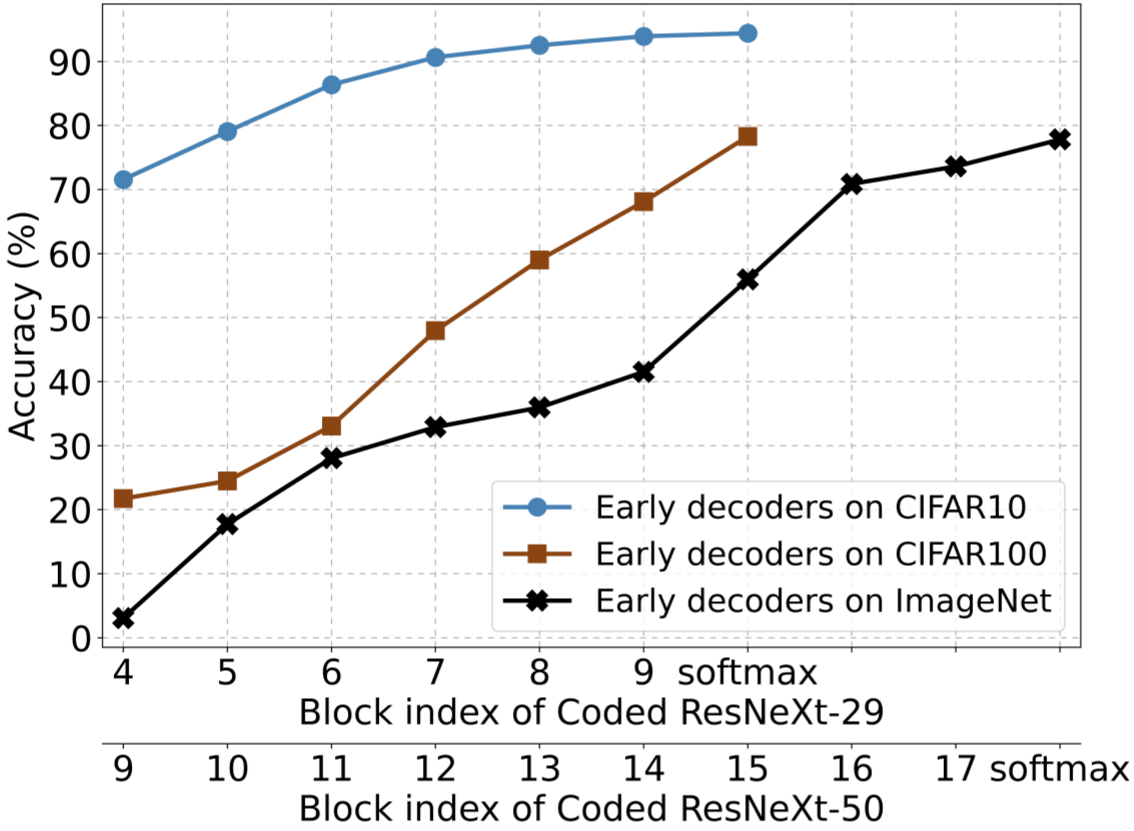
or

$$\mathcal{L}_{code,l} = \begin{cases} \frac{1}{N}\sum_{n=1}^{N}|r_l\mathcal{E}(y_n))|, & n \text{ branch does not work for class } c \\ \frac{1}{N}\sum_{n=1}^{N}|r_l\mathcal{E}(y_n)-1|, & n \text{ branch \quad work for class } c \end{cases}$$

It does not work…
But why?

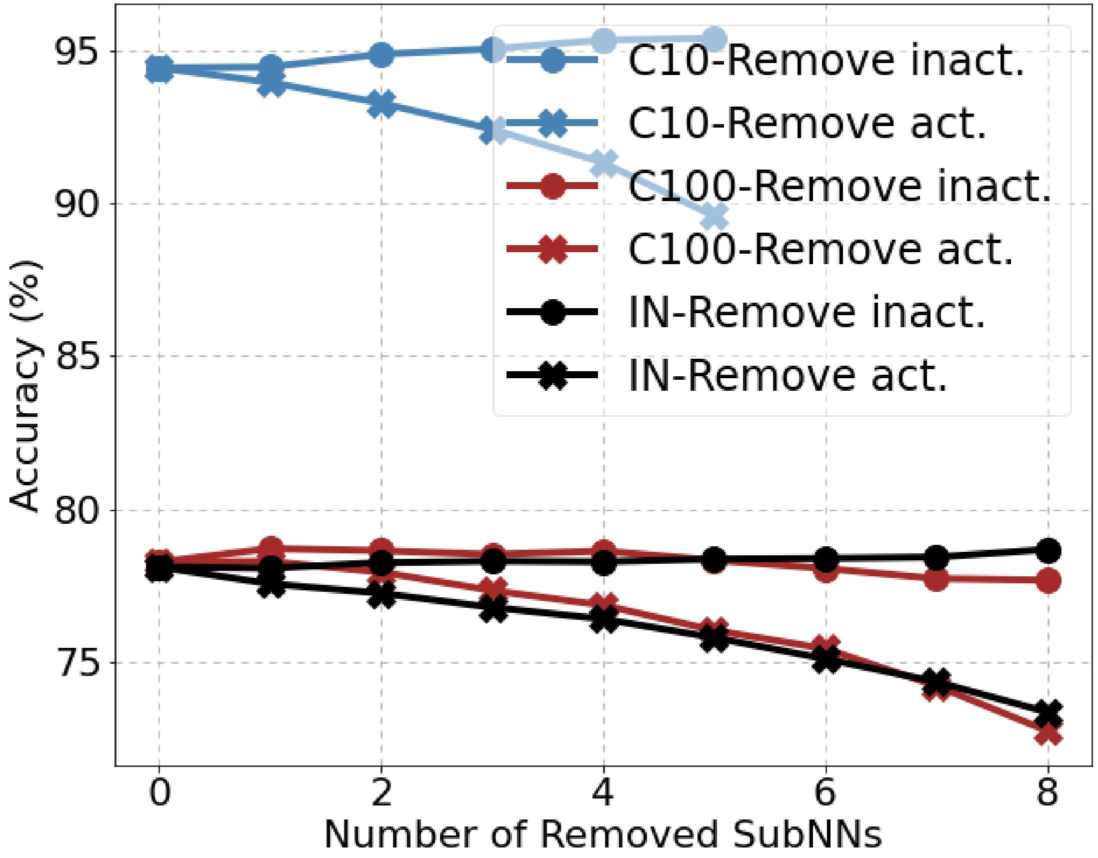
Thank you

Early decoding



Looking at a specific block, the energies of the output of the branches of that block have to agree with the codewords. Therefore, when inputting a sample, we can measure the energies of the branches of a block and try to see to which codeword they are ''closer''. That codeword correspond to the class which the early decoder will predict.

Removing branches from specific block



Looking at a specific block and sample, if we remove branches from that block that are assigned to stay active for the class of the sample, then the accuracy drops. If the branches are assigned to be inactive the accuracy increases!