Compound CNNs For Improved Classification Accuracy

Presenter: Spyros Tragoudas[*]
Joint work with: Vasileios Pentsos[*], Bijay Raj Paudel[*], Kiriti Nagesh Gowda[†], Mike Schmit[†]

[*]School of Electrical, Computer and Biomedical Engineering, Southern Illinois University, Carbondale, IL
[†]ML Computer Vision Group, Advanced Micro Devices, Inc. Santa Clara, CA

Overview

- Introduction
- The proposed mechanism
- The architecture and justification
- Example cases
- Experimental results
- Conclusion

Introduction

- **Observation**: A CNN cannot classify with equal accuracy across all classes it is trained on

TABLE I
CLASSIFICATION ACCURACY OF ESTABLISHED ARCHITECTURES ON POPULAR DATASETS.

dataset	# of classes	input network	# of classes below 20% percentile	# of conv. layers
CIFAR-10	10	ResNet-18	2	16
		VGG16	2	13
CIFAR-100	100	ResNet-18	25	16
		VGG16	22	13
Tiny-ImageNet	200	ResNet-18	36	16
		VGG16	48	13
MNIST	10	ResNet-18	2	16
		VGG16	1	13

- **Observation**: **Improved accuracy with CNN that focus on classes sharing similar features**
  Example case - network trained on classes "cats" and "dogs" on Cifar-10, 1000 images per class, original network VGG16:
    - Reduced "dog" misclassification 110 to 74
    - Reduced "cat" misclassification 130 to 3

SIU Southern Illinois University CARBONDALE

AMD

Introduction

Contribution

- A method to improve the accuracy of a Convolutional Neural Network (CNN) by adding shallow CNNs without increasing the inference time.
- Can be used on any already trained CNN, regardless of its complexity or accuracy.
- Does not require retraining of the original CNN or customizing datasets.

Novelty

- Improve classification accuracy for classes where the input CNN underperforms.
- Shallow CNNs per class that operate concurrently, reduce the number of false positives for the class, and may defer classification to the input CNN.

CNNs for low accuracy classes

- Automated method to select classes using the confusion matrix M_{ij} of the original input CNN, which has been trained on all classes.
- Shallow CNN for each such class *i* (anchor) also consider other classes (supporting) using threshold θ.

Example: Confusion Matrix of Resnet18 on CIFAR10, with θ = 12

Predicted class *i*

True class *j*

	0	1	2	3	4	5	6	7	8	9
0	948	1	14	7	3	0	2	2	17	6
1	3	979	0	0	1	0	1	0	0	16
2	13	0	931	12	16	11	13	2	2	0
3	6	2	12	876	16	64	15	5	3	1
4	6	0	14	7	954	6	5	7	0	1
5	5	0	12	55	13	908	0	6	0	1
6	5	0	12	14	4	5	959	0	0	1
7	3	1	8	9	15	7	0	955	0	2
8	15	6	2	2	0	1	1	1	961	11
9	7	26	2	2	0	0	0	0	6	957

Classes in the swallow CNN for row *j*

Green cells: **Correctly classifications of class *j***
Light green cells (supporting classes of *j*): **Misclassifications of class *j* that exceed θ**
White cells: **Misclassifications of class *j* below θ**

The *union* of the red and the green cells defines a *set of selected classes* in a swallow CNN for *i*

For instance, the set of selected classes for class "3" is {3, 4, 5, 6}

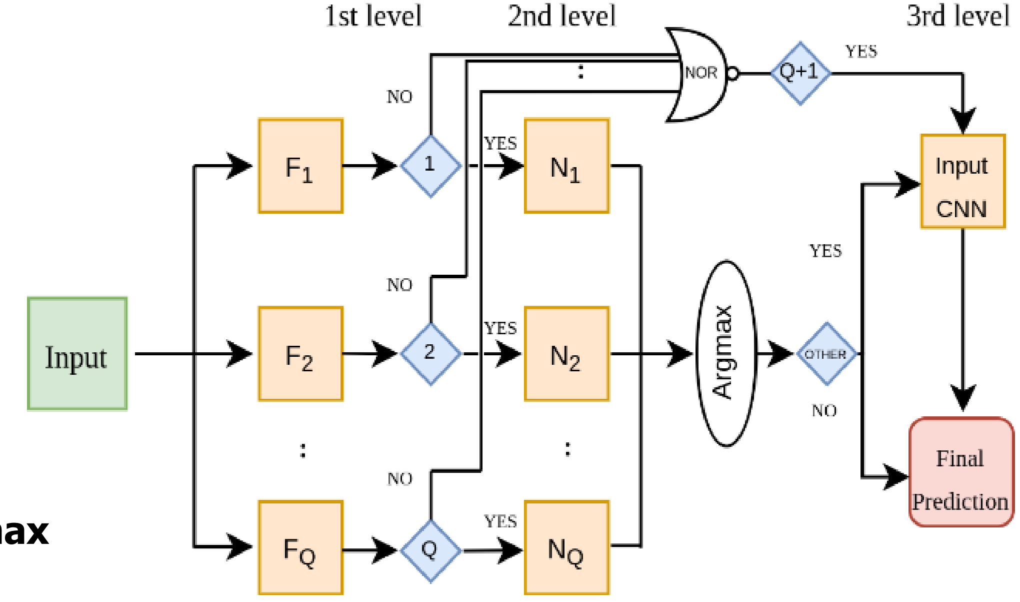
The proposed mechanism

A pair of shallow CNNs for each low accuracy class: A classification network N_i and a filter network F_i

- **1st level -** Filters F_i are binary networks operating in parallel, assuming a multi-processor device.
  - If input is predicted by F_i to be anchor class *i* it is directed to N_i, else to the original input CNN
- **2nd level -** Classification networks N_i. Available predictions:
  - The anchor class *i*
  - One of the supporting classes
  - The "other" class

Example: Cifar-10 with Resnet-18

Anchor class	Supporting classes	Classes that form class "other"
3	4,5,6	0,1,2,7,8,9



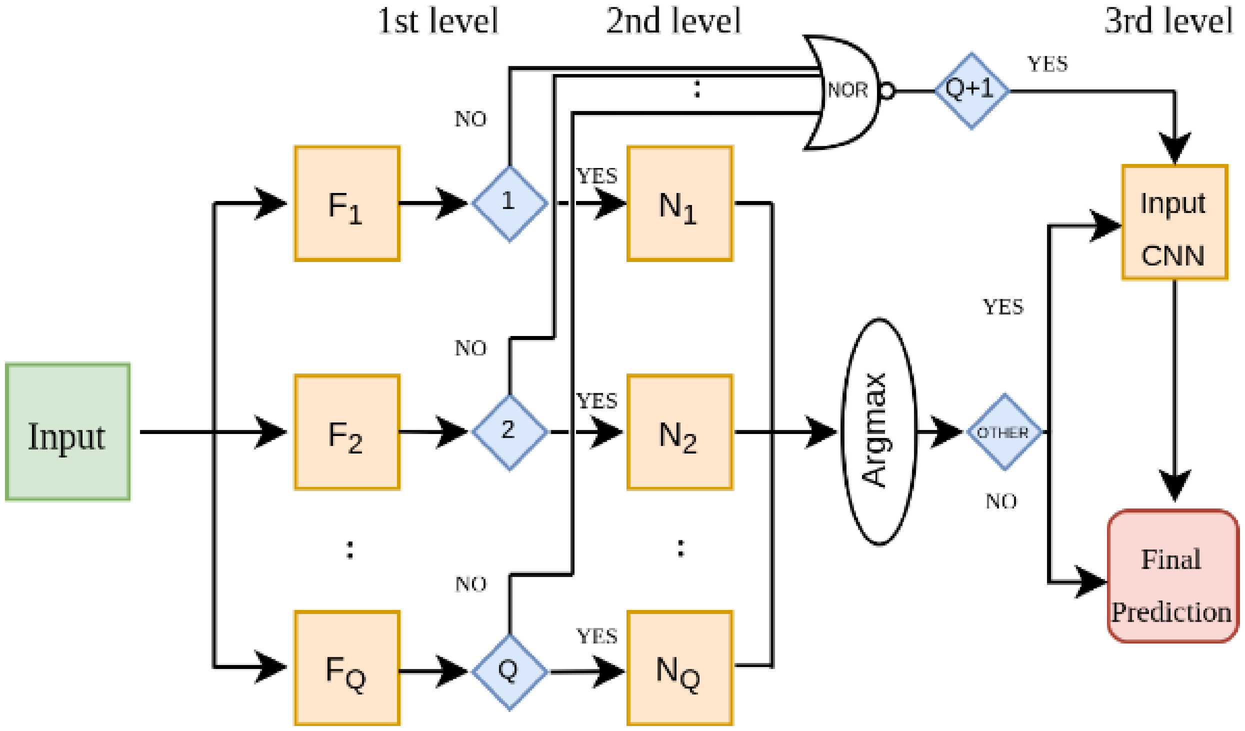
Enabled N_i are also fired *in parallel*. Final class prediction by module Argmax is the one with the *highest probability*.

SIU Southern Illinois University CARBONDALE AMD

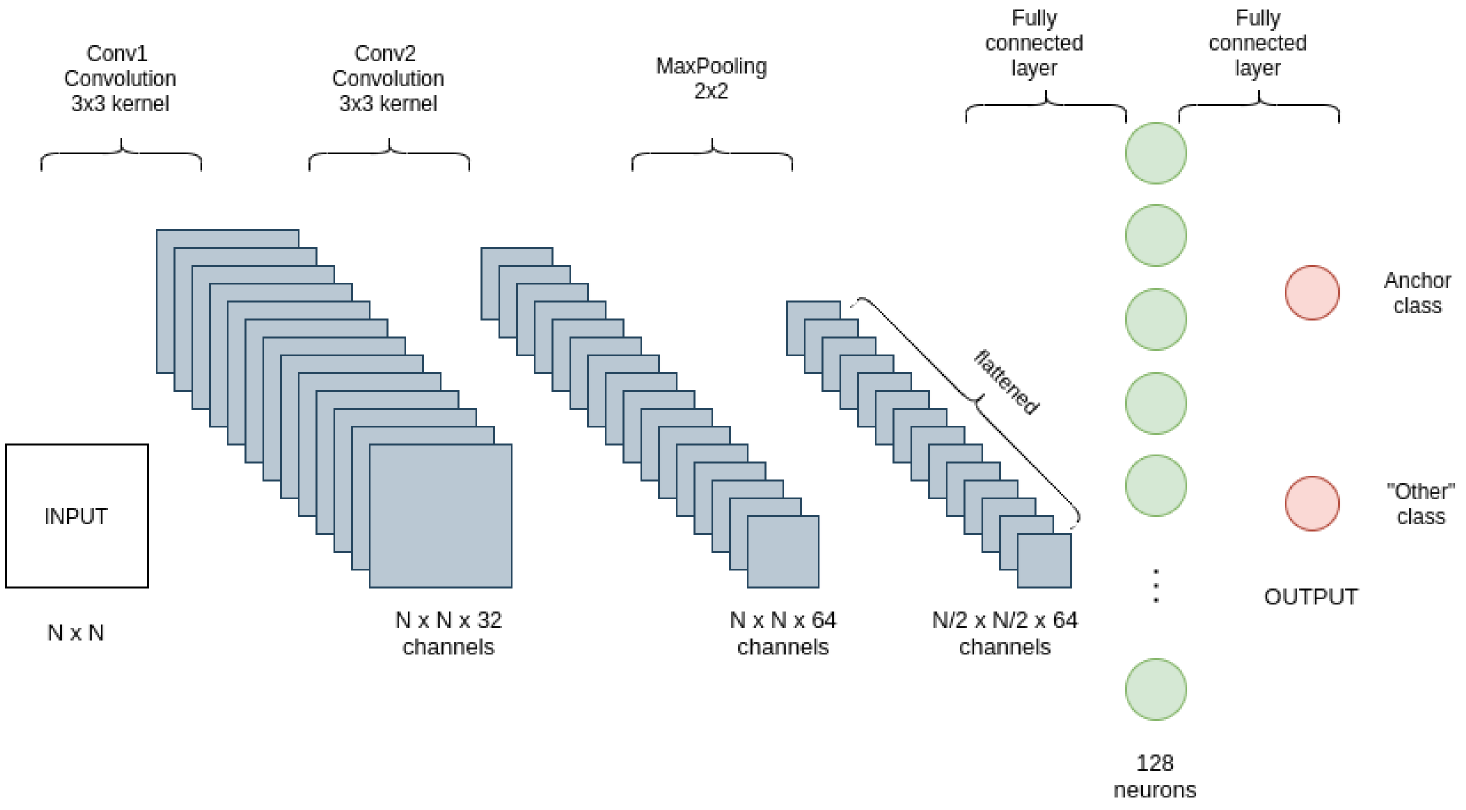
The schematic of the proposed mechanism

- **3^{rd} level -** The original input CNN that is enabled only when all F_i predictions are negative or when prediction of an N_i is "other"

- **Rectangles:** Represent CNNs.
  - F_i and N_i are shallow CNN with inference time less than that of the input CNN.
- **Diamonds:** Represent Binary Decision logic modules.
  - NOR gates and the Argmax module that have negligible inference time.



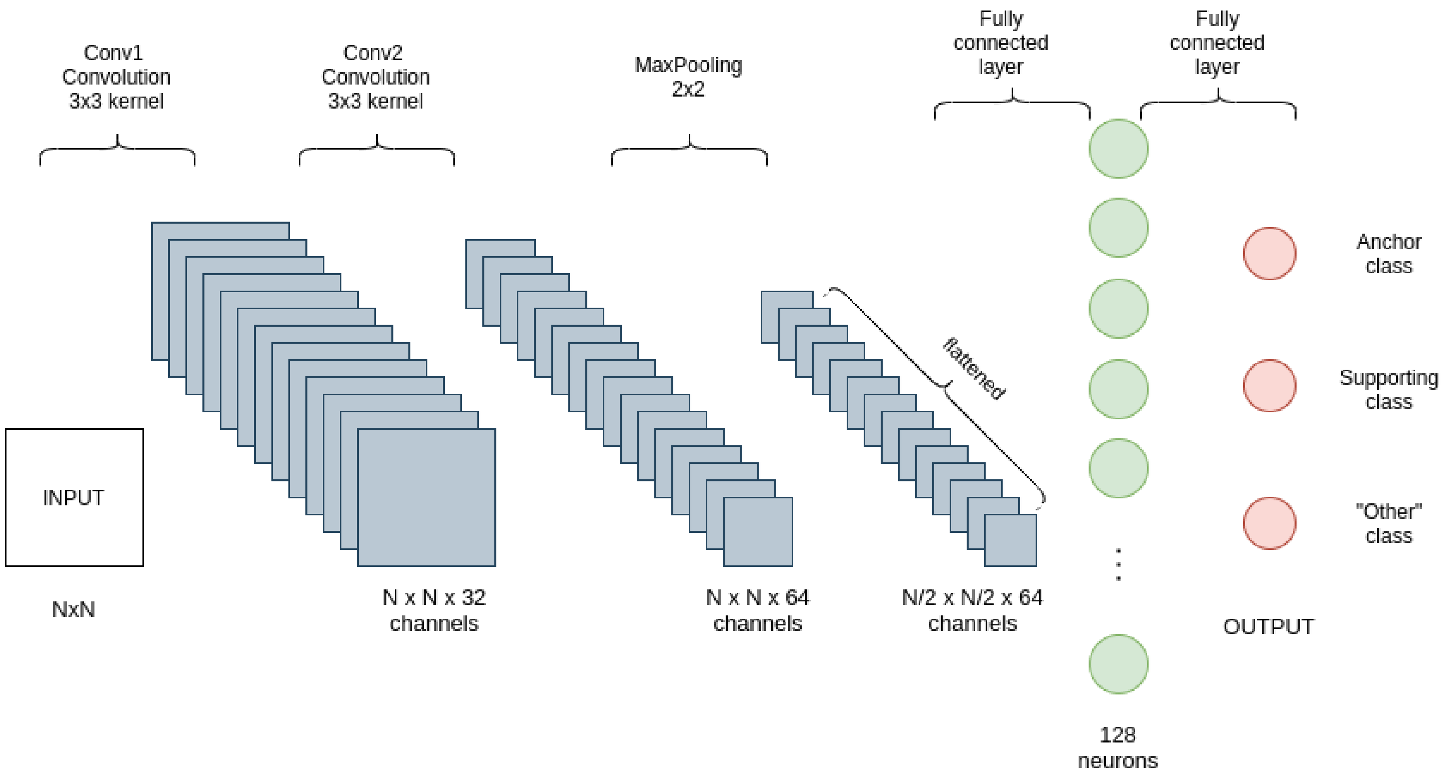
F_i network architecture



F_i network architecture

- F_i networks are shallow by design, with only 2 Convolutional layers
- In contrast, the input CNNs VGG16 and ResNet-18 have 13 and 16 convolutional layers respectively
- Each F_i classifies the input as belonging to a certain low-accuracy class or not

N_i network architecture



N_i classification network architecture

- N_i networks are also shallow by design, with only 2 Convolutional layers
- In contrast, the input CNNs VGG16 and ResNet-18 have 13 and 16 convolutional layers respectively
- Each N_i has three possible outputs
- The computational complexity (in number of trainable parameters) of each F_i - N_i pair is **69.1% - 91.3% lower** than the input CNNs

Justification of the proposed structure

- The F_i networks at the 1^{st} level are designed to isolate each low-accuracy class from the total dataset
- However, the false positives affect the accuracy at the 1^{st} level
- The N_i networks at the 2^{nd} level are designed to distinguish among a low-accuracy class and its most frequent false positive classes
- N_i trained to handle a specific subset of the dataset which share similar features, not the whole dataset
- The input CNN on the 3^{rd} level acts as a safety-net for those cases that fooled F_i and N_i handling the classes as well

Illustration

Input architecture CNN ResNet-18 and dataset CIFAR-100

Class 11 is “boy” and class 35 is “girl”.

- For an input with true class 11, filters F11 and F35 both predicted positive (“YES”).
- They enabled their respective classification networks N11 and N35.
- N11 predicted class “boy” with probability of 64.2 % and N35 predicted class “girl” with probability 36.1 %.

True class	Class	Filter F_i prediction	N_i prediction	N_i probability (%)
boy	boy	yes	boy	64.2
	girl	yes	girl	36.1

- Final prediction by module Argmax was class “boy”.

Illustration

Input architecture CNN is VGG16 and dataset CIFAR-100

Class 11 is “boy”, class 35 is “girl”, and class 98 is “woman”.

- For an input with true class 11, F11 , F35 and F98 predicted positive (“YES”) and enabled N11, N35 and N98.
- Network N11 predicted “other” with probability of 72.8 %.
- Network N35 predicted class 35 with probability 54.5 %.
- Network N98 predicted the class 98 with probability 48.6 %.

True class	Class	F_i prediction	N_i prediction	N_i probability (%)	Input CNN prediction
Man	boy	yes	other	72.8	man
	girl	yes	girl	54.5	
	woman	yes	woman	48.6	

- Module Argmax chose class “other” and the image is directed to the input CNN that predicted class 11.

Results – classification accuracy

TABLE II
ACCURACY IMPROVEMENT ON THE SELECTED CLASSES OF THE INPUT CNN.

dataset	# of classes	input network	# of selected classes	selected classes (%)	gain in accuracy (%)	max possible gain in accuracy (%)	efficiency (%)
CIFAR-10	10	ResNet-18	2	100	4.785	7.527	63.57
		VGG16	2	100	4.012	9.027	44.44
CIFAR-100	100	ResNet-18	7	28	11.667	66.667	17.50
		VGG16	4	18	7.104	118.579	6.00
Tiny-ImageNet	200	ResNet-18	3	8.3	32.653	53.061	61.54
		VGG16	5	10.4	36.765	83.824	43.86
MNIST	10	ResNet-18	2	100	0.135	0.757	17.78
		VGG16	1	100	0.561	1.478	37.93

- $efficiency = \dfrac{accuracy\ gain}{max\ possible\ accuracy\ \ gain}$
- Selected classes (%)
  the (%) percentage of selected classes over total number of the low accuracy classes

- Max possible gain in accuracy (%)
  Gain in the accuracy of the selected classes, if all instances of the selected classes were correctly classified

Results – overall classification accuracy improvement

TABLE III
OVERALL ACCURACY IMPROVEMENT ON THE DATASETS

dataset	input network	gain in acc. (%)	max possible gain in acc. (%)	achieved acc. (%)
CIFAR-10	ResNet-18	1.04	2.8	95.32
	VGG16	1.61	4.14	94.52
CIFAR-100	ResNet-18	0.29	0.8	66.14
	VGG16	0.36	2.17	62.57
Tiny-ImageNet	ResNet-18	0.10	0.52	68.44
	VGG16	0.24	1.14	57.54
MNIST	ResNet-18	0.04	0.45	99.35
	VGG16	0.08	0.29	99.21

- Presented approach performed well on datasets with relatively small number of classes
- The accuracy of the original CNN: Achieved accuracy (5th column) - Gain in accuracy (3rd column)

Results – inference overhead

TABLE IV
INFERENCE OVERHEAD ON THE DATASETS

dataset	input network	inference time (s)	inference time increase (%)
CIFAR-10	ResNet-18	2.389	9.04
	VGG16	2.354	8.18
CIFAR-100	ResNet-18	1.622	-1.03
	VGG16	2.381	-1.93
Tiny-ImageNet	ResNet-18	3.319	-0.19
	VGG16	2.127	0.21
MNIST	ResNet-18	1.830	3.08
	VGG16	1.766	2.84

- All F_i and N_i networks run in parallel. Approach was implemented with PyTorch.
- Time improvement: Many inputs were classified at the 2nd level instead of the 3rd level (by the input CNN).
- Approach suitable for real-time operations

Conclusion

- A methodology that augments an existing Convolutional Neural Network to improving its classification accuracy for certain classes where it underperforms
- These classes were identified from the confusion matrix of the input CNN
- The proposed structure consists of cascading shallow CNNs, which precede the input CNN, and operate concurrently to minimize the overhead
- Experimental results show significant increase in classification accuracy without increasing inference time

Resources

Vasileios Pentsos, Bijay Raj Paudel, Spyros Tragoudas, Kiriti Nagesh Gowda, and Mike Schmit. "Improved CNN classification accuracy with the addition of shallow cascading CNNs." In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 988-991. IEEE, 2021.