CS 642—Machine Learning **Spring 2003**

Lecture 17: Convolutional Neural Networks

Lecturer: Anshumali Shrivastava *Scribe: Johnson Wu, Patrick Yee, Shuxin Li*

1 Convolution neural networks.

A convolution neural network (CNN) is a deep learning architecture for image processing and vision. CNNs are similar to traditional neural networks and can be written as an alternating series of **convolution** and **pooling** layers, followed by a single **fully connected** layer.

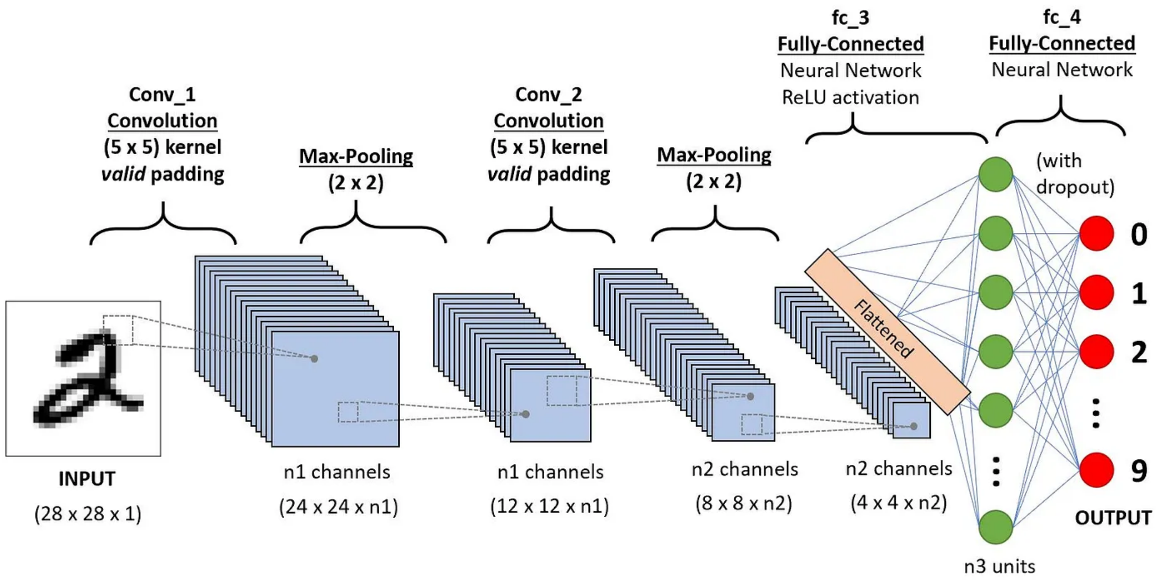


Figure 1: CNN Architecture

1.1 Convolution layers.

In a convolution layer, a sub-patch of fixed size (called the **kernel**) is passed over the feature matrix and used to compute a series of convolved features.

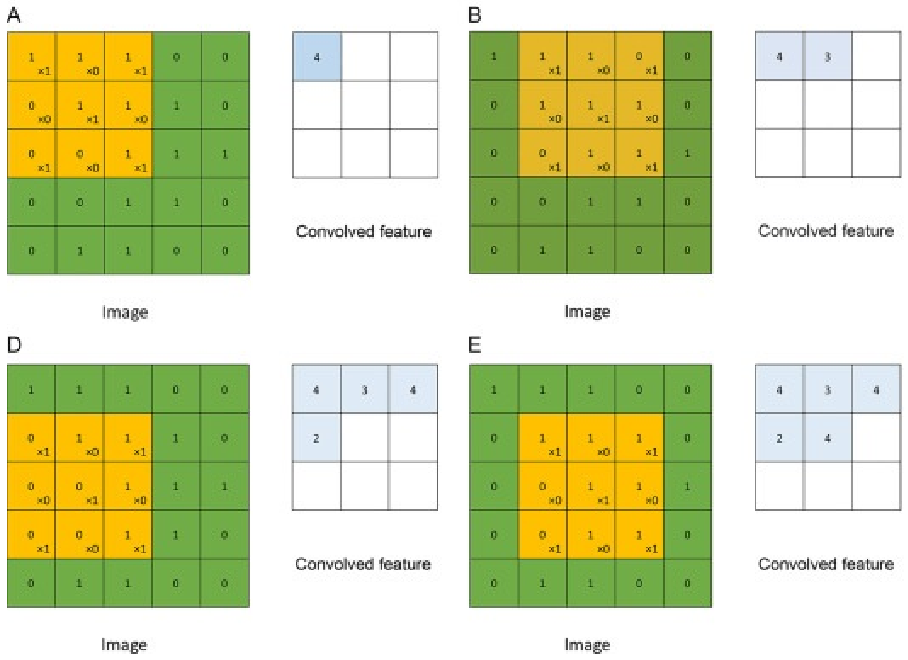


Figure 2: A convolution of a feature matrix (green) with a 3×3 kernel (yellow)

1.1.1 Stride.

The speed (step size) at which the kernal passes over the feature matrix is called the **stride**, and is almost always set to 1.

1.1.2 Padding.

Padding is an artificial boundary of zeroes that surrounds the feature matrix to control the size of the output of each convolution. With no padding, each convolved feature is necessarily smaller than its corresponding feature matrix.

- **Valid padding:** No padding. Convolved features are smaller than inputs.
- **Same padding**. If the kernel is of size $k \times k$, same padding is padding of size $k - 1$. Convolved features are the same size as inputs.
- **Full padding.** Padding larger than k. Convolved features are bigger than inputs and have extra zeroes on the boundary.

1.1.3 Activation function.

Similar to traditional neural networks, convolved features are passed through an activation function to achieve nonlinearity (Usually ReLU).

1.2 Pooling layers.

In a CNN, convolution layers are followed by pooling layers where the dimensionality of the convolved features are reduced by an aggregate function over patches of the convolved matrix. This greatly reduces storage costs during the training phase. The two most common types of pooling are **max-pooling** and **avg-pooling**, which use the maximum and mean aggregate functions, respectively.

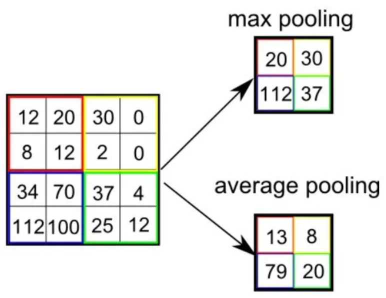


Figure 3: Max/avg-pooling reduces the dimensions of convolved features.

In practice, max-pooling performs much better than avg-pooling and is used in most modern CNNs.

1.3 Fully-connected layer.

The last layer of a CNN is a fully connected layer which is the same as that of a classical neural network. The input vector is a flattened (raveled) representation of the pooled features after the last pooling layer, and while the convolution layers usually use ReLU as an activation function, nonlinearity can be introduced to the fully connected layer through any number of functions depending on the task.

2 AlexNet, EfficientNetV2

2.1 AlexNet

AlexNet is a deep convolutional neural network architecture that was introduced in 2012 by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. It was the first deep neural network to achieve a significant improvement in performance on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), a benchmark for object detection and image classification at large scale.

Before AlexNet, most computer vision tasks relied on shallow neural networks with only a few layers. AlexNet changed this by introducing a much deeper architecture with multiple convolutional layers. It consists of eight layers, including five convolutional layers and three fully connected layers.

The AlexNet was trained on two separate GPUs, the following figure depicts the respective responsibilities between the two GPUs. One GPU runs the layer parts at the top of the figure while the other runs the layer parts at the bottom. The two GPUs only communicate at certain layers.

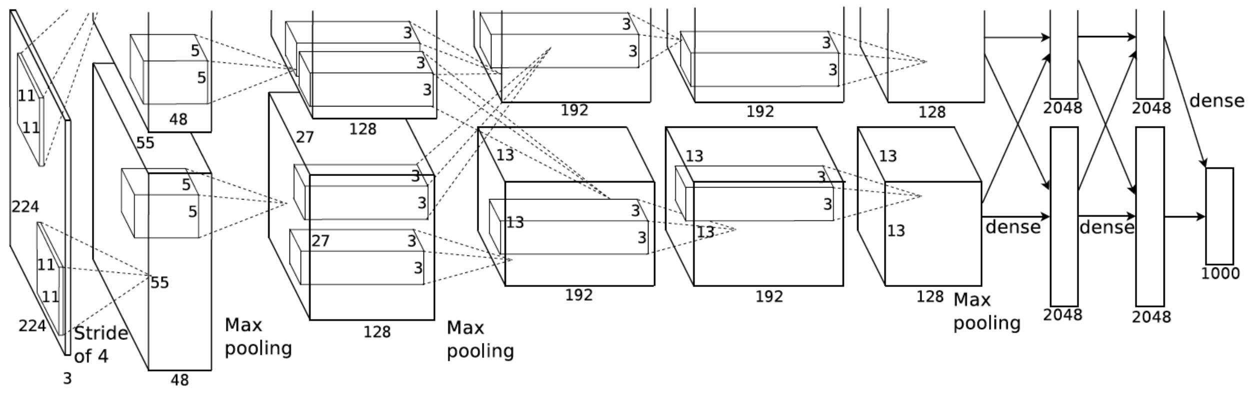


Figure 4: AlexNet Architecture

2.1.1 Relu Activation

One of the key innovations of AlexNet was the use of ReLU (Rectified Linear Unit) activation function instead of the traditional sigmoid function. This allowed for faster training of the network and helped to avoid the problem of vanishing gradients. AlexNet also used data augmentation techniques, such as cropping and horizontal flipping, to increase the size of the training dataset and reduce overfitting.

2.1.2 Result

AlexNet achieved state-of-the-art performance on the ImageNet dataset, reducing the top-5 error rate from 26.2% to 15.3%. This breakthrough in performance led to a resurgence of interest in deep learning and paved the way for many subsequent advances in the field.

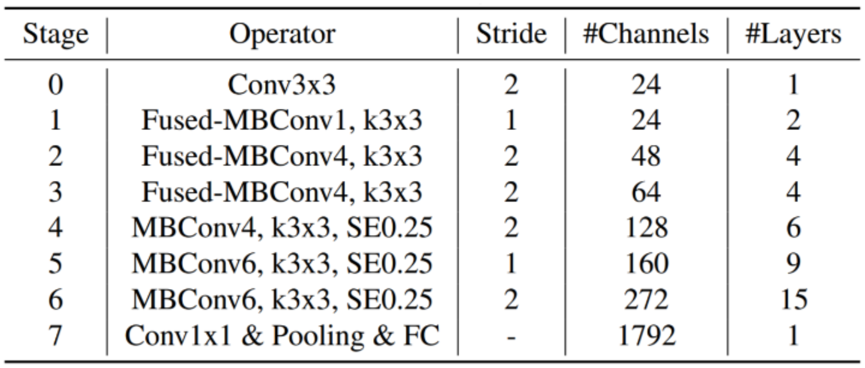
Today, AlexNet is considered a seminal architecture in the development of deep neural networks for computer vision and has been influential in the design of many subsequent models.

2.2 EfficientNetV2

EfficientNetV2 is a family of deep neural network architectures designed to be more efficient in terms of computation and memory usage while maintaining high accuracy on various computer vision tasks, such as image classification and object detection. It is built upon the EfficientNetV2 architecture, which was introduced in 2019 and achieved state-of-the-art performance on multiple image classification benchmarks.

EfficientNetV2 achieves its efficiency by using a compound scaling method that simultaneously scales up the network depth, width, and resolution. Specifically, a new Fused-MBConv module combining multiple operations into a single operation.

EfficientNetV2 is also available in multiple sizes, ranging from the small EfficientNetV2-S to the large EfficientNetV2-L. The different sizes are designed to balance efficiency and performance and can be used in a wide range of applications with different computational resources and requirements.



Stage	Operator	Stride	#Channels	#Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	272	15
7	Conv1x1 & Pooling & FC	-	1792	1

Figure 5: EfficientNetV2 Architecture

2.2.1 Fused-MBConv

The Fused-MBConv module is an extension of the Mobile Inverted Residual Bottleneck (MBConv) module, which is a building block in efficientNetV1 neural networks. The MBConv module consists of three main operations: depthwise convolution, pointwise convolution, and a bottleneck layer. The depthwise convolution reduces the number of input channels, the pointwise convolution increases the number of output channels, and the bottleneck layer reduces the computational cost of the module.

The Fused-MBConv module combines these three operations into a single operation, which reduces the memory bandwidth and computational cost of the module. Specifically, the Fused-MBConv module performs a combined depthwise convolution and pointwise convolution, followed by a fused bottleneck layer. This reduces the number of operations and memory accesses needed to perform the same computation, leading to faster and more efficient models. The Fused-MBConv module has been shown to improve the performance of neural networks on a variety of tasks, including image classification and object detection. It is particularly effective when used in conjunction with other efficiency techniques, such as channel pruning and quantization.

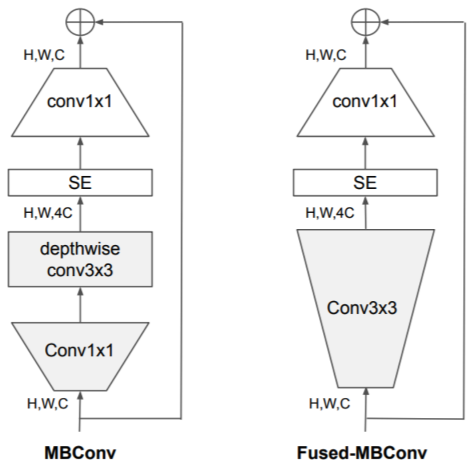


Figure 6: MBConv, Fused-MBConv

2.2.2 Result

EfficientV2 has achieved state-of-the-art performance on multiple computer vision tasks, including image classification on ImageNet, object detection on COCO, and semantic segmentation on ADE20K. It has also been used as a backbone network for various downstream tasks, such as face recognition and image generation.

3 Vision Transformer

Over the years, convolutional neural networks have been the state-of-the-art methods in computer vision. However, with the increase in image resolution, CNN architecture has been found to be computationally overwhelming. For example, training with a dataset of 4K images, each containing 16 million pixels, can require hundreds of billions of floating-point operations, which can be very demanding on hardware resources.

Motivated by the scaling success of the Transformer architecture in natural language processing, Dosovitskiy et al. (2021) applied the standard Transformer directly to images with the fewest modifications possible. This approach resulted in the creation of a new architecture known as the Vision Transformer (Vit).

3.1 Architecture

The architecture is composed of transformer blocks, each of which contains a multi-head self-attention layer and a feed-forward layer. The Vision Transformer (ViT) utilizes self-attention mechanisms to extract features from images. The self-attention layer calculates the relative importance of each pixel in the image by considering its relationship with all other pixels.

In order to feed the image data into the Transformer architecture, the team structured the input image data in a way that resembles the input of text data. The overall method consists of the following operations

1. Split the image into identical size patches
    - patch size (e.g., 16x16)
2. Flatten the 2D patches
3. Map the flattened patches to D dimensions, with a trainable linear projection.
4. Prepend a trainable embedding to the sequence of embedded patches, which stands for [class].
5. Augmented the sequence with positional embeddings, hence introducing positional information
6. Feed the sequence as an input to a Transformer encoder

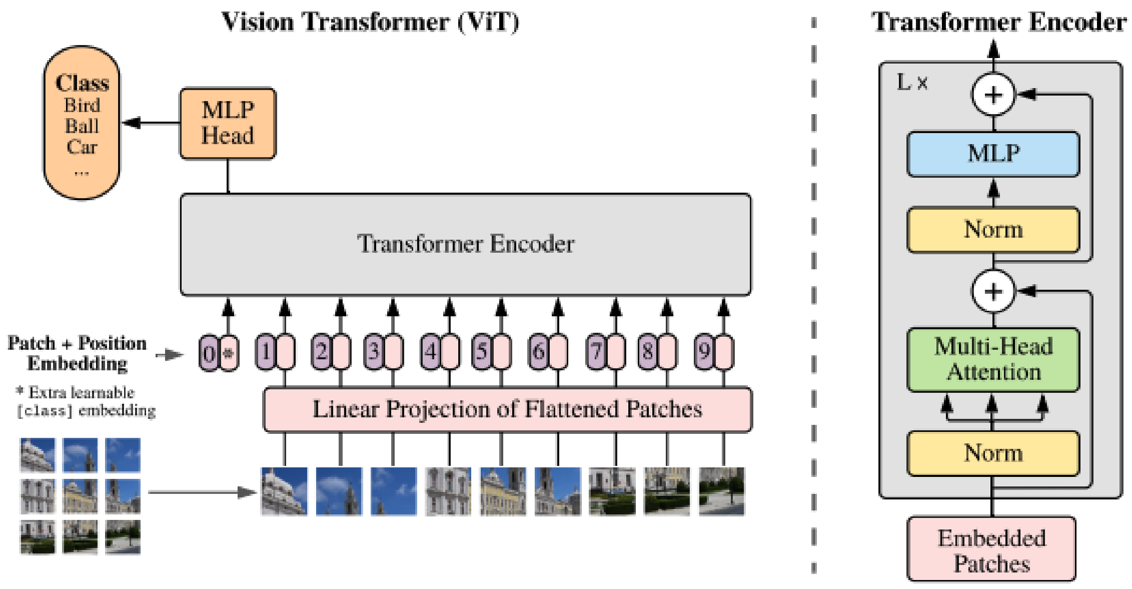


Figure 7: ViT architecture

3.2 Experiment

In order to evaluate the representation learning capabilities and computational cost of other CNN networks and Vision Transformer (ViT). The model was pre-trained on datasets of varying sizes and evaluated several benchmark tasks in computer vision.

3.2.1 Dataset

The model was pretrained on three datasets, public ILSVRC-2012 ImageNet dataset with 1k classes and 1.3M images, its superset public ImageNet-21k with 21k classes and 14M images (Deng et al., 2009), and in-house JFT (Sun et al., 2017) with 18k classes and 303M high-resolution images.

3.2.2 Result

When pre-trained on ImageNet with 1.3M images, ViT is slightly worse than ResNet. The outcome is expected because the Transformer lack of certain inductive biases that are inherent to CNNs, such as translation equivariance and locality. However, when pre-trained on larger datasets (ImageNet-21K, JFT), the performance was able to tie or outperform the state of the art CNN model on multiple image recognition benchmarks.

3.2.3 Scaling

Since the original goal of ViT was to address the scaling limitations of traditional CNNs, an experiment was conducted to evaluate the performance versus pre-training costs of various models (including 7 ResNets and 6 Vision Transformers). The results showed that Vision Transformers dominate ResNets on the performance/compute tradeoffs. ViT uses only 2 4× less computation to achieve the same performance.

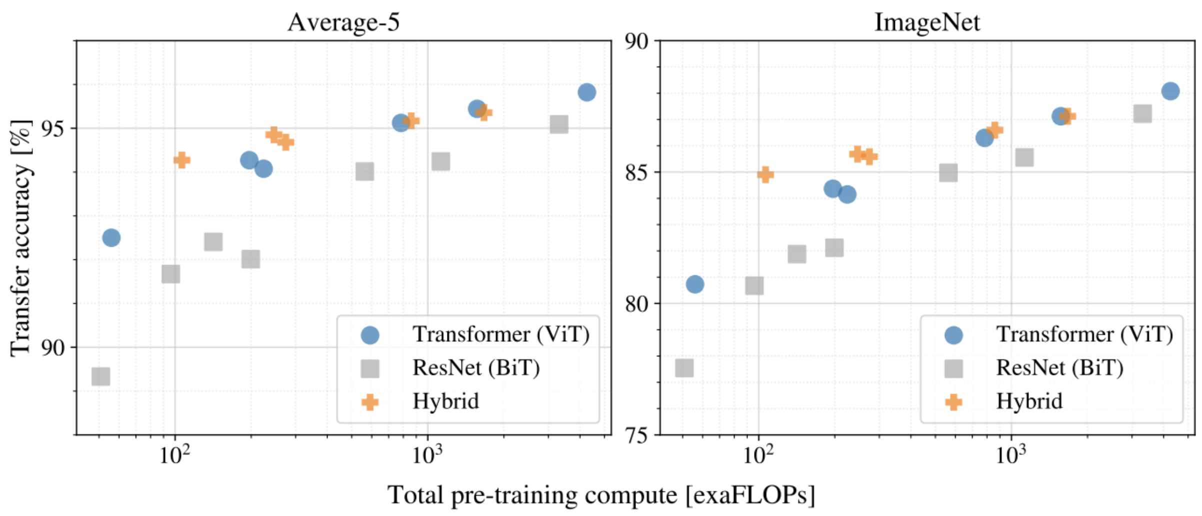


Figure 8: Computation/accuracy of each model

References

[1] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, *NIPS 2012*

[2] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, Rethinking the Inception Architecture for Computer Vision, *ArXiv*

[3] Mingxing Tan, Quoc V. Le, EfficientNetV2: Smaller Models and Faster Training, *ArXiv*

[4] Dosovitskiy, Alexey, Beyer, Lucas, Kolesnikov, Alexander, Weissenborn, Dirk, Zhai, Xiaohua, Unterthiner, Thomas, Dehghani, Mostafa et al., An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, *ArXiv*