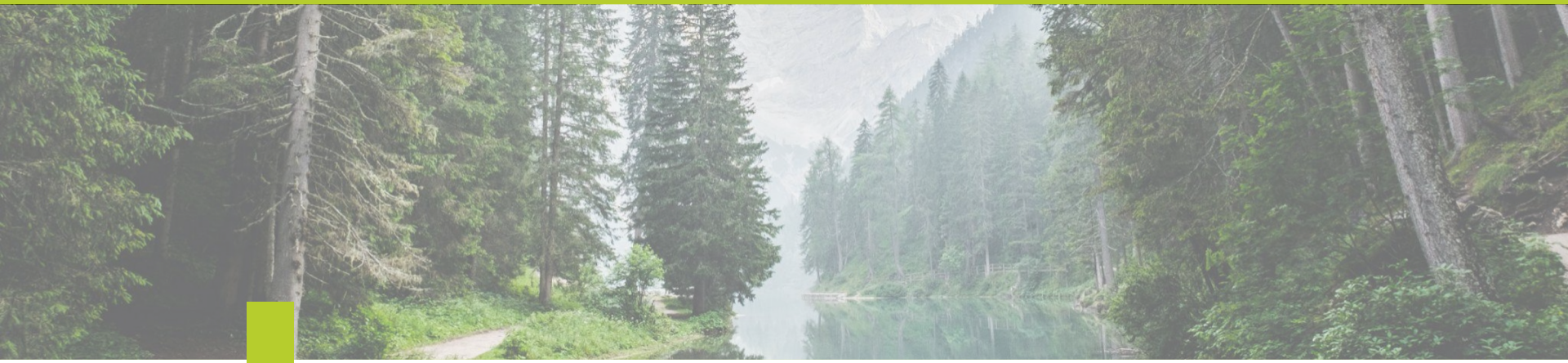


Leveraging AI in Waste Management

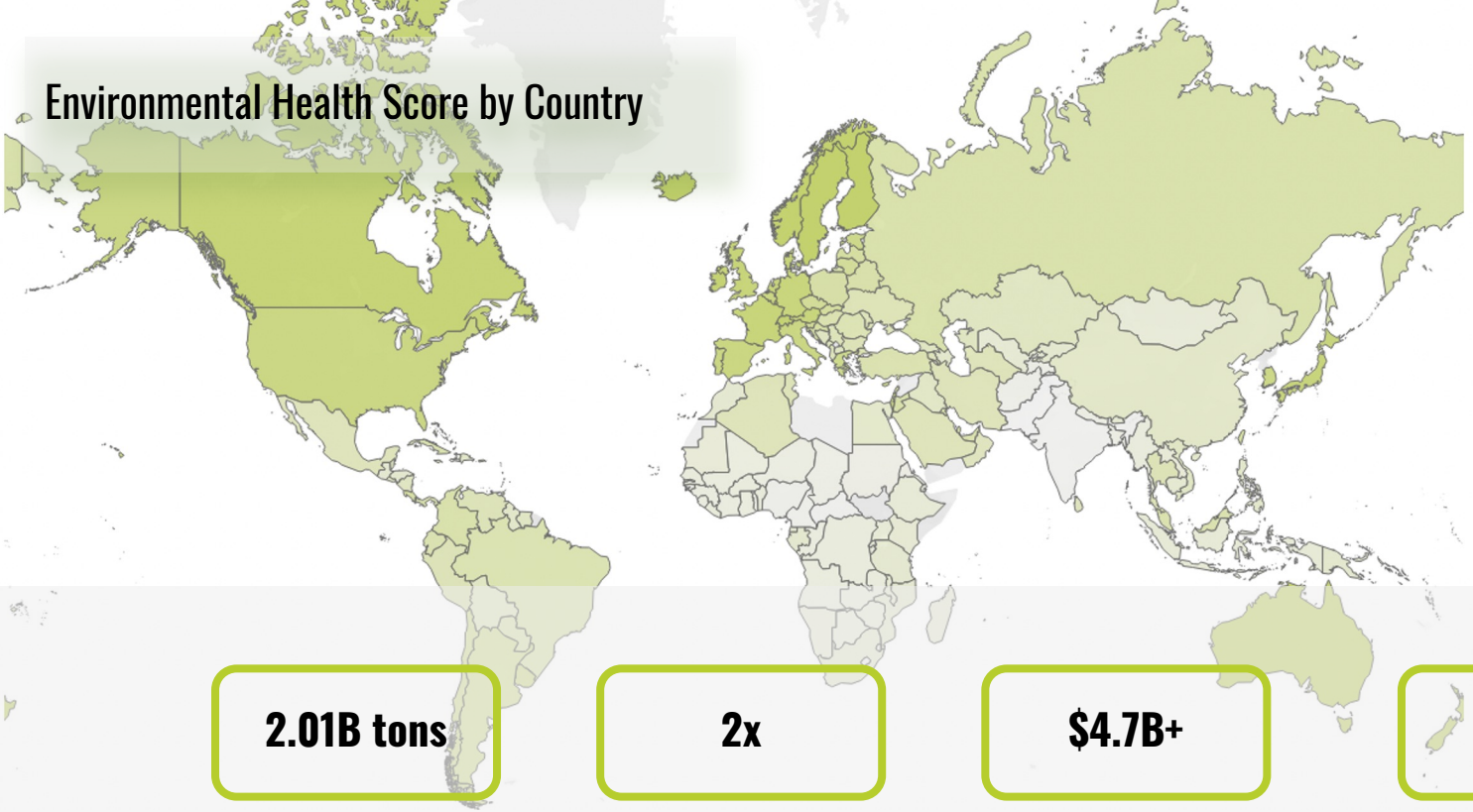
Deploying Various Deep Learning Architectures in Garbage Classification

Team 1: Qing Wen, Linfeng Yan, Lin Ye, Henry Yu, Lin Zhang

Agenda

1. Background & Objective
2. Data Overview
3. Methodology
4. Experiment Results
5. Conclusion

Environmental Health Score by Country



2.01B tons

Solid waste generated annually

2x

Waste generation is expected to be doubled by 2050 in North America

$4.7B+

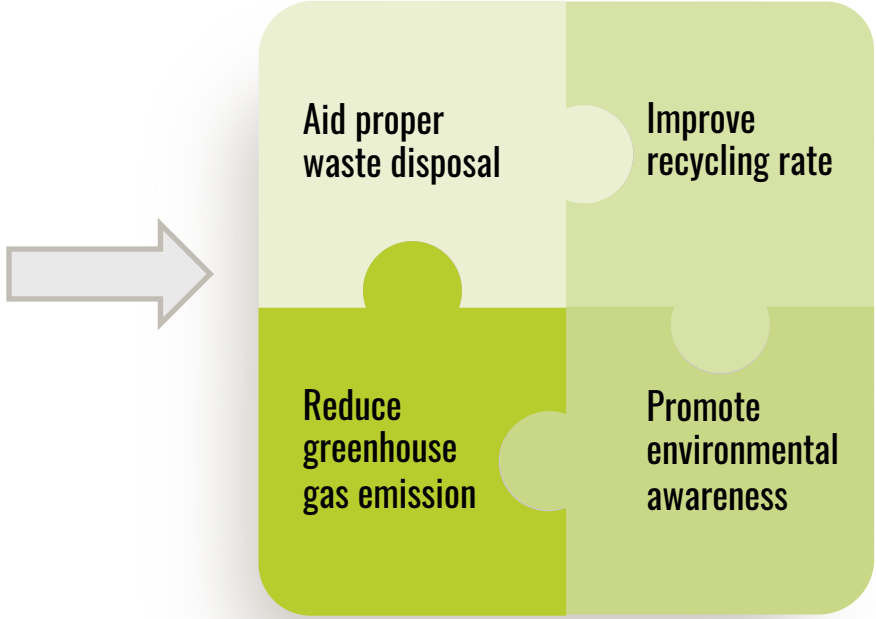
For waste management program worldwide

$500M

Budget dedicated to waste management services in Canada

Project Goal

To leverage modern AI techniques to accurately and efficiently classify different types of waste materials



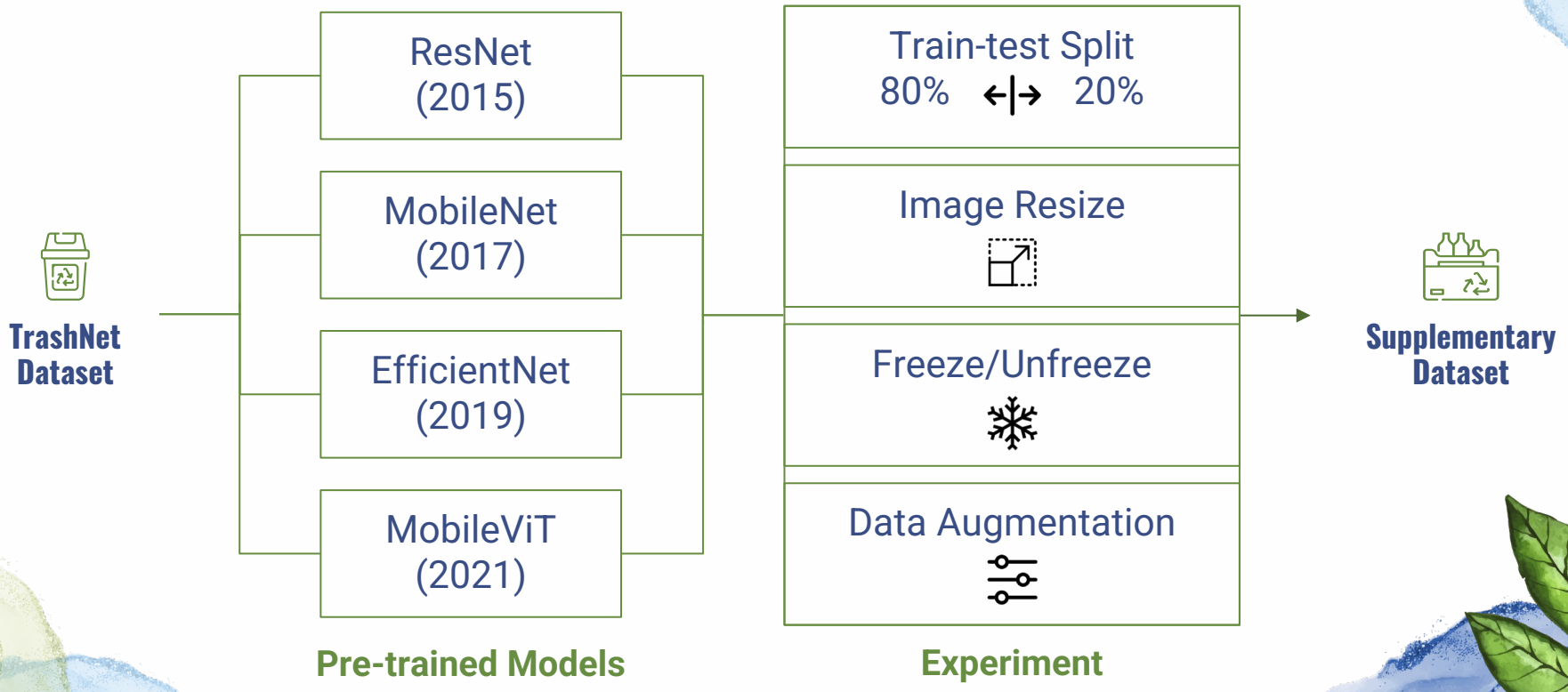
Agenda

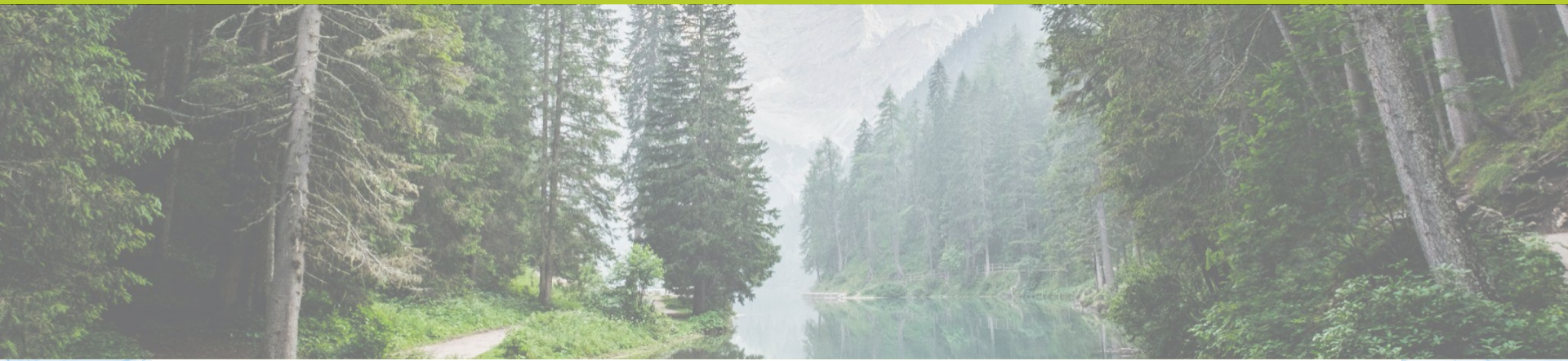
Data Overview

	TrashNet	Garbage Classification
# Observations	2,467	1,890
# Categories	6	6
Categories	Cardboard, Glass, Metal, Paper, Plastic, Trash	Cardboard, Glass, Metal, Paper, Plastic, Trash
Description	Primary training dataset	Experimental dataset
Sample Image		

Agenda

Experiment Setup



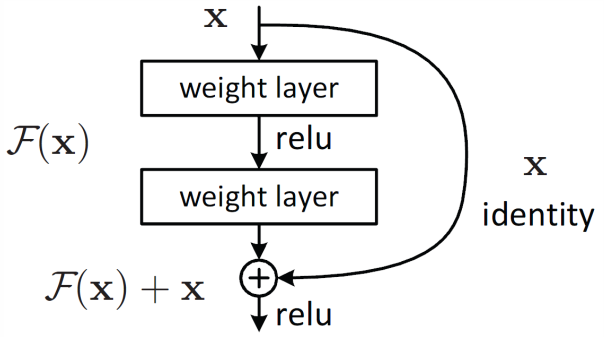


Model 1: ResNet

Team 1: Qing Wen, Linfeng Yan, Lin Ye, Henry Yu, Lin Zhang

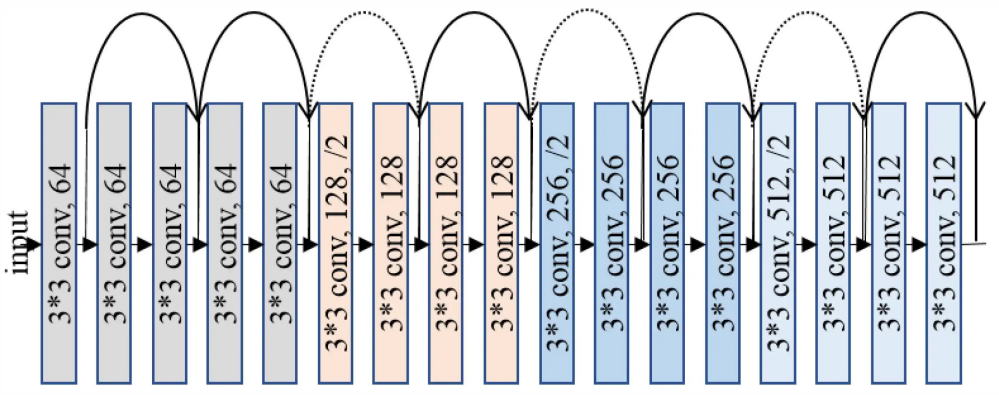
ResNet

- Designed to support hundreds or thousands of convolutional layers
- Networks with a large number of layers can be trained easily without increasing the training error percentage
- **Residual blocks** allow the network to learn residual mappings instead of direct mappings



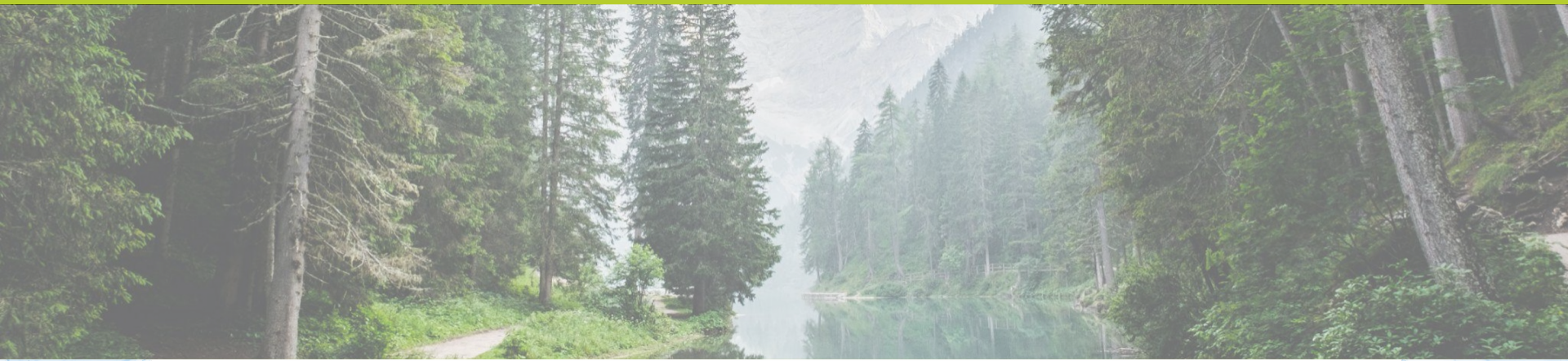
A Residual Block

ResNet



ResNet18 Model

- Above is a ResNet18 model which consists of 18 layers
- The blocks between the curved arrow represent a Residual Block
- The dotted arrows represent the shortcuts to match dimensions
- We chose **ResNet101**, which constructs a 101-layer ResNet by using more 3-layer blocks
- The additional layers can capture more complex features and patterns in images



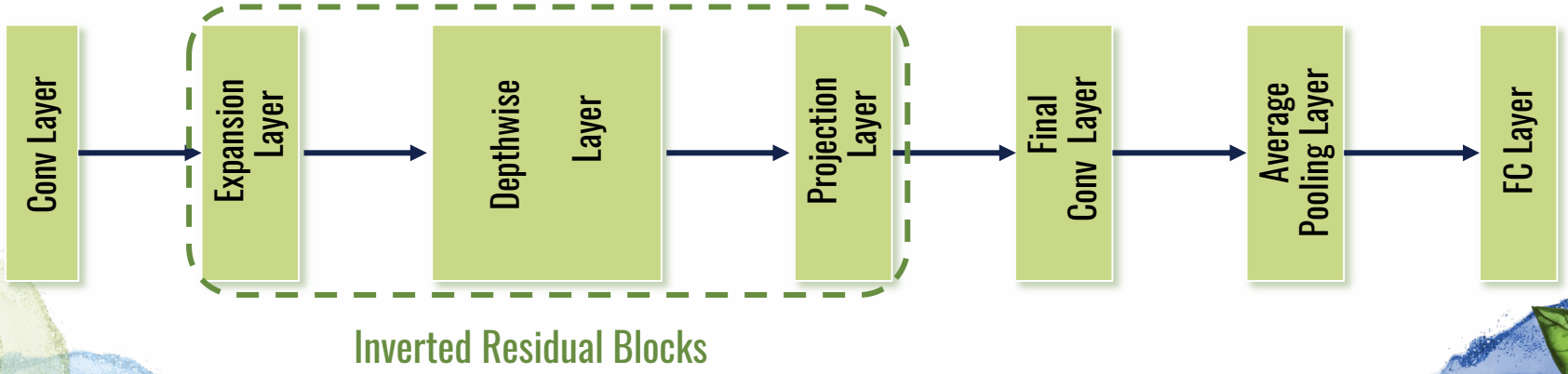
Model 2: MobileNet

Team 1: Qing Wen, Linfeng Yan, Lin Ye, Henry Yu, Lin Zhang

MobileNet

- A lightweight and efficient deep neural network
- Utilized MobileNetV2
- Robust feature extraction
- Efficient

Overall Structure



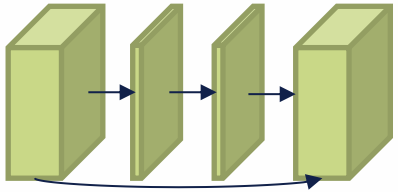
MobileNet-V2

Inverted Residual Blocks

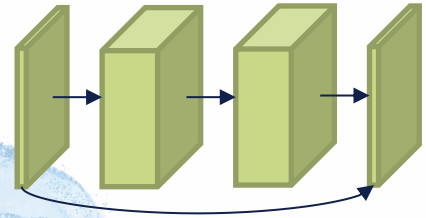
Components

- Expansion layer (1x1)
- Depthwise layer (3x3)
- Projection layer (1x1)

Traditional Residual Block



Inverted Residual Block



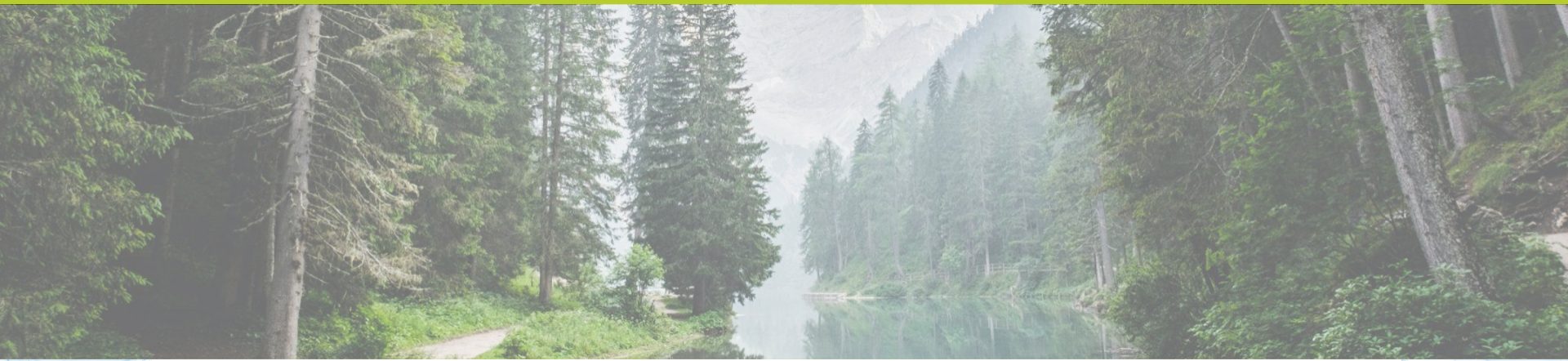
Linear Bottleneck

Purposes

- Reduce the computational complexity and model size
- Maintain the network's ability to learn and extract features

Where to apply

- Projection layer
- Final convolutional layer



Model 3: EfficientNet

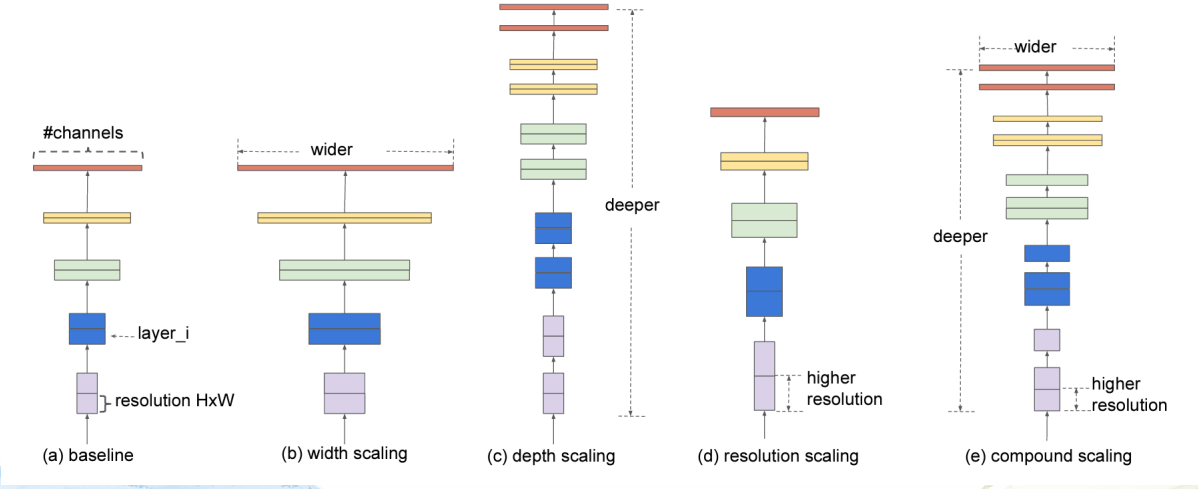
Team 1: Qing Wen, Linfeng Yan, Lin Ye, Henry Yu, Lin Zhang

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

- Propose a new scaling method that uniformly scales all dimensions of depth/width/resolution for CNNs
- Surpass state-of-the-art accuracy with an order of magnitude fewer parameters and FLOPS (floating point operations)
- EfficientNet-B7 achieves state-of-the-art 84.3% top-1 accuracy on ImageNet, while being 8.4x smaller and 6.1x faster on inference than the best existing ConvNet

Different Scaling Methods for CNN

- **Depths (d):** The intuition is that deeper ConvNet can capture richer and more complex features, and generalize well on new tasks.
- **Width (w):** Wider networks tend to be able to capture more fine-grained features and are easier to train
- **Resolution (r)**: With higher resolution input images, ConvNets can potentially capture more fine-grained patterns.



Different Scaling Methods for CNN

- **Observation 1:** Scaling up any dimension of network width, depth, or resolution improves accuracy, but the accuracy gain diminishes for bigger models
- **Observation 2:** In order to pursue better accuracy and efficiency, it is critical to balance all dimensions of net work width, depth, and resolution during ConvNet scaling.

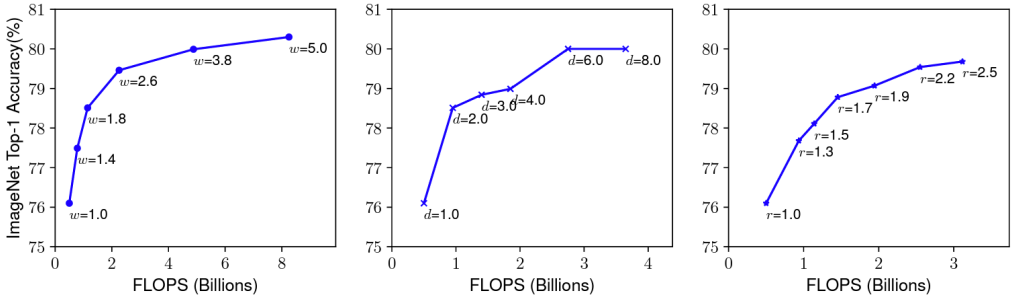


Figure 3. **Scaling Up a Baseline Model with Different Network Width (w), Depth (d), and Resolution (r) Coefficients.** Bigger networks with larger width, depth, or resolution tend to achieve higher accuracy, but the accuracy gain quickly saturate after reaching 80%, demonstrating the limitation of single dimension scaling. Baseline network is described in Table 1.

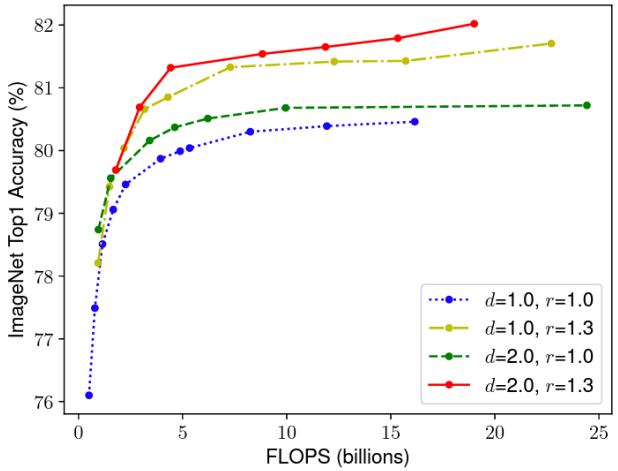


Figure 4. **Scaling Network Width for Different Baseline Networks**. Each dot in a line denotes a model with different width coefficient (w). All baseline networks are from Table 1. The first baseline network (d=1.0, r=1.0) has 18 convolutional layers with resolution 224x224, while the last baseline (d=2.0, r=1.3) has 36 layers with resolution 299x299.

Compound Scaling Method

- Notably, the FLOPS of a regular convolution op is proportional to d, w^2, r^2
- Control the overall FLOPS < 2^{φ}

$$\text{depth: } d = \alpha^{\phi}$$
$$\text{width: } w = \beta^{\phi}$$
$$\text{resolution: } r = \gamma^{\phi}$$
$$\text{s.t. } \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$$
$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

Baseline: EfficientNet B0

- First fix φ=1
- Do a small grid search of α, β, γ
- Best values are α=1.2, β= 1.1, γ=1.15, under constraint of $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$

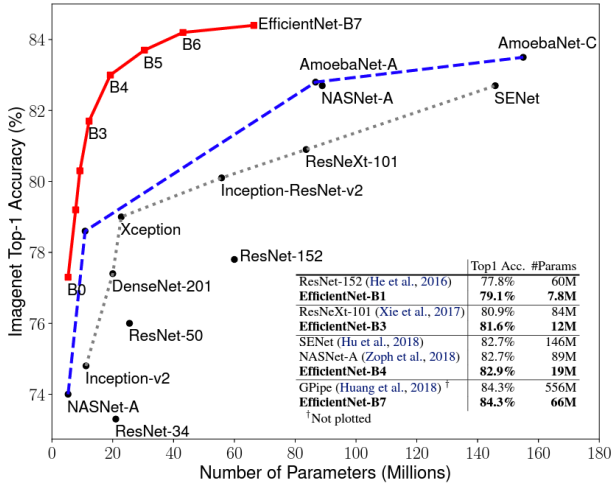
Table 1. **EfficientNet-B0 baseline network** – Each row describes a stage i with $\hat{L}_i$ layers, with input resolution $\langle \hat{H}_i, \hat{W}_i \rangle$ and output channels $\hat{C}_i$. Notations are adopted from equation 2.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

EfficientNet Performance on ImageNet

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPs	Ratio-to-EfficientNet
EfficientNet-B0	**77.1%**	**93.3%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**79.1%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**80.1%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.6%**	**95.7%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.9%**	**96.4%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.6%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.8%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.3%**	**97.0%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

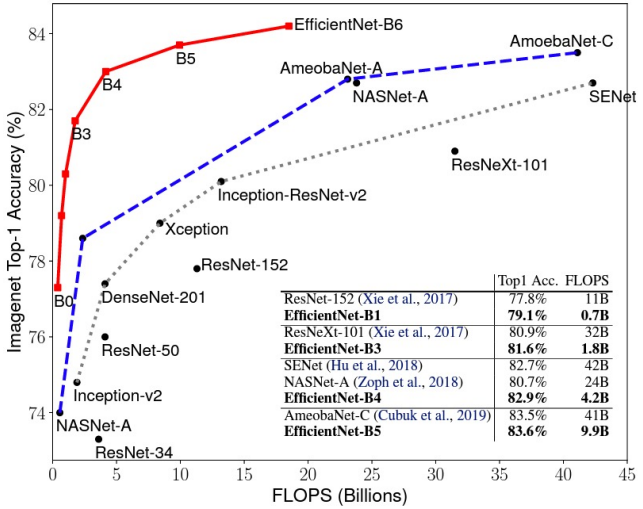
EfficientNet Performance on ImageNet



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

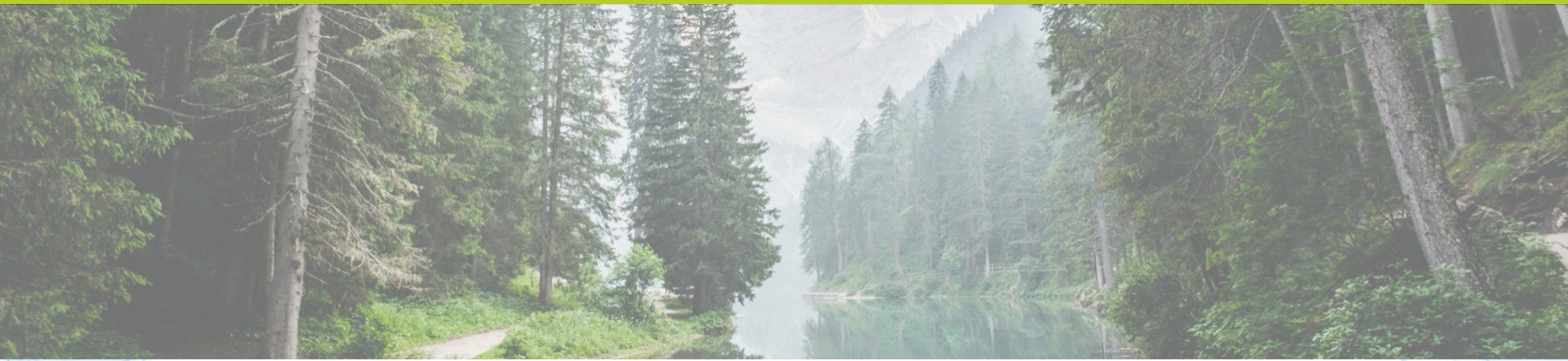
†Not plotted

Figure 1. **Model Size vs. ImageNet Accuracy.** All numbers are for single-crop, single-model. Our EfficientNets significantly outperform other ConvNets. In particular, EfficientNet-B7 achieves new state-of-the-art 84.3% top-1 accuracy but being 8.4x smaller and 6.1x faster than GPipe. EfficientNet-B1 is 7.6x smaller and 5.7x faster than ResNet-152. Details are in Table 2 and 4.



	Top1 Acc.	FLOPS
ResNet-152 (Xie et al., 2017)	77.8%	11B
EfficientNet-B1	**79.1%**	**0.7B**
ResNeXt-101 (Xie et al., 2017)	80.9%	32B
EfficientNet-B3	**81.6%**	**1.8B**
SENet (Hu et al., 2018)	82.7%	42B
NASNet-A (Zoph et al., 2018)	80.7%	24B
EfficientNet-B4	**82.9%**	**4.2B**
AmeobaNet-C (Cubuk et al., 2019)	83.5%	41B
EfficientNet-B5	**83.6%**	**9.9B**

Figure 5. **FLOPS vs. ImageNet Accuracy –** Similar to Figure 1 except it compares FLOPS rather than model size.



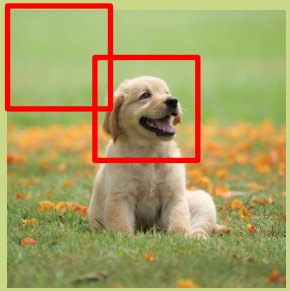
Model 4: Mobile ViT

Team 1: Qing Wen, Linfeng Yan, Lin Ye, Henry Yu, Lin Zhang

Inductive Bias

- Built in assumptions about spatial relationships and local structure within the input data

Locality



- Focus on smaller region
- Good at finding edge, corners

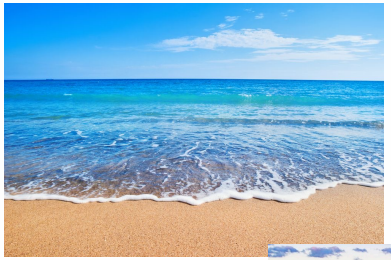
Translation Equivaraince

- Apply same filter across entire input
- Detect the object no matter where it is

Limitation of CNNs

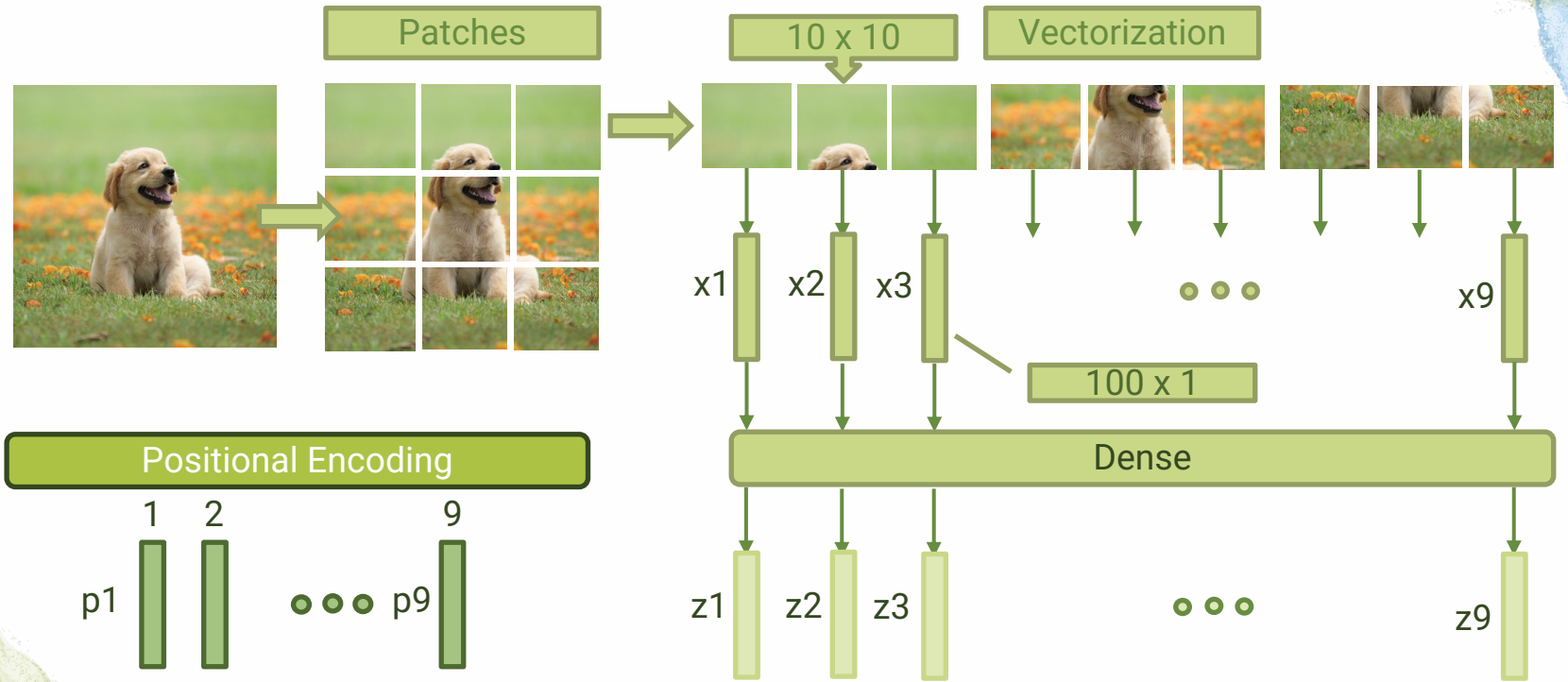
Limited Global Context:

- Scene (beach, forest, city)



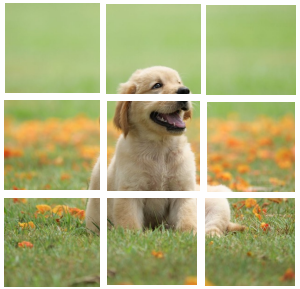
- Object Relationship (person holding a leash connected to a dog)

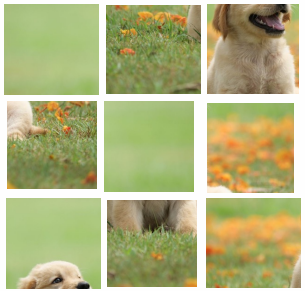
ViT (Vision Transformer)



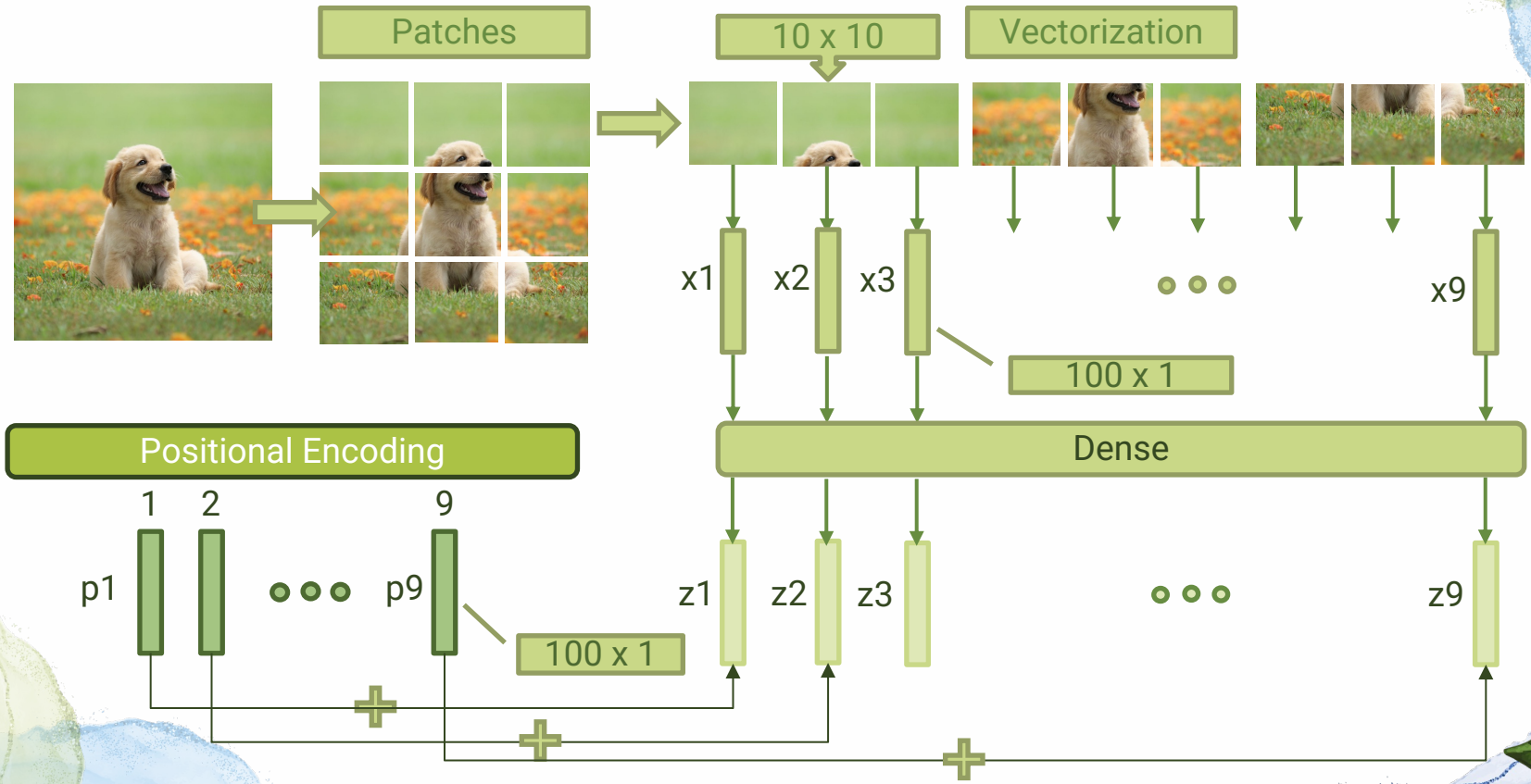
Why positional encoding?

- Transformers rely on self-attention mechanisms to model relationships between input element.
- Self-attention is permutation invariant: treats input elements as an unordered set
- The following two pictures would have the same output if we don't add positional encoding when we use Transformer
- We want to let transformer understand that these two pictures are different since the relative positions of the patches play a crucial role in understanding the context of the image

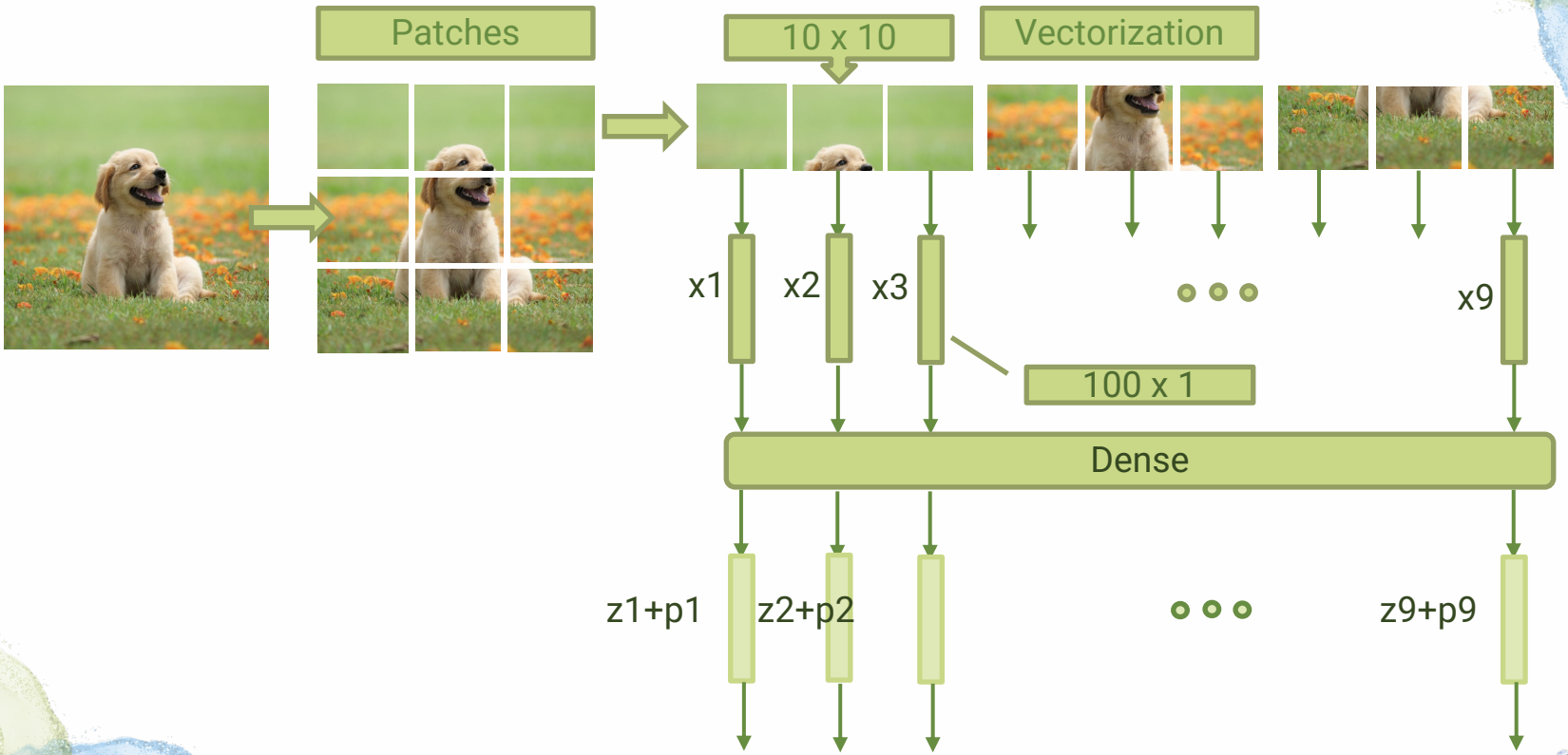




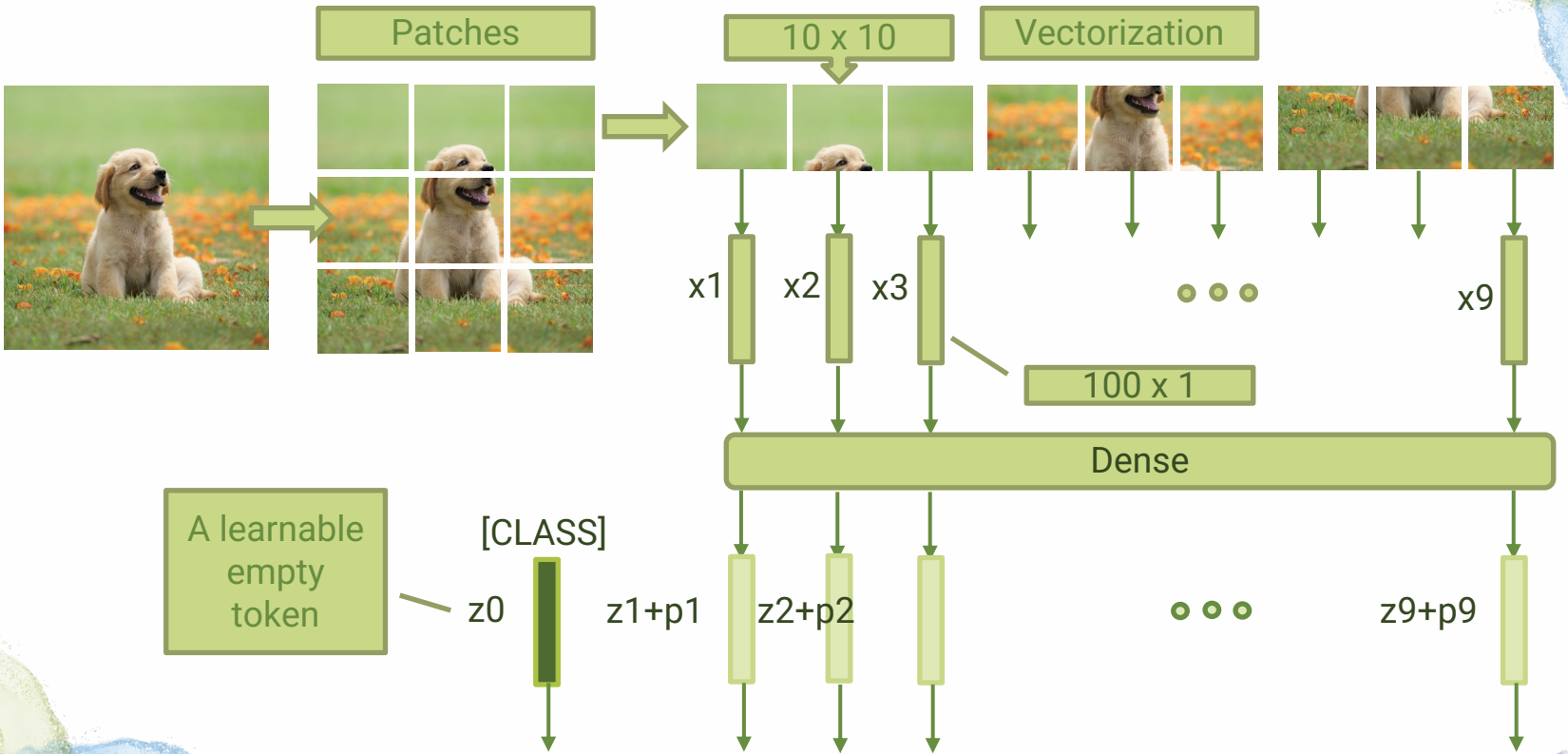
ViT (Vision Transformer)



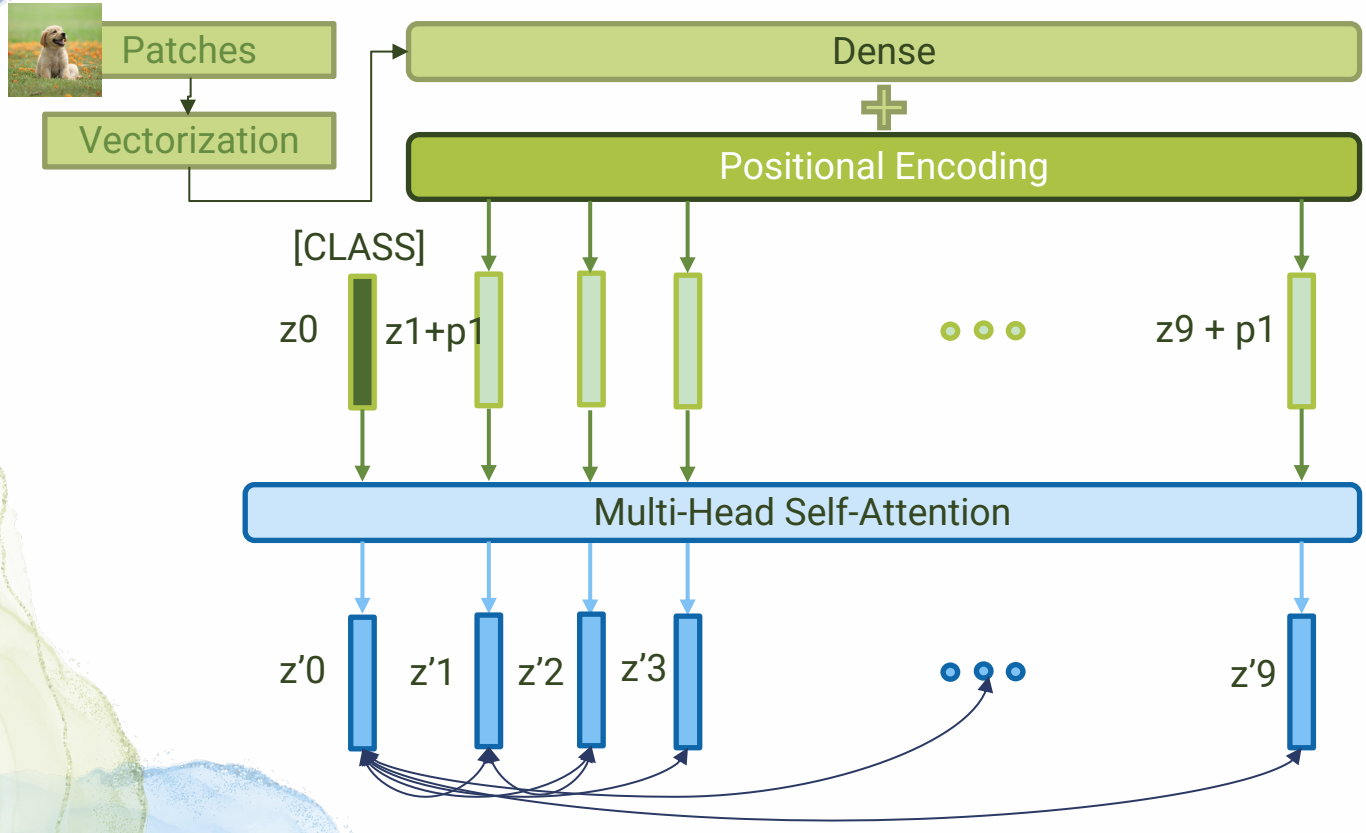
ViT (Vision Transformer)



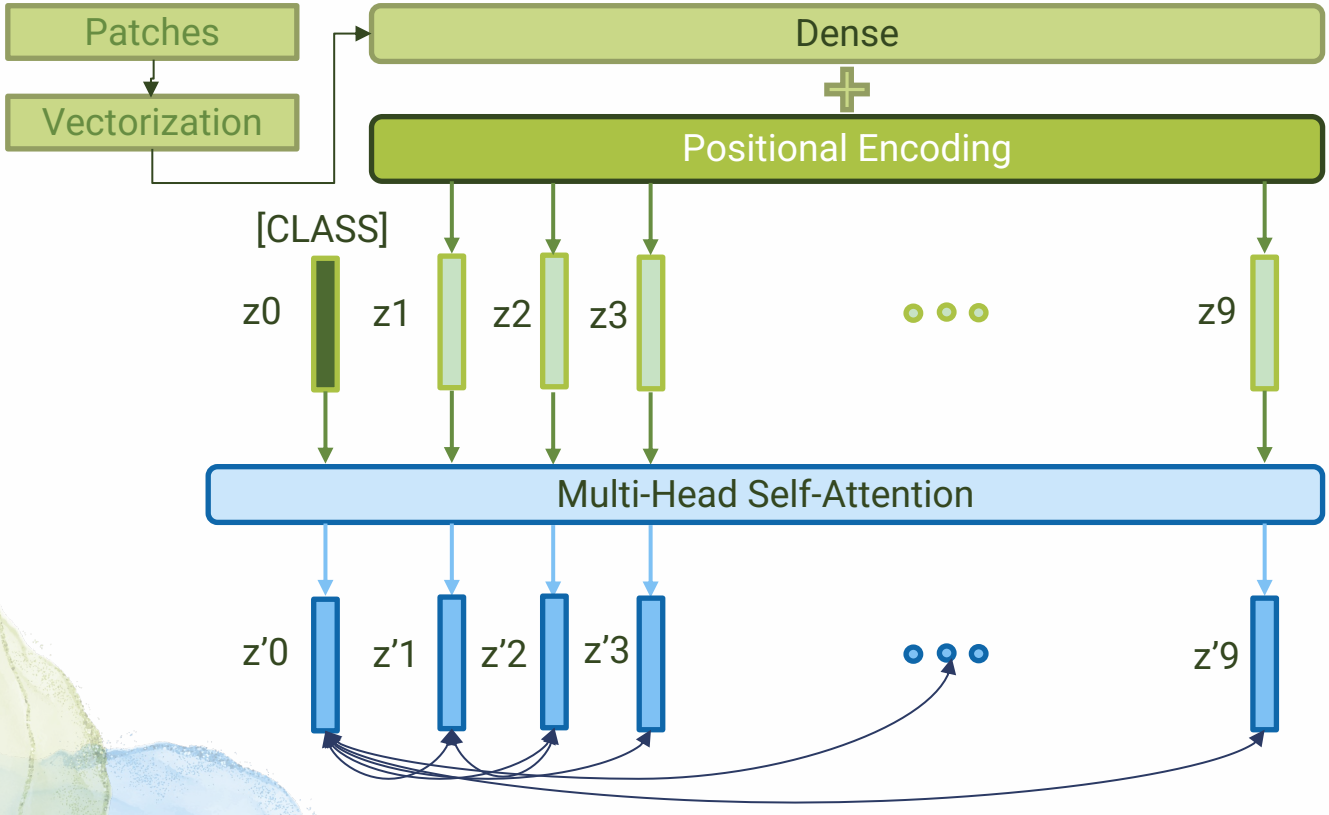
ViT (Vision Transformer)

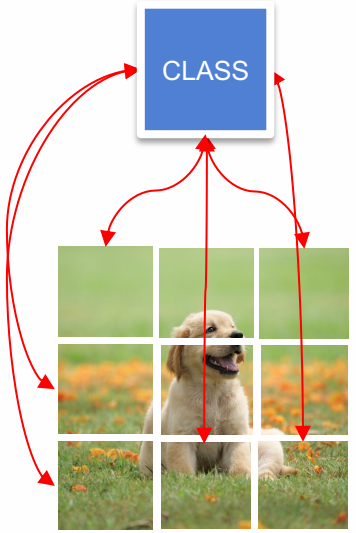


ViT (Vision Transformer)

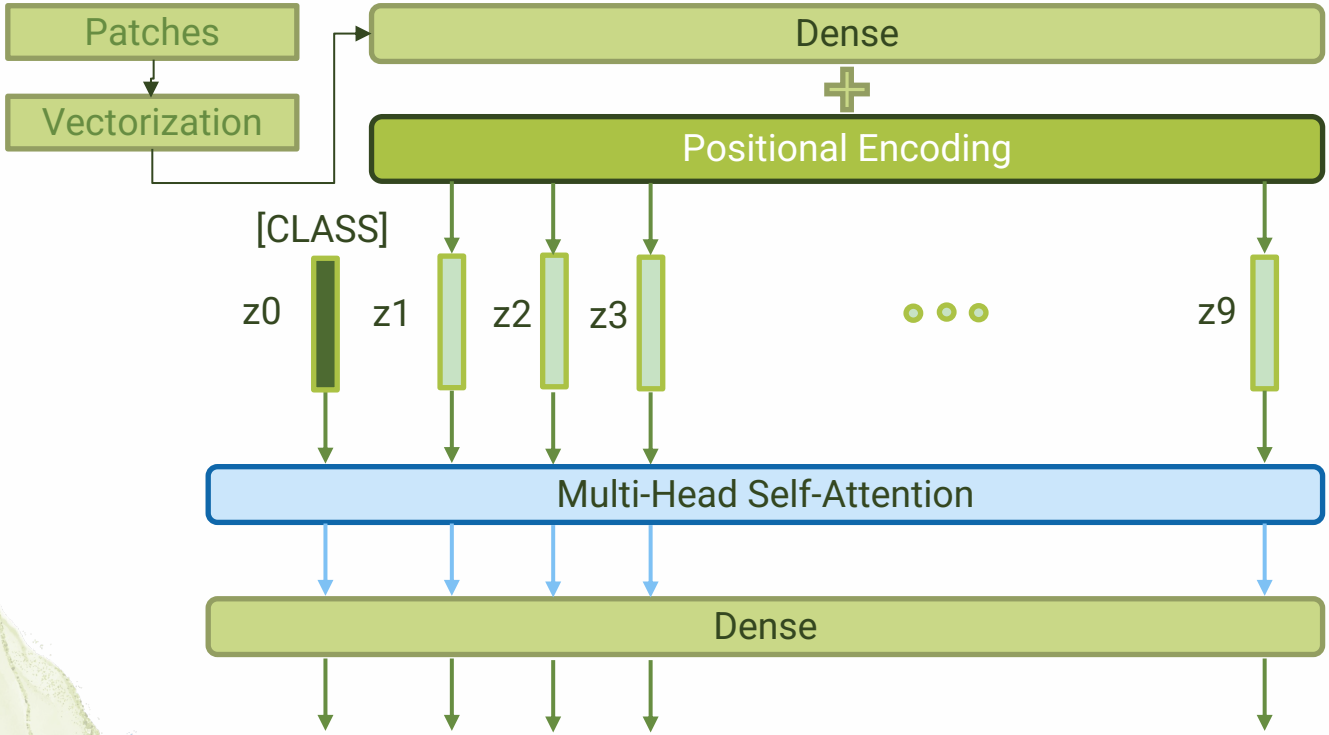


ViT (Vision Transformer)

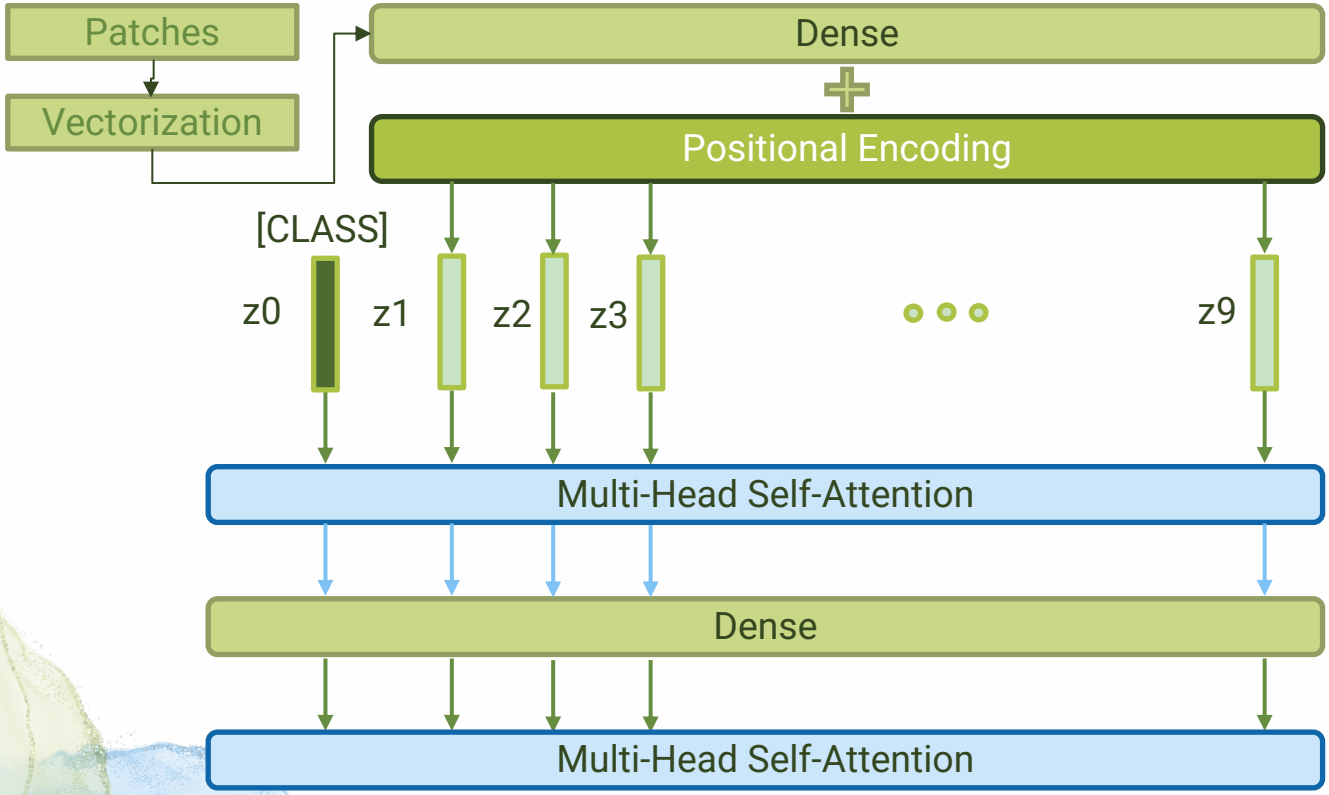




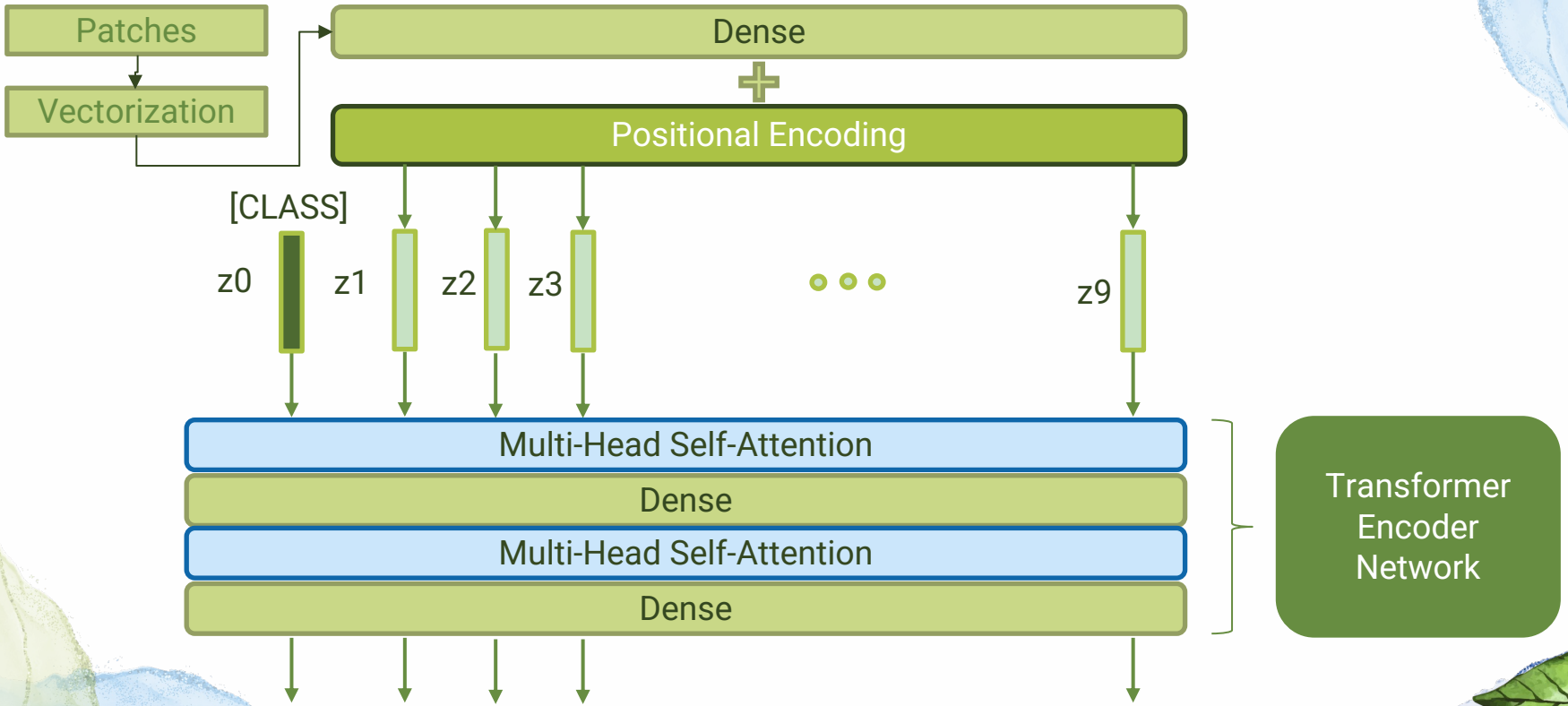
ViT (Vision Transformer)



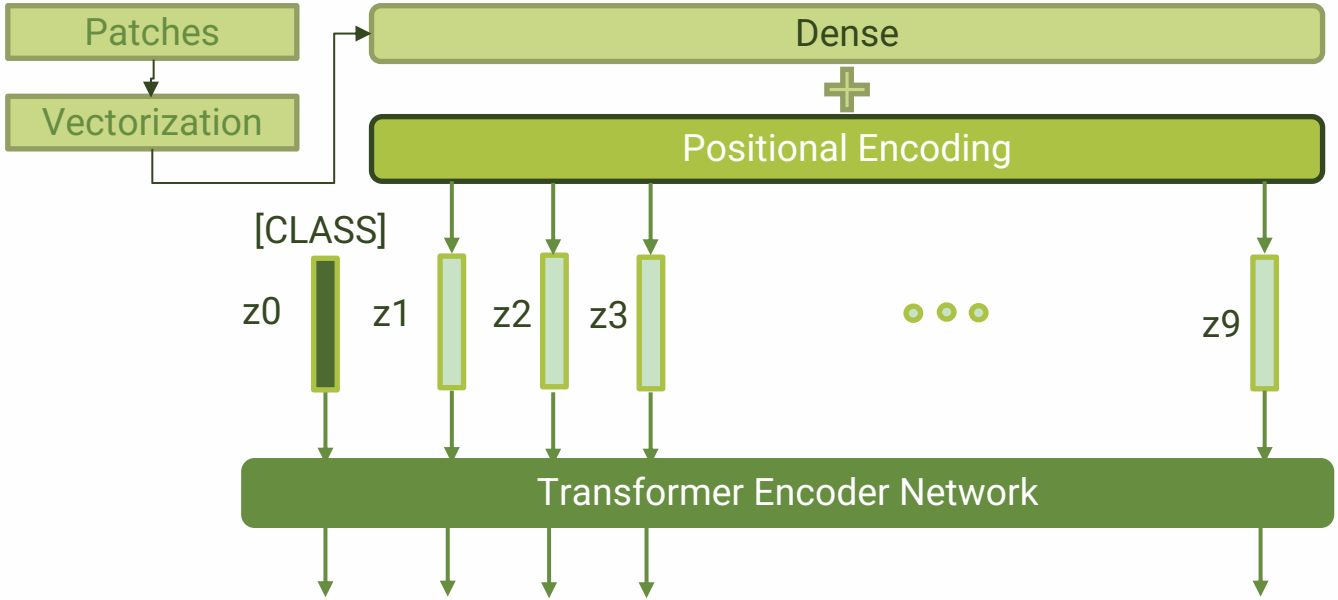
ViT (Vision Transformer)



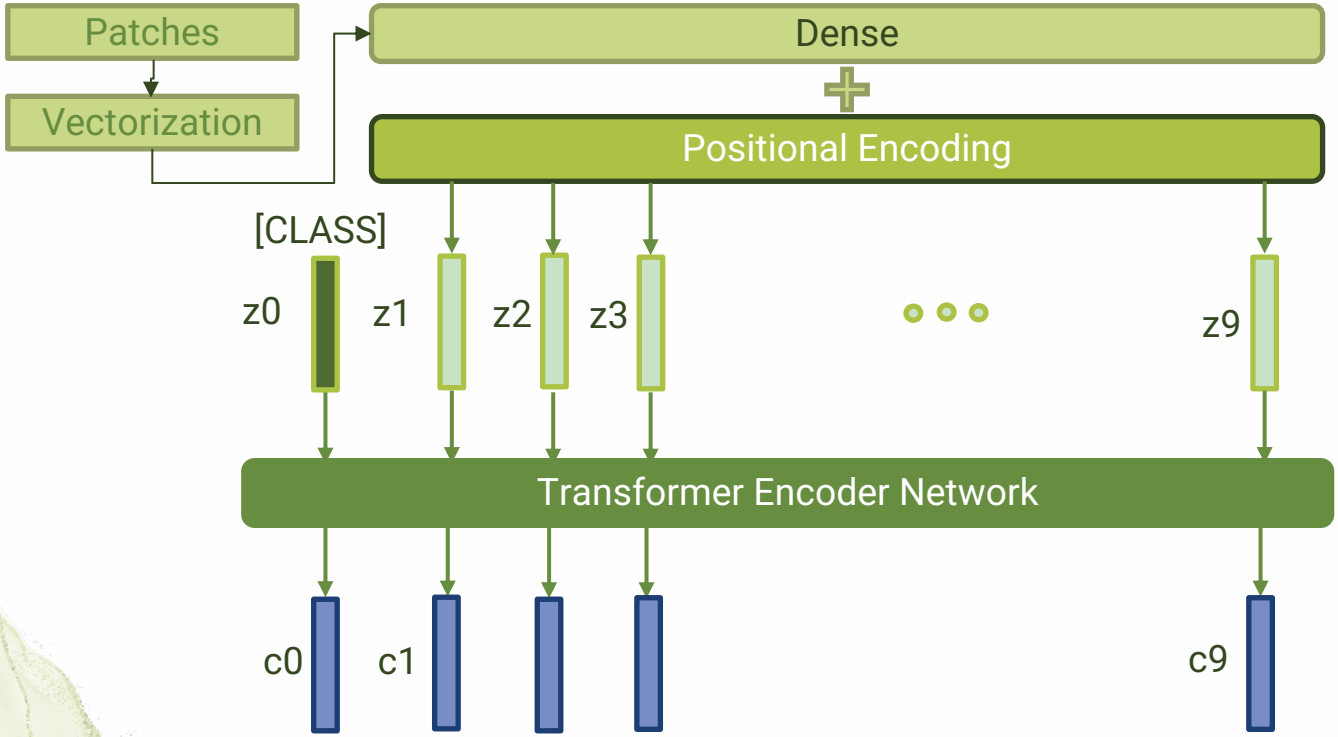
ViT (Vision Transformer)



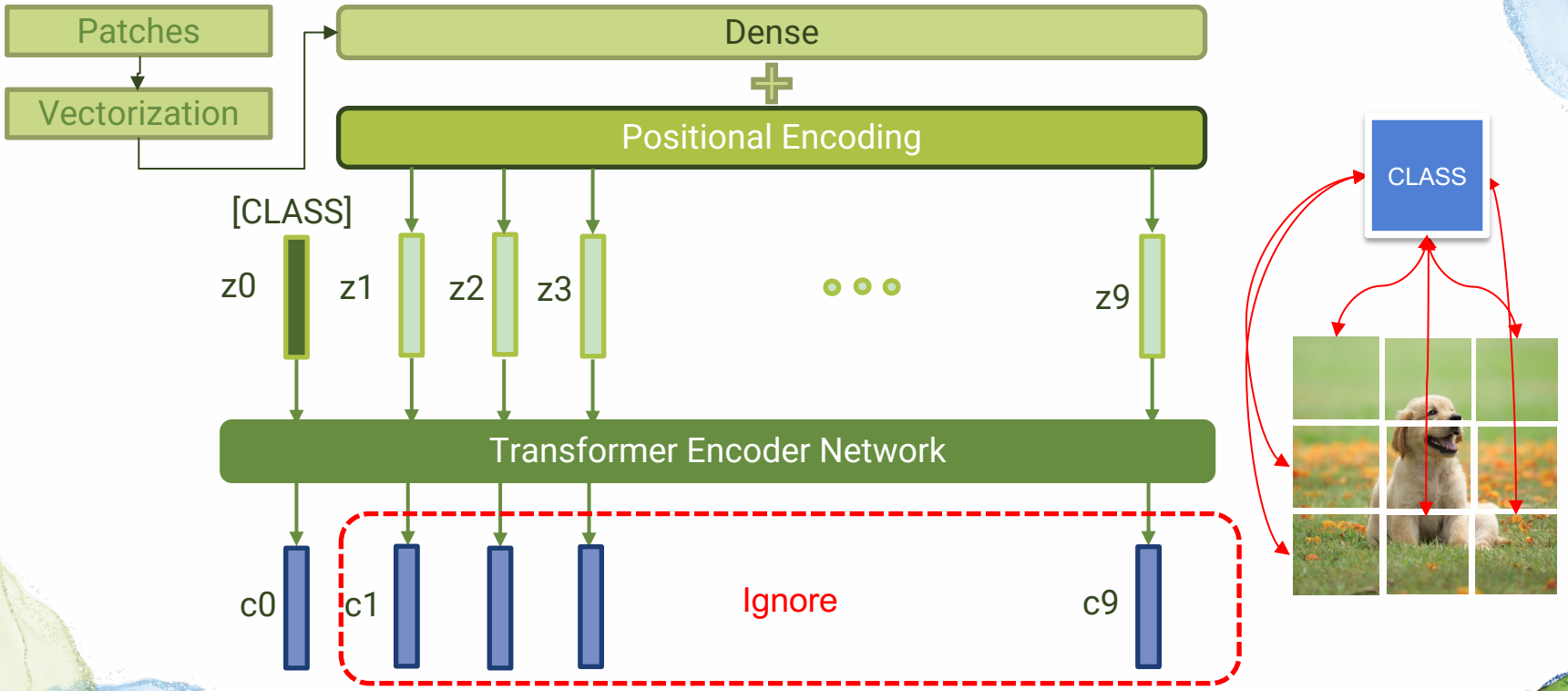
ViT (Vision Transformer)



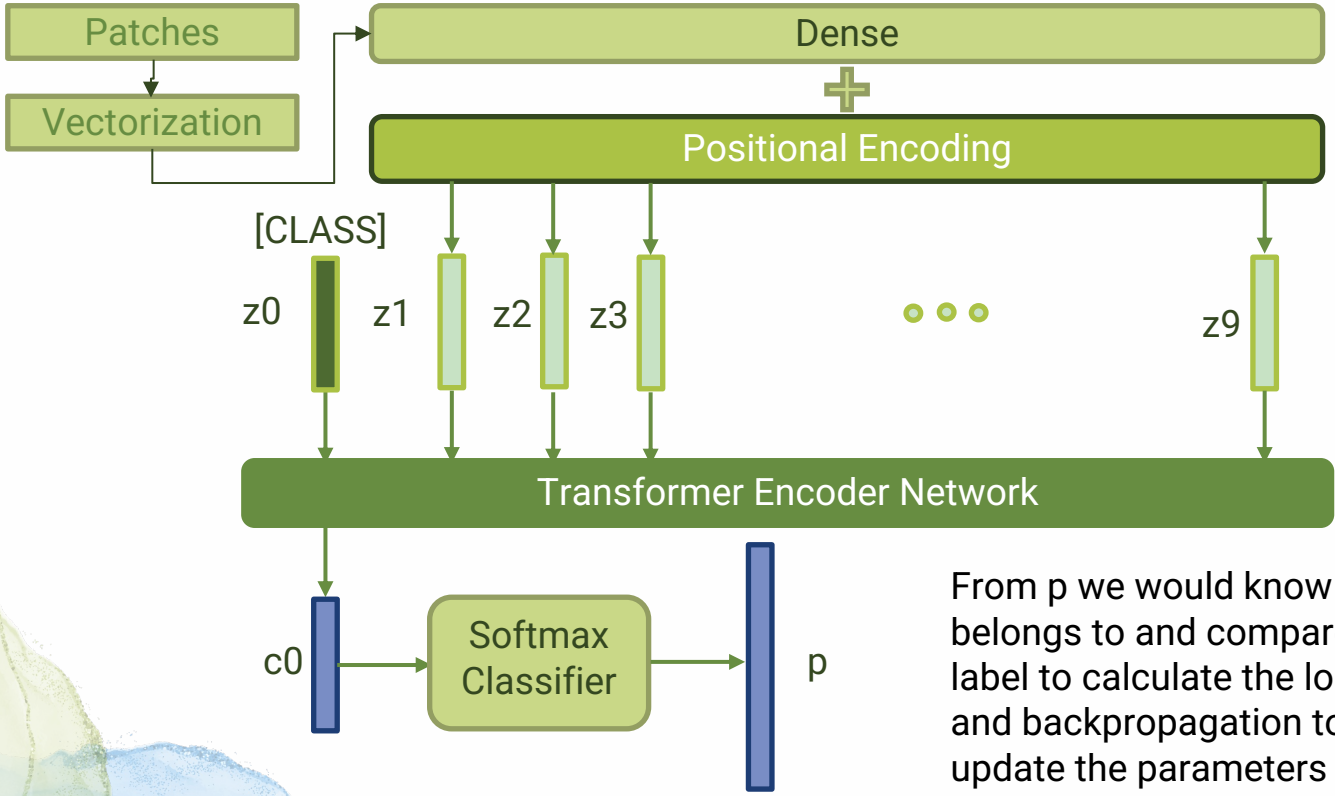
ViT (Vision Transformer)



ViT (Vision Transformer)



ViT (Vision Transformer)



From p we would know which class it belongs to and compare with the true label to calculate the loss function and backpropagation to learn and update the parameters and c0

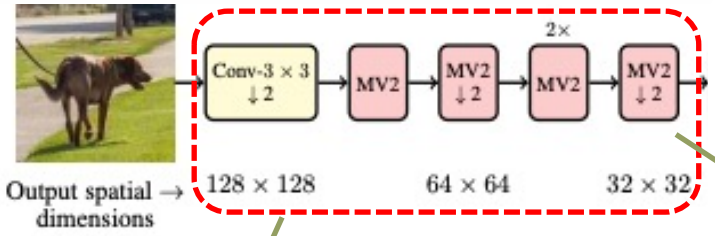
Limitation of ViT

- Too many parameters, requires high computing power
- Lack of spatial inductive bias (not sensitive to spatial information which is really important in CV)
  - Requires much more data to pre-train (compare to CNNs)
  - Heavily relaying on L2 regulation and augmentation (very sensitive to augmentation; accuracy decreases dramatically if no augmentation)
  - Used positional encoding

- Because it introduced positional encoding it would be hard to transfer to other tasks.
  - Length of the sequence of positional encoder is fixed
  - Training on 224 x 224 images and test on 512 x 512 images
    - Theoretically would have better performance but reality is worse

MobileViT (Combination of CNNs and ViT)

- Light-weighted, requires less data to train, mobile friendly
- Combines the advantages of CNNs (spatially inductive bias and less sensitivity to data augmentation) and ViTs (global processing)

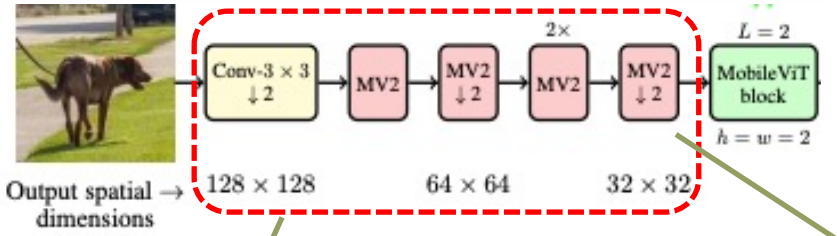


CNNs: Learn local representations

MV2: Inverted Residual Block

MobileViT (Combination of CNNs and ViT)

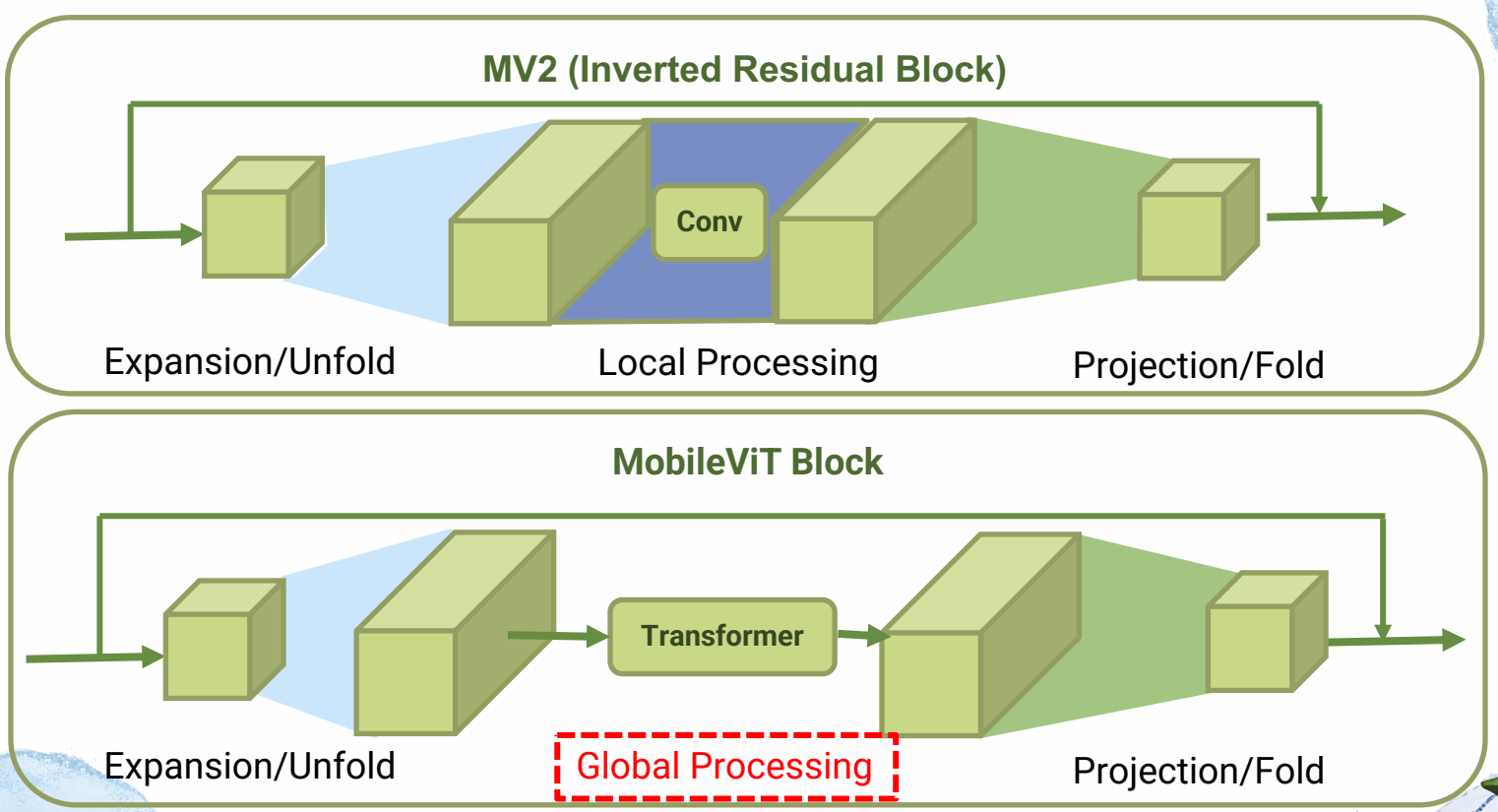
- Light-weighted, requires less data to train, mobile friendly
- Combines the advantages of CNNs (spatially inductive bias and less sensitivity to data augmentation) and ViTs (global processing)



CNNs: Learn local representations

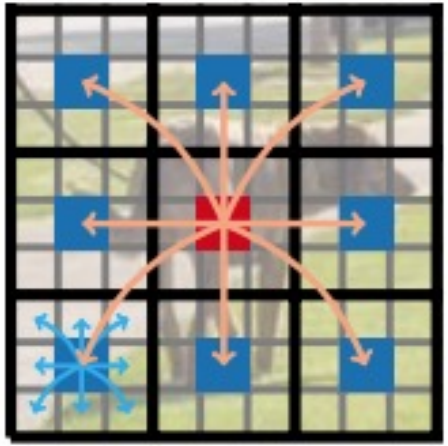
MV2: Inverted Residual Block

MobileViT Block

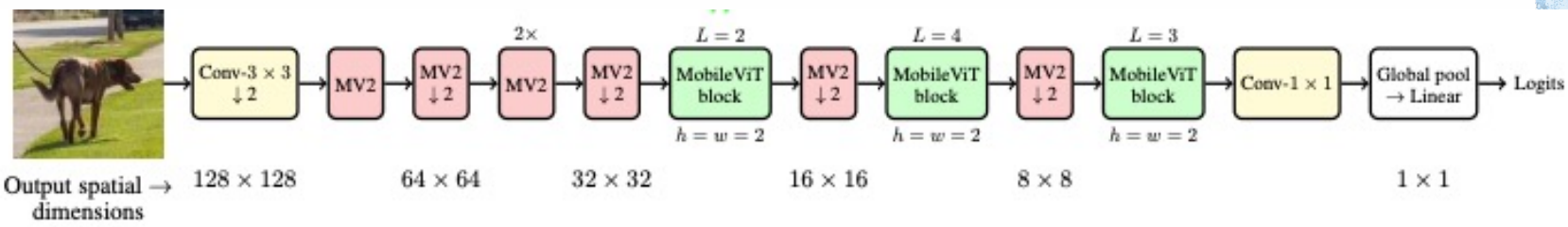


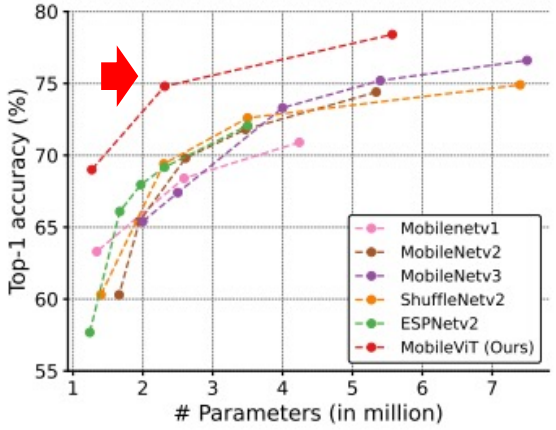
MobileViT Block (Global Processing)

- The **red** pixel learns information from blue pixels with **transformers**
- The **blue** pixel already encodes local information by using **convolution**
- Thus, the **red** pixel actually encodes information from all pixels in an image



MobileViT (Combination of CNNs and ViT)





(a) Comparison with light-weight CNNs

Model	# Params. ⇓	Top-1 ⇑
MobileNetv1	2.6 M	68.4
MobileNetv2	2.6 M	69.8
MobileNetv3	2.5 M	67.4
ShuffleNetv2	2.3 M	69.4
ESPNetv2	2.3 M	69.2
MobileViT-XS (Ours)	2.3 M	**74.8**

(b) Comparison with light-weight CNNs (similar parameters)

Model	# Params. ⇓	Top-1 ⇑
DenseNet-169	14 M	76.2
EfficientNet-B0	5.3 M	76.3
ResNet-101	44.5 M	77.4
ResNet-101-SE	49.3 M	77.6
MobileViT-S (Ours)	5.6 M	**78.4**

(c) Comparison with heavy-weight CNNs

Agenda

Experiment Results Comparison

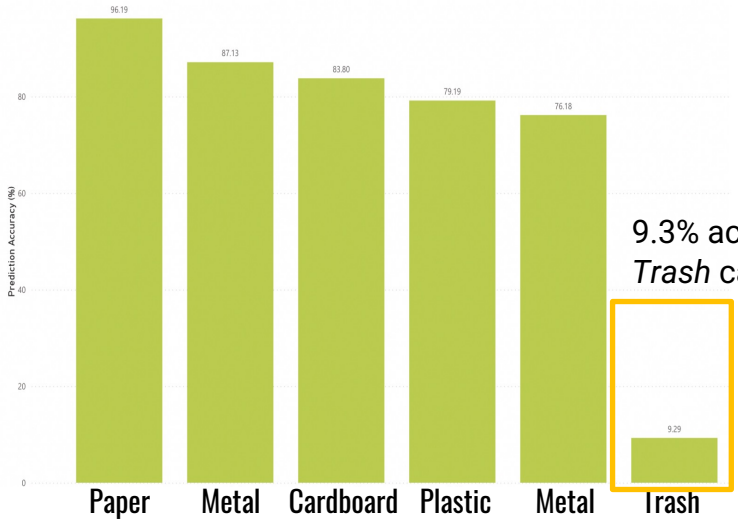
	Model	Frozen Weights	Data Augmentation	Top 1 Accuracy (%)	Top 3 Accuracy (%)
1	ResNet-101	No	Yes	86.98	98.44
2	ResNet-101	No	No	89.32	99.48
3	ResNet-101	Yes	Yes	88.78	98.62
4	ResNet-101	Yes	No	89.17	99.21
5	MobileNet-V2	No	Yes	76.57	94.88
6	MobileNet-V2	No	No	74.41	95.47
7	MobileNet-V2	Yes	Yes	78.54	96.46
8	MobileNet-V2	Yes	No	75.59	97.24
9	EfficientNet-B7	No	Yes	87.99	99.21
10	EfficientNet-B7	No	No	86.22	99.21
11	EfficientNet-B7	Yes	Yes	86.51	98.62
12	EfficientNet-B7	Yes	No	85.63	99.41

- ResNet-101 with unfrozen weights and no data augmentation (Experiment 3) produced the best Top 1 and Top 3 Accuracy

What have the model classified wrong?

Overall top 3 accuracy: 89.86%

Prediction accuracy (%) on experimental dataset



9.3% accuracy in *Trash* category

Experimental Dataset

- Web-scrapped with clear background
- Only masks & diapers
- In good conditions

Training Dataset

- Mixed type of garbage
- In distorted conditions

Agenda

1. Background & Objective
2. Data Overview
3. Methodology
4. Experiment Results
5. **Conclusion**

Conclusion

Limitation

Model trained limited trash categories

Model trained on clear-background images only, limiting its application scope

Unable to leverage Mobile ViT fully

Future Steps

Train the model on more exhaustive trash categories and create fine-grained subcategories for better performance.

Expand the model's capacity by training on diverse trash images, including those found in the wild or underwater, using available resources like waste-datasets-review

Further Reading

AI Implementation in garbage classification

1. https://www.hindawi.com/journals/jece/2022/7608794/?msclkid=a4050768546511f603ff632b554bb80e&utm_source=bing&utm_medium=cpc&utm_campaign=HDW_MRKT_GBL_SUB_BNGA_PAI_DYNA_JOUR_X_PJ_GROUP3&utm_term=\%2Fjournals\%2Fjece\%2F&utm_content=JOUR_X_PJ_GROUP3_JECE
2. https://www.semanticscholar.org/paper/RecycleNet\%3A-Intelligent-Waste-Sorting-Using-Deep-Bircanoglu-Atay/44e9a393795ce7ccd61b7b1c91e7c83d8e42b94d
3. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8930948
4. https://ieeexplore.ieee.org/document/8100117/authors#authors
5. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9165751

MobileNetV2

1. https://arxiv.org/abs/1801.04381
2. https://ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html
3. https://iq.opengenus.org/mobilenetv2-architecture/
4. https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification-8febb490e61c

Further Reading

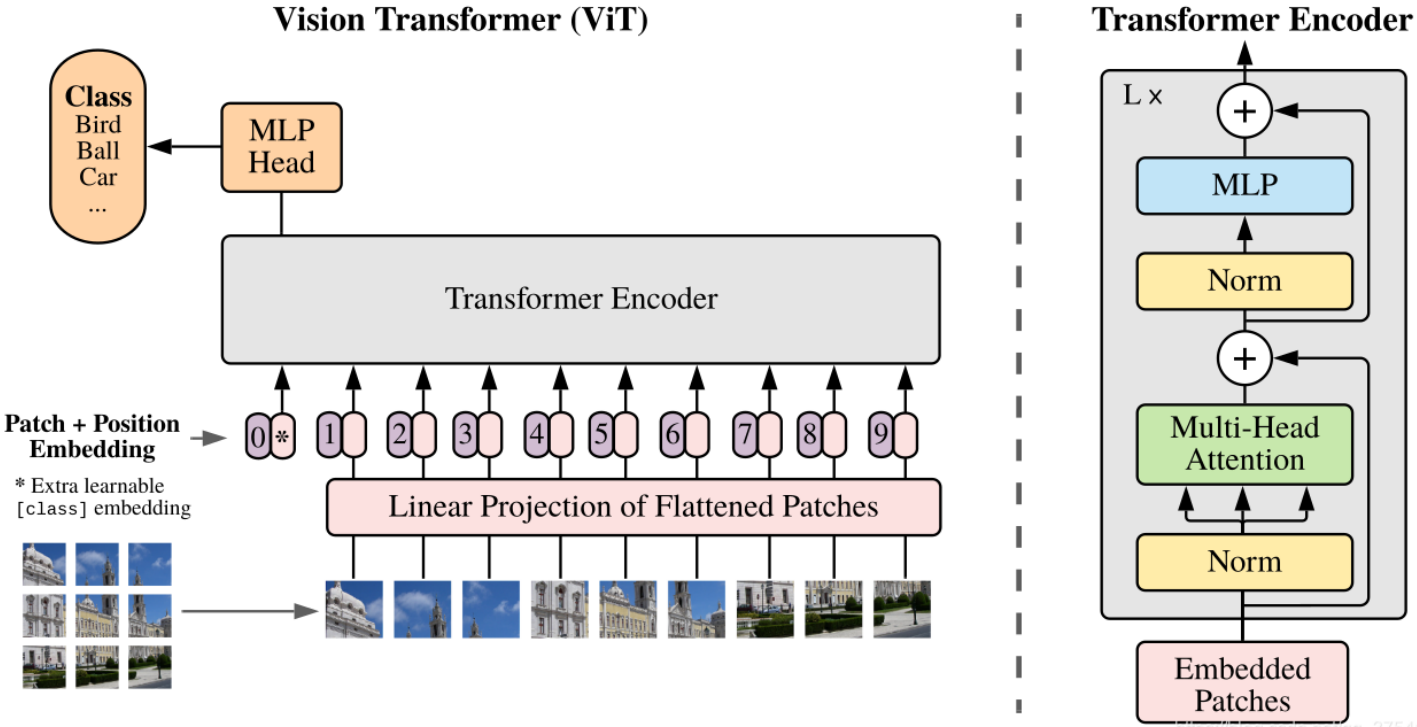
EfficientNet

1. https://ai.googleblog.com/2019/05/efficientnet-improving-accuracy-and.html
2. chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://arxiv.org/pdf/1905.11946.pdf
3. https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/
4. https://keras.io/api/applications/efficientnet/

MobileViT

1. https://idiotdeveloper.com/vision-transformer-an-image-is-worth-16x16-words-transformers-for-image-recognition-at-scale/
2. https://idiotdeveloper.com/what-is-mobilevit/
3. https://arxiv.org/abs/2010.11929 : An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
4. https://arxiv.org/pdf/2110.02178.pdf: MOBILEVIT: LIGHT-WEIGHT, GENERAL-PURPOSE AND MOBILE-FRIENDLY VISION TRANSFORMER

Appendix: ViT




Thank You!
Q & A