Transfer Learning in Deep Learning

Accelerating model training with pre-learned knowledge

MUKESH KUMAR

What is Transfer Learning?

Pre-trained Model Reuse

Apply existing knowledge to new, related tasks

Limited Data Solution

Works when training data is scarce

Resource Efficiency

Avoid expensive training from scratch

Why Use Transfer Learning?

Time & Resource Savings

Reduces computation needs dramatically

Less Labeled Data Required

Train with smaller datasets

Performance Boost

Higher accuracy on limited data

Versatile Applications

Images, text, speech recognition

Typical Workflow

1

Select Model

Choose pretrained base

2

Remove Layers

Cut off final layers

3

Add New Layers

Build task-specific head

4

Configure Freezing

Decide which layers to train

5

Train

Fit on new data

Popular Pretrained Models

For Images

- VGG16 / VGG19
- ResNet50
- InceptionV3
- MobileNet

For Text

- BERT
- GPT
- RoBERTa
- T5

https://keras.io/api/applications/

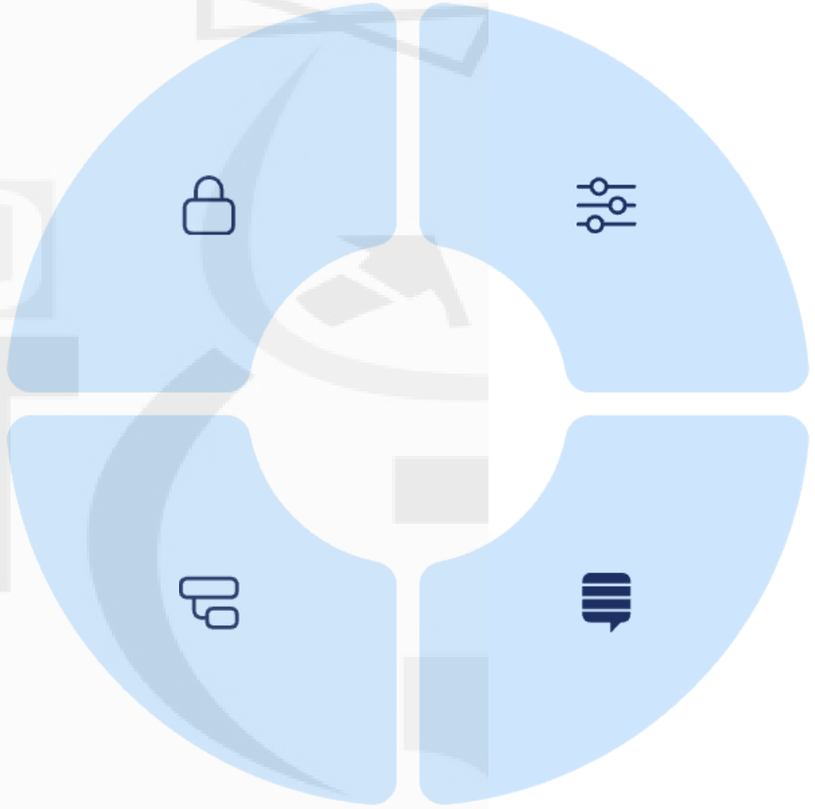
Types of Transfer Learning

Feature Extraction

Freeze base, train only new layers

Fine-Tuning

Unfreeze some or all previous layers



Multi-task Learning

One base model, multiple output tasks

Domain Adaptation

Transfer across different data domains

Use Case – Cats vs Dogs

High Accuracy

Strong results with limited data

Custom Classification Head

Dense layers with softmax output

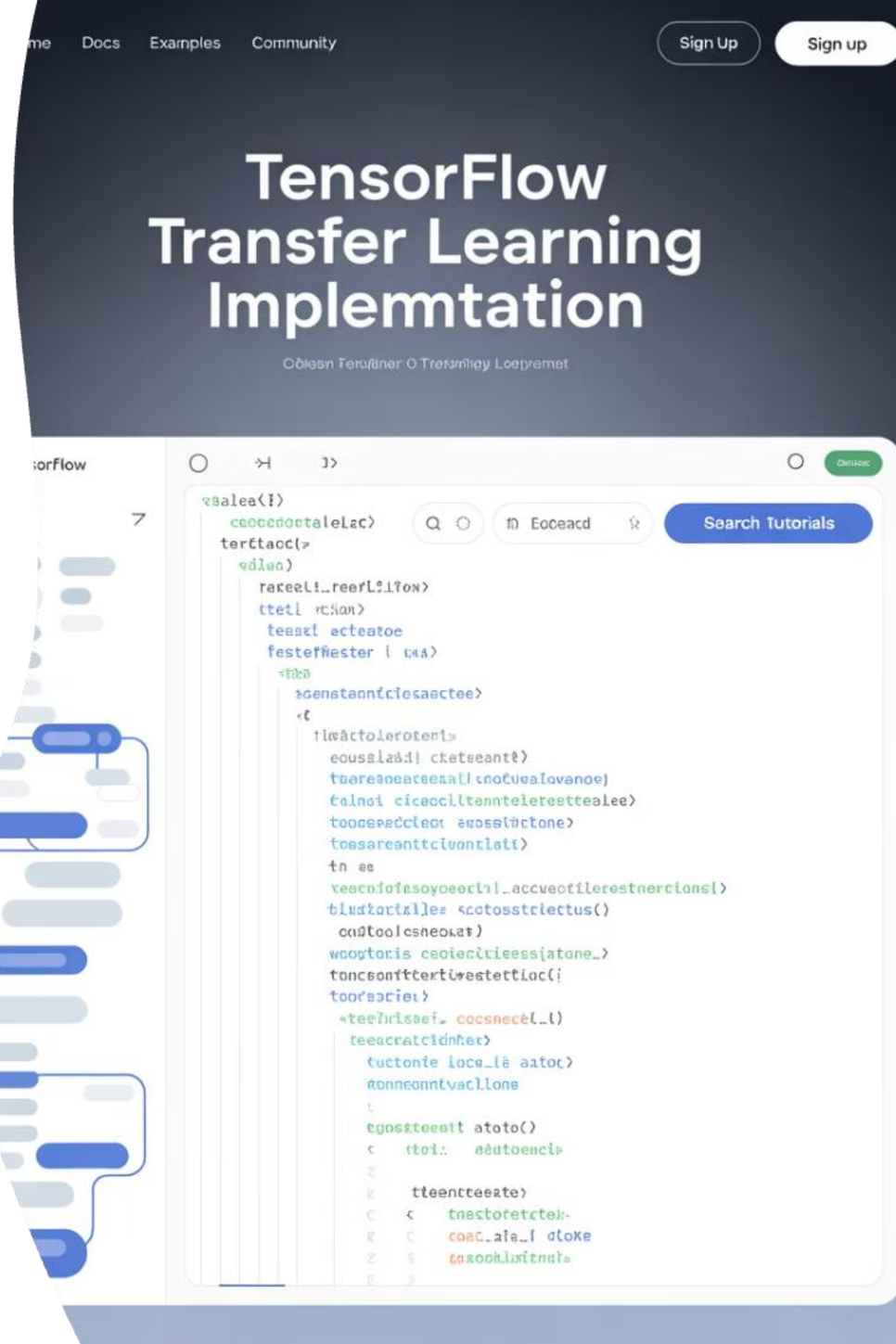
Base: MobileNetV2/ResNet

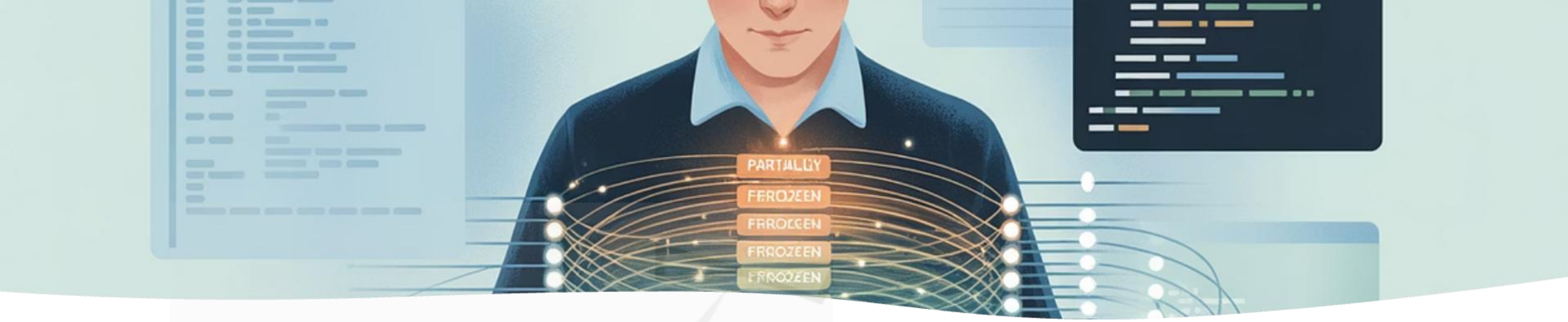
Pre-trained feature extractors

Code Example (TensorFlow/Keras)

```
base_model = tf.keras.applications.MobileNetV2(
    input_shape=(128, 128, 3),
    include_top=False,
    weights='imagenet'
)
base_model.trainable = False  # Freeze base

model = tf.keras.Sequential([
    base_model,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```





Fine-tuning Example

Initial Training

Train with frozen base first

Unfreeze Partially

Enable training on deeper layers

Continue Training

Use smaller learning rate

Practical Tips

Input Normalization

Match base model's preprocessing

Data Augmentation

Rotate, flip, zoom to increase diversity

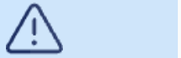
Learning Rate Schedule

Reduce LR during fine-tuning

Progressive Unfreezing

Train head first, then unfreeze gradually

Common Pitfalls



Overfitting Small Data

Use regularization techniques

Wrong Input Size

Match base model's expected dimensions

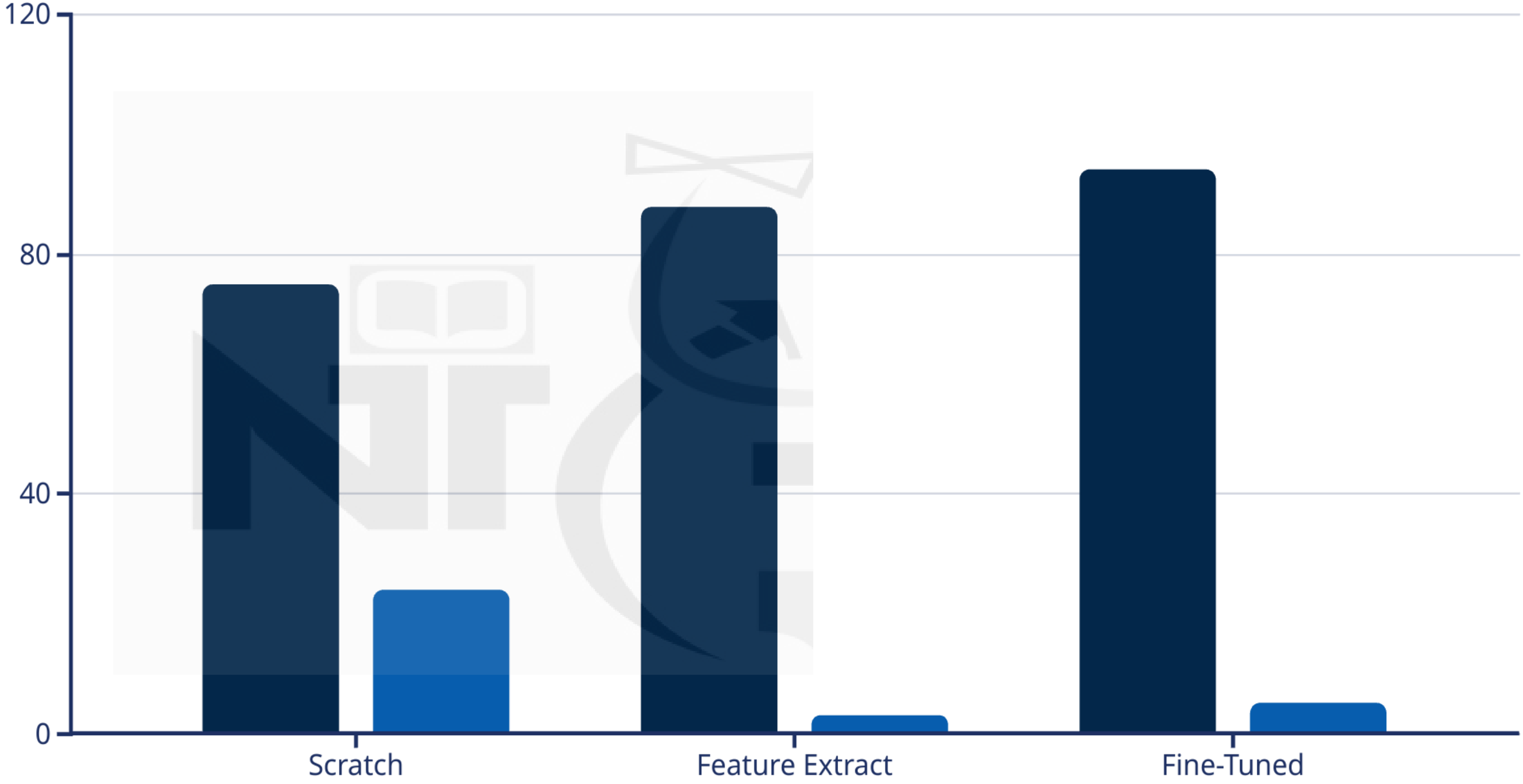
Forgetting to Freeze Layers

Check trainable parameters count

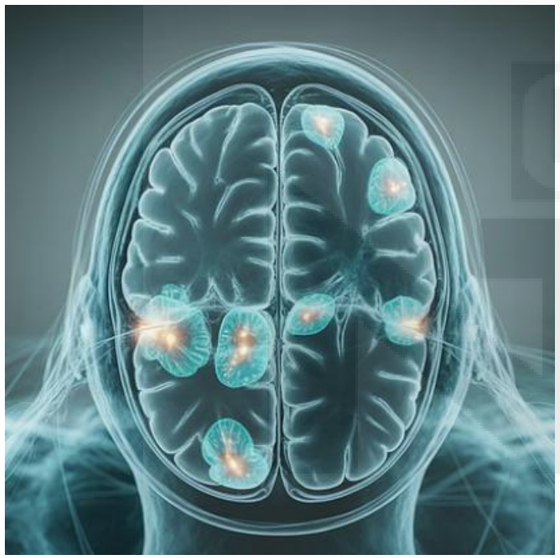
Incompatible Base Models

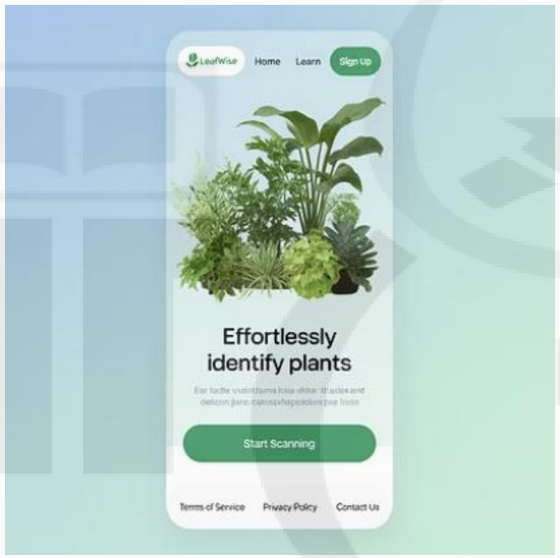
Ensure domain relevance

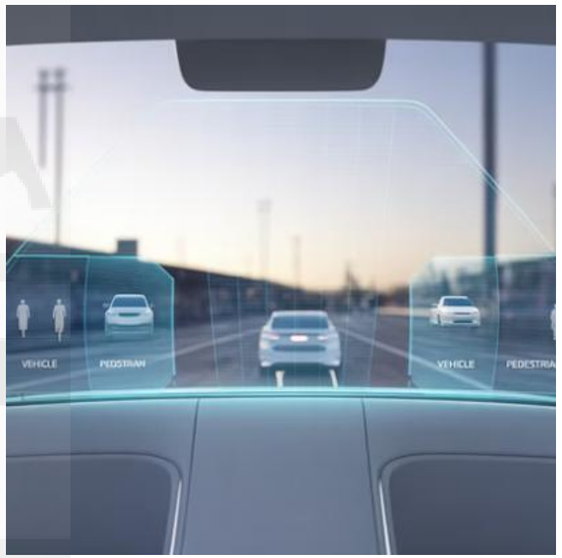
Performance Comparison



Real-World Applications







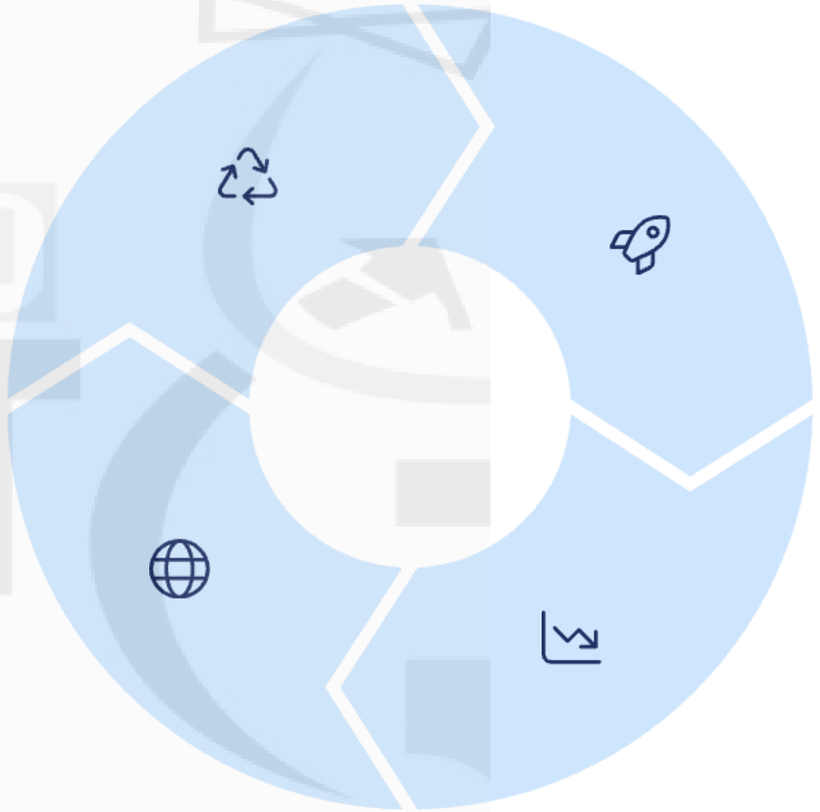
Summary

Leverage Existing Knowledge

Reuse pre-trained models

Wide Applicability

Vision, NLP, audio and more



Resource Efficiency

Save time and computation

Performance Boost

Higher accuracy, less data