# Helwan University

# Neural Networks and Deep learning

| الدرجة | ID | الاسم |
|---|---|---|
| | 20210011 | ابوبكر خالد ابوبكر فرج |
| | 20220177 | زياد تامر مصطفى سعد زغلول |
| | 20220307 | عمار هشام عبدالملك |
| | 20220087 | إياد محمد مجدى |
| | 20220478 | مصطفي درويش مصطفي |

# Classification Project Using the Stanford Cars Dataset

## Project Overview

This project focuses on classifying car types using the Stanford Cars dataset. Four deep learning models were explored and compared:

- VGG-19 (built from scratch)
- ResNet-50 (pretrained)
- Inception V1 (pretrained)
- Vision Transformer (ViT, built from scratch)

The goal of this study is to compare these models in terms of accuracy, architectural design, advantages and disadvantages, and to analyze the reasons behind their performance. Additionally, we discuss the architectural modifications made to adapt each model to the project requirements and dataset constraints.

---

## Stanford Cars Dataset

The Stanford Cars dataset was obtained from Kaggle due to unavailability from the original source. The original dataset contains 196 classes; however, for this project, the number of classes was reduced to the top 20 most frequent classes.

- Total images used: 947
- Training set: 80% (757 images)
- Validation set: 20% (190 images)

Due to the absence of labels in the official test set, the validation set was used as the test set for evaluation purposes. It is important to note that this is a relatively small dataset for a multi-class classification task.

---

## Data Preprocessing and Engineering

Two preprocessing pipelines were implemented: one for training and one for validation.

### Training Pipeline

The training pipeline includes extensive data augmentation to increase data diversity:

- Resize images to 256×256
- Center crop to 224×224
- Horizontal flip with probability 0.5
- Random rotation up to 10 degrees
- Normalization using ImageNet mean and standard deviation

### Validation Pipeline

For validation, data augmentation was minimized to ensure clean evaluation:

- Resize to 256×256
- Center crop to 224×224
- Normalization using ImageNet statistics

### Dataset Handling

- StratifiedShuffleSplit was used to ensure balanced class distribution across training and validation sets
- A custom CustomDataset class was implemented to apply the appropriate preprocessing pipeline
- DataLoader was used with a batch size of 32

---

## VGG-19 Model (Built From Scratch)

### 1. Architectural Philosophy: Simplicity and Depth

The VGG architecture is based on the idea that depth, rather than complex filters, leads to better feature learning. Instead of large convolutional kernels, VGG uses stacked 3×3 convolutions, allowing deeper networks with fewer parameters and more non-linearity.

### 2. Implementation Details

The VGG19_Scratch class was implemented following the original VGG philosophy:

- A configuration list defines the sequence of convolutional and max-pooling layers
- A helper function dynamically constructs the feature extractor
- Extracted features are flattened and passed to the classifier

### 3. Architectural Modification: Funnel Strategy

Standard VGG-19 uses large fully connected layers (4096 → 4096 → Classes). In this project, a tapered funnel architecture was implemented:

4096 → 2048 → 20

This modification:

- Reduces the parameter count by approximately 8 million
- Improves training efficiency
- Acts as a regularization mechanism, reducing overfitting on the small dataset

---

## ResNet-50 Model (Pretrained)

### 1. Architectural Philosophy: Residual Learning

As networks became deeper, training degradation emerged due to vanishing gradients. ResNet addresses this issue using skip connections, allowing information to bypass layers and preserving gradient flow.

### 2. Transfer Learning Strategy

Instead of training from scratch, ResNet-50 pretrained on ImageNet was used:

- Pretrained weights provided robust feature extraction
- Training began from an informed initialization rather than random weights

### 3. Architectural Modification: Head Replacement

To adapt ResNet-50 to the 20-class Stanford Cars task:

- Convolutional layers were frozen to preserve learned features
- The final fully connected layer was replaced:

```
model.fc = nn.Linear(in_features=2048, out_features=20)
```

This approach allows efficient learning while minimizing overfitting.

---

## Inception V1 Model (Pretrained)

### 1. Architectural Philosophy: Multi-Scale Feature Extraction

Inception networks address object scale variation by applying multiple convolutional filters (1×1, 3×3, 5×5) in parallel within each module. This enables the model to capture both fine-grained and large-scale features simultaneously.

### 2. Transfer Learning Approach

The GoogLeNet (Inception V1) architecture pretrained on ImageNet was used:

- Efficient computation via bottleneck 1×1 convolutions
- Strong performance with relatively low computational cost

### 3. Architectural Modification

The final classification layer was replaced to support 20 classes:

```
model.fc = nn.Linear(in_features=1024, out_features=20)
```

This preserves the powerful multi-scale feature extractor while adapting the classifier to the project task.

---

## Vision Transformer (ViT) Model (Built From Scratch)

### 1. Architectural Philosophy: Images as Sequences

Unlike CNNs, Vision Transformers treat images as sequences of patches, enabling global context modeling through self-attention rather than local convolution.

- Image is split into 16×16 patches
- Each patch is treated as a token
- Self-attention captures long-range dependencies across the image

### 2. Model Implementation

The VisionTransformer model consists of:

- Patch Embedding: Converts the image into 196 tokens
- Positional Embedding: Injects spatial information
- Transformer Encoder: Uses multi-head self-attention

### 3. Custom Lightweight Configuration

Due to dataset limitations, a shallow ViT architecture was designed:

- Embedding dimension: 384
- Number of attention heads: 6
- Transformer depth: 4 blocks

This configuration significantly reduces model complexity while retaining the core advantages of attention-based learning, helping to prevent overfitting on the small dataset.

---

## Conclusion

This project demonstrates a comparative study of CNN-based and transformer-based architectures on a small-scale car classification task. Each model exhibits distinct strengths and limitations, highlighting the importance of architectural choice, transfer learning, and dataset size when designing deep learning systems.