**Algorithm 1: Training and Evaluation of Deep Learning Models**

1. Set training parameters:

 - batch\_size ← 16

 - image\_size ← (224, 224)

 - epochs ← 300

2. For each class in Datasets:

 count ← number of images in class

 log class distribution

3. Initialize ImageDataGenerator with:

 - rescale, rotation, shifts, zoom, flip, brightness adjustment (±20%)

 - validation\_split = 0.2

4. Create training and validation generators from Datasets using ImageDataGenerator (80% training, 20 validation)

5. For each model architecture do:

 a. Load base model (YOLOv8, MobileNetV3, ResNetV2, EfficientNetV2, VGG16, ConvNeXt-Tiny, SwinV2-Base, DaViT) with pretrained weights (e.g., ImageNet)

 b. Freeze layers if needed (partial fine-tuning)

 c. Build final model:

 - BatchNormalization

 - GlobalAveragePooling

 - Dropout

 - Dense softmax output layer (num\_classes)

 d. Compile model:

 - Optimizer: Adam

 - Loss: SparseCategoricalCrossentropy

 - Metric: Accuracy

 e. Define callbacks:

 - ReduceLROnPlateau

 - EarlyStopping

 f. Train model on training set with validation

 g. Save trained model in Keras format

 h. Convert and save model in TFLite format

 i. Evaluate model on validation set:

 - Predict class probabilities

 - Compute predictions using argmax

 - Generate normalized confusion matrix

 - Print classification report (Precision, Recall, F1)

 - Plot F1-confidence curve

 - Compute mean average precision (mAP@0.5 and mAP@0.5:0.95)