Deep Learning Report4

Shivangi Rai 2321003

March 27, 2024

1 Introduction

In this report, we evaluate the performance of various pre-trained deep learning models on the Street View House Numbers (SVHN) dataset. The goal is to assess how well these models perform in recognizing digits from real-world street images.

2 Data Preprocessing

In the provided code, data preprocessing is applied to the SVHN dataset before feeding it into the deep learning models for training and evaluation. Let's break down the preprocessing steps in detail:

1. **Resize Images**: The images in the SVHN dataset are resized to a fixed size of 224x224 pixels. This resizing step is necessary because deep learning models like VGG and ResNet were trained on ImageNet dataset, where the images are of fixed size. Resizing ensures that the input images have consistent dimensions, which is required for feeding them into the pre-trained models without causing errors.

2. **Convert to Tensor**: The images are converted into PyTorch tensors. PyTorch tensors are multi-dimensional arrays that are compatible with deep learning frameworks like PyTorch. Converting images to tensors allows for efficient computation during training and evaluation, as tensors can be directly operated upon by the neural network models.

3. **Normalization**: The pixel values of the images are normalized using the mean and standard deviation of the ImageNet dataset. The mean and standard deviation used for normalization are $[0.485, 0.456, 0.406]$ and $[0.229, 0.224, 0.225]$ respectively. This normalization step helps in standardizing the pixel values across different channels (RGB) and brings them to a similar scale. Normalization helps in stabilizing and speeding up the training process of deep neural networks.

Normalization Formula:

$$normalized\_image = \frac{original\_image - mean}{std\_dev}$$

- $normalized\_image$ is the image after normalization.
- $original\_image$ is the image before normalization.
- $mean$ is the mean value used for normalization.
- std_dev is the standard deviation value used for normalization.

By applying these preprocessing steps, the input data is appropriately formatted and scaled to ensure effective training and evaluation of the deep learning models on the SVHN dataset.

article

3 Methodology

In the provided code, model selection and modification involve choosing pre-trained deep learning models and adapting them to the task of digit recognition on the SVHN dataset. Let's break down this process in detail:

3.1 Model Selection

1. **Pre-Trained Models**: The code selects several popular pre-trained deep learning models. These models have been pre-trained on large-scale datasets like ImageNet for tasks such as image classification. The selected models are:
    - AlexNet
    - VGG16
    - ResNet18
    - ResNet50
    - ResNet101
2. **Choice Rationale**: These models are chosen due to their widespread usage, availability in the torchvision library, and their effectiveness in various computer vision tasks. Each model offers different architectures and complexities, allowing for a diverse range of experimentation.

3.2 Model Modification

After selecting the pre-trained models, the code modifies them to adapt them to the specific task of digit recognition on the SVHN dataset. This modification involves adjusting the output layer of each model to match the number of classes in the SVHN dataset (which is 10, corresponding to digits 0-9).

1. **Identification of Last Layer**: For each selected model, the code identifies the last layer responsible for classification. In convolutional neural networks like VGG and ResNet, this is typically a fully connected layer at the end of the network.
2. **Adaptation of Last Layer**: The last layer of each model is replaced with a new fully connected layer having 10 output units. This new layer enables the model to predict the probability distribution over the 10 classes of digits present in the SVHN dataset.
3. **Modification Function**: To streamline this process, a function named `modify_model` is defined. This function takes a pre-trained model as input, identifies the last layer, and replaces it with a new layer suited for the digit recognition task.

3.3 Outcome

The outcome of this process is a set of modified pre-trained models, each tailored to the task of digit recognition on the SVHN dataset. These models retain the features learned during pre-training on ImageNet while being adapted to the specific characteristics of the SVHN dataset.

3.4 Importance

Model selection and modification are crucial steps in deep learning projects. By leveraging pre-trained models, developers can benefit from the extensive knowledge encoded in these models through pre-training on large datasets. Modification allows for fine-tuning these models to suit the specific requirements of the target task, thereby accelerating the development process and improving performance.

article

4 Training and Evaluation

In the provided code, the deep learning models are trained and evaluated for digit recognition on the SVHN dataset. Let's discuss the training and evaluation process in detail:

4.1 Training

1. **Training Data Loader**: The SVHN dataset is loaded using a data loader (`train_loader`). This data loader fetches batches of training samples from the dataset, allowing for efficient processing during training.

2. **Training Loop**: The models are trained using a training loop, where they are iteratively updated to minimize the loss function. Each epoch consists of multiple iterations over the entire training dataset.

3. **Optimizer**: The Adam optimizer is used for training the models. It is a popular choice for deep learning tasks due to its adaptive learning rate and momentum properties.

4. **Loss Function**: The Cross-Entropy Loss function is employed, which is commonly used for multi-class classification tasks like digit recognition. It computes the loss between the predicted class probabilities and the ground truth labels.

4.2 Evaluation

1. **Test Data Loader**: Similarly, a data loader (`test_loader`) is used to load batches of test samples from the SVHN dataset. These samples are used for evaluating the trained models.

2. **Evaluation Loop**: The trained models are evaluated using a separate evaluation loop, where they process the test samples to make predictions. The accuracy of the predictions is then computed by comparing them against the ground truth labels.

3. **Model Evaluation**: The accuracy of each model is calculated as the percentage of correctly predicted labels out of the total number of test samples.

5 Results

The results of training and evaluation provide insights into the performance of the models on the digit recognition task.

Table 1: Accuracy of Different Models on SVHN Dataset

Model	**Accuracy (%)**
AlexNet	19.59
VGG16	19.59
ResNet18	95.30
ResNet50	94.52
ResNet101	94.86

6 Conclusion

The disparity in performance between the older architectures (AlexNet and VGG16) and the newer architectures (ResNet18 and ResNet50) highlights the importance of model selection and architectural advancements in deep learning. The relatively low accuracy of AlexNet and VGG16 underscores the need for selecting appropriate architectures tailored to the task at hand. In this case, ResNet architectures, with their deeper and more sophisticated designs, prove to be more suitable for digit recognition on the SVHN dataset. The high accuracy achieved by ResNet18 and ResNet50 reaffirms the effectiveness of these architectures and suggests that they are well-suited for digit recognition tasks, even without extensive fine-tuning or additional modifications.

In conclusion, the choice of deep learning architecture significantly impacts model performance, and selecting modern, well-suited architectures like ResNet18 and ResNet50 can lead to substantial improvements in accuracy for digit recognition tasks like those found in the SVHN dataset.