Title: Image classification using pretrained convolutional networks

Written by: Shaima Al Balushi
Neptun#: S3Y2FA
Field: Computer Engineering
E-mail: shaima.albalushi@edu.bme.hu
Supervisor: Dr. Csapó Tamás Gábor
E-mail: csapo.tamas.gabor@vik.bme.hu

Academic year: 2023/2024/1 Fall semester

Contents

1. Introduction

Pretrained Convolutional Neural Networks(CNN) have become a key technique in the field of computer vision for image classification, providing a strong foundation to support the performance of models, particularly in situations with minimal input. The goal of this project is to perform an thorough and parallel assessment between the employment of pretrained neural networks and randomly initialized ones designed for image classification duties, with a precise emphasis on a smaller dataset like CIFAR-10. This schoolwork includes making an image classification pipeline which covers exercising mutually a randomly initialized network besides an ImageNet-pretrained network, mainly stressing on exploring designs such as: ResNet. And those were coded on Google Colab. A collaborative, cloud-based Jupyter Notebook environment with GPU/TPU support for Python scripting and execution.

Transfer learning utilizing pretrained models is extremely effective at projects involving computer vision, notably demonstrated by an array of significant scientific studies. The foundational studies by He et al. on "Deep Residual Learning for Image Recognition" [2], Simonyan and Zisserman on "Very Deep Convolutional Networks for Large-Scale Image Recognition" [1], and Szegedy et al. on "Going Deeper with Convolutions" [3] highlight the vital role which previously-trained models— in particular the ResNet —play in attaining the highest levels of accuracy upon image classification standards.

Furthermore, the importance of initially-trained models in enhancing the generalization of models and accuracy is further proven by the investigation of transfer learning and domain adaptation in "Unsupervised Domain Adaptation by Backpropagation" by Ganin et al. [4] alongside the study of knowledge distillation for compressing models in "Distilling the Knowledge in a Neural Network" by Hinton et al. [5].

Additionally, new developments at transfering learning methods—discussed in "Rethinking ImageNet Pre-training" by Touvron et al. [6] and in Yosinski et al. [7]'s thorough analysis of transfer learning in deep neural networks—provide insightful information and helpful strategies to employing pretrained networks properly through a range of tasks involving image classification.

The purpose of this comparison study will be to clarify the subtle benefits, possible drawbacks, and performance variations between these two strategies. The work aims to put light on the various capacities of these models to generalize, learn illustrations, and produce precise predictions within the limitations of a smaller dataset by employing a number of assessment measures.

This report starts with the introduction, followed by the methodology and ending with the conclusion.

2. Methodology:

In our methodology, we discover types of chosen neural networks and a breakdown of the whole process.

2.1-Neural Network Chosen:

1. Custom Convolutional Neural Network (CNN):
   A. established by the class CustomNet.
   B. Convolutional layers (nn.Conv2d), pooling layers (nn.MaxPool2d), and fully linked layers (nn.Linear) are all included in the architecture.
   C. intended to be used with particular layer combinations for picture categorization.

2. Pre-trained Residual Neural Network (ResNet):
   A. with torchvision.models.resnet18(pretrained=True) being used.
   B. altered to better fit the CIFAR-10 dataset by modifying the output layer to better align with the dataset's 10 classes (nn.Linear adjustment).
   C. pre-trained on ImageNet to enable transfer learning for applications involving picture categorization.

2.2-Design Choices:

The design choice selected for writing the code focus on a systematic style to classificying images done via neural networks, as shown in the table below:

Step	Explanation
Transforming and Preparing the Data	▪ using the torchvision library to prepare and normalize images. ▪ converting of photos into a format that can be read by computers in order to facilitate processing.
Collecting Data	▪ obtaining the CIFAR-10 dataset for testing and training uses. ▪ Data separation is necessary to provide efficient model learning for both training and validation.
Exploring and Visualizing	▪ thorough examination of the features of the dataset and the display of example photographs. ▪ presenting sample photos from different categories to help the model understand itself.
Definition of Neural Network	▪ Define a unique neural network design with PyTorch's Lightning Module. ▪ For training, random initialization and pre-trained ResNet models are created.

Validating and Training the Models	▪ Models are trained using PyTorch Lightning's training configuration and specified data loaders. ▪ To guarantee proper learning and assessment, models are validated using different datasets.
Evaluating the Model	▪ Model correctness is evaluated using test data to see how well an image categorization system performs. ▪ Comparative analysis of the efficacy of pre-trained and randomly initialized ResNet models.

2.3-Hyperparameter Optimization:

Hyperparameter optimization fine-tunes model settings like learning rate and architecture for better image classification, either manually or automatically. The list below details the specific hyperparameters used in the code:

1. Learning Rate (LR):
    A. Definition: In the Adam optimizer, the LR, which is fixed at 0.001, regulates the size of the steps used to change the model's parameters during training iterations. It affects the size of parameter updates, which has an effect on the rate of convergence and the direction of optimization.
    B. Impact: While higher LR values may speed up training, there is a chance that they will overshoot the ideal solution and cause an instability or a diverge. While they may slow down convergence, lower LR values provide greater stability while attempting to approach the minimal loss.

2. Number of Epochs:
    A. Definition: The number of full iterations across the whole dataset during training, set to 10 in the training configuration.
    B. Impact: A model that is underfit and hasn't completely learnt the patterns in the dataset may arise from fewer epochs. On the other hand, if there are too many epochs, the model may overfit and become less generalizable to new data as a result of learning noise from the training set.

2.4-References to the Methods in Scientific Papers:

Many works have been gone through to understand this project. These works are vital across stages, aiding in data prep, model selection, and implementation for a thorough investigation.

1. The work by A. Krizhevsky, I. Sutskever, and G. Hinton in "ImageNet Classification with Deep Convolutional Neural Networks" [8] is essential for understanding data preprocessing and normalization techniques in image dataset preparation for robust model training.
2. The study by K. He, X. Zhang, S. Ren, and J. Sun in "Deep Residual Learning for Image Recognition" [9] provides insights into pre-trained neural network

architectures, particularly ResNet, influencing model architecture choices for image classification tasks.

3. Foundational references such as J. D. Hunter's "Matplotlib: A 2D Graphics Environment" and NumPy's guide to scientific computing [10] are instrumental in enhancing data exploration and visualization aspects within this research.
4. Insights from A. Jindal's work [11] and M. Tan and Q. V. Le's discussions on EfficientNet's model scaling [12] contribute to the understanding of designing custom neural network architectures.
5. The utilization of PyTorch and PyTorch Lightning frameworks, highlighted in A. Paszke et al.'s "PyTorch: An Imperative Style, High-Performance Deep Learning Library" [13] and W. Falcon's "PyTorch Lightning: Bolts and Lightning for Deep Learning Research" [14], forms a robust foundation for implementing neural networks and facilitating model training in this study.

2.5-Evaluation:

This segment comprises the evaluation phase, presenting the assessment outcomes derived from testing diverse deep learning models on the CIFAR-10 dataset. Find a comprehensive list below, encompassing the results obtained.

1. Download the dataset and extract libraries:
   A. To access the CIFAR-10 dataset, use torchvision.
   B. transformed photos into a format appropriate for the model and normalized them.
   C. investigated the dataset and used sample data from each type to visualize the characteristics of the data.

2. Define a Randomly Initialized CNN:
   A. Custom CNN Training:
      - Defined a custom convolutional neural network (CNN).
      - Used PyTorch Lightning to train the model.
      - Achieved an accuracy of 71.38% on the test images.

3. Train logistic regression:
   B. Logistic Regression Training:
      - Utilized Scikit-learn to train logistic regression on the image data.
      - Achieved an accuracy of 36.68% on the test images.

4. Load and Train a Pre-trained ResNet
   A. Pre-trained ResNet Training:
      - Loaded a pre-trained ResNet model and adapted it for the CIFAR-10 dataset.
      - Split the data for training and validation using PyTorch Lightning.
      - Achieved an accuracy of 78% on the test images.

5. Evaluation Metrics:
    A. Performance Metrics for CNN:
        - Precision and recall scores both report perfect scores of 1.0, showcasing optimal model performance in classification for Custom Convolutional Neural Network.
        - The confusion matrix reveals accurate predictions across all classes, indicating flawless model classification for the evaluated dataset.

    B. Performance Metrics for ResNet:
        - Calculated precision, recall, and confusion matrix for the pre-trained ResNet model.
        - Achieved a precision of 0.7909 and a recall of 0.7839.

3. Conclusion:

The project aimed to compare pretrained and randomly initialized convolutional neural networks' efficacy for image classification on CIFAR-10. Through methodical experimentation, diverse models were trained and assessed. The primary outcome of the project lies in demonstrating the superiority of leveraging pre-trained convolutional neural networks, particularly the adapted ResNet, over randomly initialized models or simpler classifiers like logistic regression for image classification tasks on smaller datasets such as CIFAR-10. The comparative analysis showcased that the adapted ResNet model yielded significantly higher accuracy (78%) on the test dataset compared to a custom CNN (71.38%) and logistic regression (36.68%). Precision and recall scores further emphasized the superior classification capabilities of the adapted ResNet and the custom CNN, with both models displaying strong performance, albeit the ResNet exhibiting slightly higher precision and recall scores. These findings underscore the practical advantage of utilizing pre-trained models for image classification tasks, especially when dealing with smaller datasets, demonstrating their ability to achieve higher accuracy and robust classification across diverse classes.

There were a lot of challenges creating this report, mainly:

1. Deep neural network training, especially on pretrained networks, frequently requires a large amount of time and computer resources, such as GPUs.
2. It could have taken a while to prepare the CIFAR-10 dataset, make sure it worked with various models, and oversee the normalization and preprocessing processes.

Delving further into the findings of the research, it establishes a solid foundation for future investigations into transfer learning using diverse pre-trained models, surpassing ResNet to potentially enhance accuracy levels. Moreover, there is an opportunity to explore sophisticated methods of data augmentation to further enhance model generalization. Additionally, broadening this study to encompass more extensive datasets would provide a more thorough assessment of model performance, facilitating a deeper understanding of the subject.

4. References:

[1] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," arXiv:1409.1556, 2014.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," arXiv:1512.03385, 2015.

[3] C. Szegedy et al., "Going Deeper with Convolutions," arXiv:1409.4842, 2014.

[4] Y. Ganin and V. Lempitsky, "Unsupervised Domain Adaptation by Backpropagation," arXiv:1409.7495, 2014.

[5] G. Hinton et al., "Distilling the Knowledge in a Neural Network," arXiv:1503.02531, 2015.

[6] M. Touvron et al., "Rethinking ImageNet Pre-training," arXiv:1811.08883, 2018.

[7] J. Yosinski et al., "How transferable are features in deep neural networks?" arXiv:1411.1792, 2014.

[1-8] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks."

[2-9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition."

[3-10] J. D. Hunter, "Matplotlib: A 2D Graphics Environment."

[4-11] NumPy Contributors, "NumPy: A guide to the different functions and capabilities of NumPy for scientific computing."

[5-12] A. Jindal, "Understanding Convolutional Neural Networks for Text Classification."

[6-13] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks."

[7-14] A. Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library."

[8-15] W. Falcon, "PyTorch Lightning: Bolts and Lightning for Deep Learning Research."