Exploring Deep Learning Techniques for Food Classification

Kalinowski, Eric

In this project, I explore different deep learning techniques to classify food images from the Food-101 dataset. The goal is to compare the performance of a custom convolutional neural network (CNN) against a pre-trained DenseNet121 model, both with and without data augmentation. To enhance the dataset, I implemented a conditional generative adversarial network (cGAN) to generate synthetic food images, aiming to improve the model's generalization. The results show that using a pre-trained model significantly outperforms the custom CNN, while data augmentation with the cGAN introduces mixed results.

***Index Terms*—Deep Learning, Convolutional Neural Network (CNN), Transfer Learning, Pre-trained Models, DenseNet121, Generative Adversarial Network (GAN), Conditional GAN, Data Augmentation, Image Classification, Food-101 Dataset.**

I. Dataset Details

The Food-101 dataset is widely recognized for image classification tasks in the food recognition domain. It contains 101 classes, each representing a specific type of food, with 1,000 images per class, making a total of 101,000 images.

The dataset is divided into two parts:

- Training set: 75% of the images (750 per class).
- Testing set: 25% of the images (250 per class).

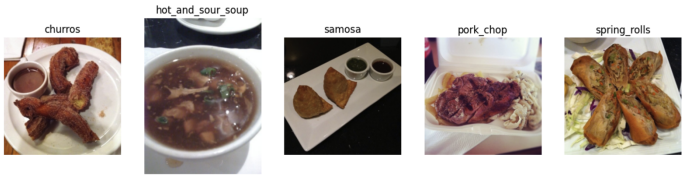

Fig. 1. Food-101 Samples.

As shown in Figure 1, the images vary in size, orientation, and quality, reflecting real-world conditions. This diversity makes Food-101 a challenging dataset for classification. Additionally, the presence of noisy labels adds complexity and underscores the importance of robust models.

II. Model Baseline

This section describes the custom Convolutional Neural Network (CNN) model used as the baseline.

A. Preprocessing

I applied the following transformations:

1) Resize to 128 × 128 pixels: All images were resized to have consistent dimensions, simplifying input handling.
2) Convert to tensors: Each image was converted into a tensor for efficient processing in PyTorch.
3) Normalization: Pixel values were scaled to the range $[-1, 1]$ with a mean of $[0.5, 0.5, 0.5]$ and a standard deviation of $[0.5, 0.5, 0.5]$ to ensure stable training.

B. Architecture of the Network

- Convolutional layers: Five layers, with filters progressively increasing from 64 to 1024 to capture features of varying complexity.
- Pooling layers: Max-pooling layers reduce spatial dimensions after each convolution, retaining the most important features.
- Batch normalization: Applied after each convolutional layer to stabilize training and improve convergence.
- Fully connected layers: Two dense layers, with the final layer producing outputs for 101 classes.

The network uses ReLU activations for non-linearity and dropout for regularization.

C. Training Process

The model was trained for 15 epochs using this settings:

- Optimizer: Adam with a learning rate of 0.0001.
- Loss function: Cross-Entropy Loss, which eliminates the need for explicit softmax in the final layer.
- Learning rate scheduler: StepLR, which halved the learning rate every 5 epochs to fine-tune training.
- Batch size: 32, selected for a balance between training speed and memory usage.

D. Results

- Top-1 accuracy: 31.14%
- Top-5 accuracy: 59.72%

These results illustrate the challenges of the Food-101 dataset and the limitations of a custom CNN. While the model captured basic patterns, it struggled to generalize effectively, highlighting the need for more advanced techniques or pre-trained models.

III. Pretrained DenseNet Model

This section describes how I fine-tuned the DenseNet121 model, a pretrained convolutional neural network. By leveraging transfer learning, this approach aimed to improve performance on a complex dataset while minimizing computational demands.

A. Preprocessing

To prepare the dataset for DenseNet121, the following transformations were applied:

1) Resize to 256 pixels: All images were resized to ensure the smaller side measured 256 pixels.
2) Center crop to 224 × 224 pixels: This ensured the input size matched the requirements of the DenseNet121 architecture.
3) Normalization: Pixel values were standardized with a mean of $[0.485, 0.456, 0.406]$ and a standard deviation of $[0.229, 0.224, 0.225]$, consistent with ImageNet preprocessing standards.

B. Network Architecture

The DenseNet121 model is a pretrained convolutional neural network originally trained on the ImageNet dataset. For this experiment:

- Feature layers: The feature extraction layers were frozen to retain the general patterns learned from ImageNet.
- Classifier: The final fully connected layer was replaced with a linear layer of size 101 to align with the number of classes in Food-101.

This adjustment allowed the model to adapt its pretrained knowledge to the specific requirements of the Food-101 dataset.

C. Training Process

The model was trained using this settings:

- Loss function: Cross-Entropy Loss, suitable for multi-class classification.
- Optimizer: Stochastic Gradient Descent (SGD) with a learning rate of 0.001 and a momentum of 0.9.
- Batch size: 32, chosen to balance computational efficiency and training stability.
- Epochs: 15, with metrics for both training and validation tracked at each epoch.

Freezing the feature layers reduced the risk of overfitting and lowered the computational cost.

D. Results

- Top-1 accuracy: 57.77%, more than double the accuracy of the baseline model.
- Top-5 accuracy: 82.06%, demonstrating strong performance when considering the top five predictions.

These results underline the effectiveness of transfer learning with pretrained models for complex classification tasks such as Food-101, showcasing their ability to leverage prior knowledge for improved performance.

IV. Conditional GAN for Data Augmentation

To enhance the dataset and improve model performance, I implemented a conditional generative adversarial network (cGAN). This network generates synthetic images conditioned on class labels, adding diversity and balance to the dataset.

A. Network Architecture

- Generator: This component creates synthetic images by combining random noise vectors (latent vectors) with one-hot encoded class labels. It uses ConvTranspose2d layers to upsample the latent vectors into realistic images of size 64×64. Batch normalization and ReLU activation functions ensure stable training and non-linearity.
- Discriminator: The discriminator distinguishes real images from synthetic ones. It combines input images with one-hot encoded labels to evaluate authenticity, using convolutional layers, batch normalization, and LeakyReLU activation. A final Sigmoid layer outputs the probability of an image being real.

B. Training Process

The training process involved iterative optimization of both components:

- Discriminator training: The discriminator was updated twice per batch using both real and synthetic images, minimizing the binary cross-entropy loss (BCE).
- Generator training: The generator was trained to produce realistic images capable of fooling the discriminator, also optimizing the BCE loss.

The key hyperparameters for training were:

- Learning rate: 0.0002
- Latent vector size: 100
- Batch size: 128
- Epochs: 25

C. Results

The generator and discriminator losses across epochs are shown in Figure 2. The generator loss gradually decreased, indicating its ability to produce more realistic images, while the discriminator loss stabilized, reflecting a balance between the two networks.

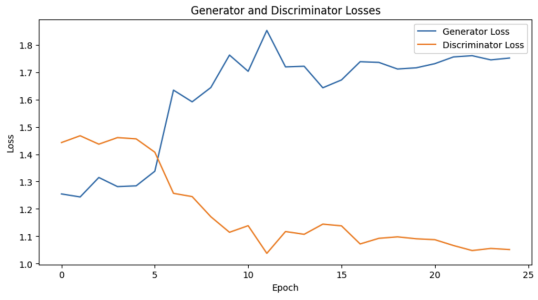

Fig. 2. Generator and discriminator losses during cGAN training.

D. Generated Images

The cGAN successfully generated realistic food images for all 101 classes in the dataset. Figure 3 shows examples of these synthetic images.

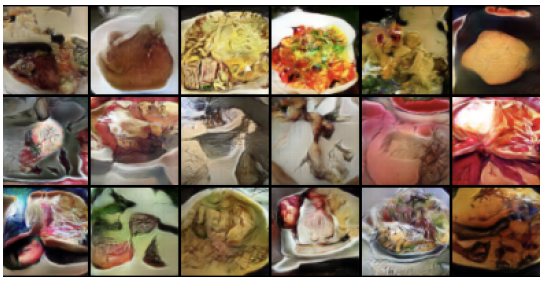

Fig. 3. Sample images generated by the cGAN for various food classes.

The inclusion of synthetic images in training increased diversity and robustness. This approach makes the model less prone to overfitting and more adaptable to unseen data.

V. DenseNet with Augmented Dataset

I used the pre-trained DenseNet121 model on a dataset that included synthetic images generated by my Conditional GAN (cGAN). The goal was to evaluate whether combining GAN-generated images with additional data transformations could improve the model's performance.

A. Dataset Preprocessing and Augmentation

1) Resize to 256 pixels: All images were resized.
2) Center crop to 224 × 224 pixels: This matched the input size required by the DenseNet121 architecture.
3) Random horizontal flip and rotation: Variability was introduced by flipping images horizontally and rotating them randomly by up to 10 degrees.
4) Normalization: Pixel values were standardized with a mean of $[0.485, 0.456, 0.406]$ and a standard deviation of $[0.229, 0.224, 0.225]$, consistent with the requirements for pre-trained models.

Additionally, the training set was augmented with synthetic images generated by the cGAN. Each class received 100 additional images, enhancing the overall diversity and size of the training dataset.

B. Model Architecture and Training

The same approach was used as in the previous experiment with the DenseNet121 model. The feature extraction layers were frozen to retain the general patterns learned from the ImageNet dataset, and the final fully connected layer was replaced with a classifier adapted to the 101 classes of the Food-101 dataset. The training configuration also remained consistent.

C. Results

- Top-1 accuracy: 55.62% on the test set.
- Top-5 accuracy: 79.66% on the test set.

Adding synthetic images generated by the cGAN increased the diversity of the training data. However, the performance was slightly lower than the DenseNet121 model trained without augmentation. This may be due to the synthetic images not fully capturing the complexity and variability of real-world data. Nevertheless, the results highlight the potential of cGANs for data augmentation, especially in scenarios where data is limited or imbalanced.

VI. Conclusion

In this project, I explored different deep learning techniques for food classification using the Food-101 dataset. The experiments included training a custom CNN, utilizing a pre-trained DenseNet121 model, and augmenting the dataset with synthetic images generated by a Conditional GAN.

- Custom CNN: The custom CNN, while capable of learning basic patterns, achieved lower accuracy compared to the pre-trained DenseNet121. This highlights the advantages of using transfer learning for complex datasets.
- DenseNet121 without Augmentation: The pre-trained model delivered the best performance, achieving a Top-1 test accuracy of 57.77% and a Top-5 accuracy of 82.06%. These results demonstrate the effectiveness of leveraging features learned on large-scale datasets like ImageNet.
- DenseNet121 with Augmentation: Adding synthetic images from the Conditional GAN slightly reduced performance, with a Top-1 test accuracy of 55.62% and a Top-5 accuracy of 79.66%. While synthetic images increased diversity, they might not fully replicate the complexity of real-world data.

This project highlighted the effectiveness of pre-trained models and explored the potential of GANs for data augmentation. While the results demonstrate promise, they also emphasize the challenges of optimizing synthetic data for practical applications. Future work should focus on improving data quality and leveraging advanced training techniques to further enhance classification performance.

References

[1] A. Krizhevsky et al., "ImageNet Classification with Deep Convolutional Neural Networks," 2012.
[2] T. Karras et al., "Progressive Growing of GANs," 2017.
[3] TensorFlow, *An End-to-End Open Source Machine Learning Platform*. Available: https://www.tensorflow.org/
[4] TensorFlow Datasets, *Ready-to-Use Datasets for Machine Learning*. Available: https://www.tensorflow.org/datasets
[5] PyTorch, *An Open Source Machine Learning Framework*. Available: https://pytorch.org/
[6] Food-101 Dataset, *A Dataset for Food Classification*. Available: https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/.
[7] I. Goodfellow et al., "Generative Adversarial Networks," 2014.