Jade Sanchez

1378 ITAI

12/2/2024

Reflective Journal

How does the choice of pre-trained model (VGG16, ResNet50, etc.) affect the results?

Impact of Architecture: It ranges from the size of a model itself, the architecture of the accessed layers, and the features they contain.
VGG16: Another one came up with a deeper but less complex one using a uniform architecture: (3x3 convolutions, max-pooling). It performs well in condition when spatial features (edges, corners) are most important in the dataset.
ResNet50: Introduces residual connections that leads to very deep networks without mentioning vanishing gradient problems. It performs well in cases where the features of the data set are somehow correlated or in a way, ordered in a hierarchical manner.
MobileNetV2: Fast and suitable for mobile/low computational conditions, this is appropriate when operating on small amount of data or when the resources of the computer are few.
Considerations: Where the datasets are complex and may present patterns that are different there is an addition of ResNet50.
Select MobileNetV2 for the case of faster training with fewer data or minimum data sets.
Choose VGG16 for fully automated tasks when simpler models are sufficient for the problem at hand since it's easiest to debug.

Analyze the confusion matrix: Are errors more common between certain classes? What might explain this?

Observations from the Confusion Matrix: Higher errors emerge when the classes are close in terms of visual resemblance such as CIFAR-10's "cat and dog" or "car and truck." Possible reasons for misclassification: Feature Similarity: The same patterns, colours, or shapes in different classes. Class Imbalance: Small sample size in some classes results in low generalization. Overfitting: The model recalls the training examples, but misses on discriminating within the differentiation in the test set. Steps to Analyze: Find out those misclassified patterns with the highest confusion in their respective classes. In addition to using accuracy as an evaluation metric transform some of the misclassified image samples in order to analyze the errors made by the model qualitatively.

Experiment with different degrees of fine-tuning (freezing more/fewer layers of the pre-trained model).

Effect of Fine-Tuning:
Freezing all layers: It's faster but the training is only limited to features learnt in the pre-training phase by the model. Excellent for related datasets seen in a similar stream of work.
Fine-tuning few layers: Changes the high-level characteristics to a newly generated dataset to provide better results for slightly altered datasets.
Fine-tuning most layers: Is more helpful when there is a shift in a dataset by a factor of two or more but needs more computer time and is more easily overfitting.
Best Practices:
The deeper layers in the convolutional model get strongly frozen; the deeper layers for refinement from the lowest layer by blocks that contain general features such as edges.
While training their subordinates, managers should apply an incremental type of unfreezing in order to better fit the environment.

If applicable to your dataset, can you collect more data for classes with higher error rates?

In the case that there are specific classes which tend to be labeled wrong often, it would be beneficial to gather more samples for such classes in order to increase the model accuracy, and achieve a better ratio of the classes.
Seek open data sets, people's invitations, or sample data generation for underrepresented samples.

What are other ways to potentially improve accuracy?

Use either shear_range, zoom_range or cutout augmentation.
Try GANs for generating synthetic data.
Class-Weighted Training: To focus on these classes during training assign higher weights to them in the loss function.
Ensemble Models: Engulfed predictions from two or more models (For instance, VGG16 and ResNet50) to have higher model stability and, to eliminate individual model biases.
Regularization Techniques: Remove dropout layers or increase the dropping rate, or use batch normalization, or early stopping to minimize the overfitting.
Hyperparameter Tuning: First of all, you will need to experiment with learning rates, batch sizes, and numbers of layers to get the best results.
Transfer Learning with Custom Layers: Replace the top layers by more complex structures as the dense layers, or the recurrent units for better of abstraction.

Experimentation and Iteration

Methodology: They fine-tuned the pre-trained model which was VGG16 and the chosen dataset was CIFAR-10. To reduce overfitting, data augmentation was practiced using ImageDataGenerator with improved performance. Moreover, other tasks, such as hyperparameter tuning optimal for model training like the learning rate, were performed.

Results: Baseline Model: VGG16 without transfer learning was successful to a limited extent because of freezing layers and the absence of augmentation. Fine-tuned Model with Augmentation: OBTAINED higher accuracy because of better feature learning, better generalization of the features. Learning Rates: From the experiment, it was observed that a learning rate of 0.001 yielded the best training convergence with reasonable and optimum training time.

Insights: Increasing the training data by a large amount led to massive improvement in the performance of the model. Specifically, unfreezing fewer layers in the pre-trained model gave better feature learning for the CIFAR 10 images. Validation loss was monitored to discover early on that overfitting occurred.

Key Details to Document:

Dataset Selection: CIFAR-10 was chosen because its number of classes is balanced, and its size allows for computation within reasonable time.

Data Preprocessing: Images were resized to a network input size and also normalized to a range of [0, 1] we further augmented the data with rotation flips and shifts.

Model Architecture: We also use transfer learning with VGG16 as the pretrained base model to which we add dense layers for CIFAR-10 classification.

Training Process: Regularity augmentation, checking points and choice of hypers were applied to optimize performance.

Metrics: The training accuracy, validation accuracy, test set loss and test set accuracy.

Risk Management Potential Risks and Mitigation Strategies:

Data Issues: This is usually caused by small size of the data set or else occurrences of imbalance between the two.

Mitigation: Data related to text generation should be augmented and the model should be pre-trained.

Computational Constraints: VGG16 is trained on a limited hardware.

Mitigation: Consider increasing the number of epochs, or try to train the model on less data at a time, or use non-trainable layers, or avail cloud services.

Overfitting: Model remembers training data, but does not perform well on new data, not seen during training.

Mitigation: Validation loss should be monitored and dropout layers used, as well as the early stopping method.

Reflection Learning Experience:

Team Members' Contributions: The most effective approach for problem solving was collaborative as exemplified by the efforts to test new data augmentations.

Individual Learning: Understood the idea of transfer learning and data gathering and preparation. Realised that in order to advance the results, systematic experimentation ought to be conducted.

Challenges: Task specific proper augmentations and tuning up its hyperparameters consumed much more time.

Achievements: Implemented a pipeline with the option of approaching transfer learning and the pipeline achieved good accuracy on CIFAR-10.

Innovation

Ideas Explored: For instance, applying state of the art enhancements such as zooming or shearing. So, the first and second strategies combined utilize two different types of pre-trained models, ResNet50 and MobileNetV2. Introducing a learning rate scheduler as a more effective form of optimization.

Project Report

Dataset Selection and Justification: CIFAR-10 was selected as it includes 10 balanced classes and is relatively small, and can be used for fast experimenting. This is suitable for judging the performance of a CNN because the images contain a realistic range of diversities.

Methodology:

Model: VGG16 will be used as the base model and hence transfer learning will be employed. To elaborate further, dense layers were incorporated towards the end.

Data Preprocessing: The images were normalized and augmented in order to mimic real world potential fluctuations.

Training: Real time data enhancement technique of type ‘batch’ was applied at batch size 32 while using a learning rate of 0.001.

Evaluation: The performance was evaluated using training/validation accuracy and confusion matrix for the tuning of the parameters, while test set evaluation was used in the final evaluation.

Works Cited (MLA Format)

Chollet, François. Deep Learning with Python. 2nd ed., Manning Publications, 2021.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press, 2016.

Karpathy, Andrej. “CS231n: Convolutional Neural Networks for Visual Recognition.” Stanford University, 2017, cs231n.stanford.edu.

Keras Documentation. “Keras: The Python Deep Learning API.” Keras.io, Keras, https://keras.io/.

"CIFAR-10 Dataset." Kaggle, https://www.kaggle.com/c/cifar-10.

TensorFlow Documentation. “VGG16 Model in Keras.” TensorFlow.org, TensorFlow, https://www.tensorflow.org/api_docs/python/tf/keras/applications/VGG16.

“Image Augmentation in Keras.” Keras Documentation, Keras, https://keras.io/api/preprocessing/image/.