Artificial Intelligence

TA-2 Assignment

Submitted By

Raj Shekhar

Enrolment no.: 240545002004

M.Sc. in Cyber Security

Semester I

Guided By

Mrs Meena Lakshmi

(Assistant Professor)

School of Cyber Security & Digital Forensics

National Forensic Science University

Bhopal Campus, M.P., 462001, India

Transfer Learning in Deep Learning

A technique where a model is pre-trained on one task is then fine-tuned for a new, related tasks, that can vary upon attributes, embeddings, and generative.

Attributes : These are specific characteristics or features of the data.

Example : An image of a dog has many attributes that includes the body shape and size, the colour patterns and fur patterns, etc, to identify its breed.

Embeddings : These are numerical representation of data, allowing complex items like words, images, audio, etc to be represented in vectors of fixed size that a model can process.

Example : Word embeddings in a text-message can be used for sentiment analysis or named entity recognition.

Generative : These are new data similar to training data, that can be fine-tuned to generate specific types of data.

Example : A GAN (Generative Adversarial Network) pre-trained on many different images can be then used to generate realistic arts or new creative contents.

There are two ways to customize a pretrained model :

- **Feature Extraction** : It uses the knowledge learned by the pre-trained model to extract meaningful features as samples for training a new model of a specific task.

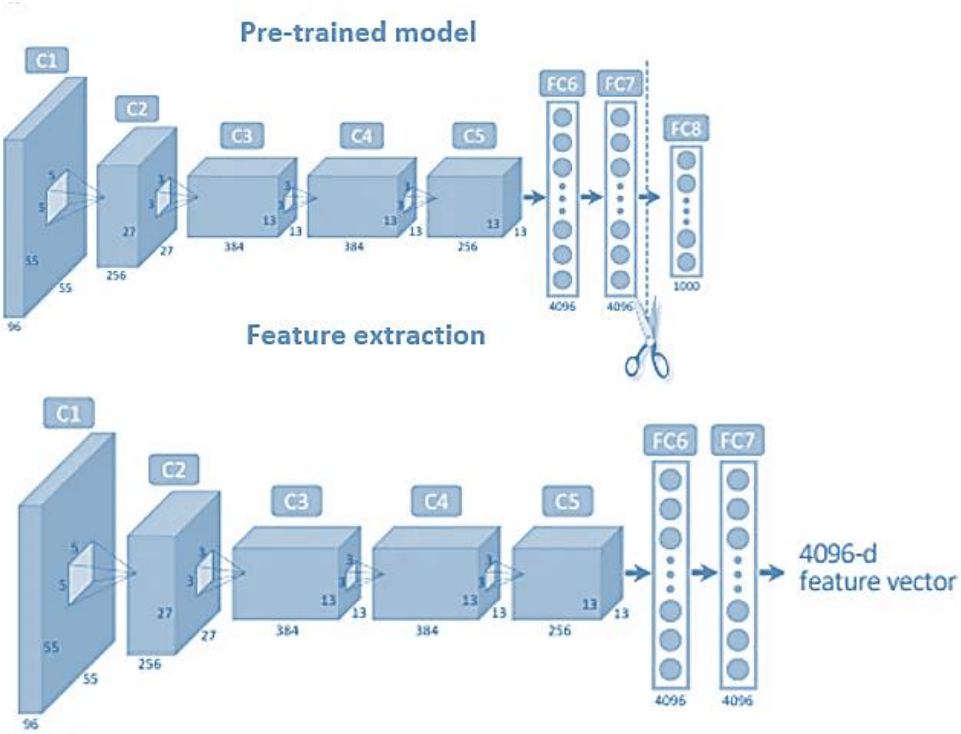

Fig.1: Feature Extraction of 4096-d feature vector from a pre-trained model

Example

```
# Train KNN (simulated pretrained model) for feature extraction
knn_model = KNeighborsClassifier( n_neighbors=3 )
knn_model.fit( X_train, y_train )

# Use the KNN model to extract class probabilities as features
X_train_features = knn_model.predict_proba( X_train )
X_test_features = knn_model.predict_proba( X_test )
y_train_features = some targets specific for probabilities

# Train a new classifier using the extracted features (Logistic Regression)
log_reg_model = LogisticRegression()
log_reg_model.fit( X_train_features, y_train_features )
```

- **Fine-Tuning** : It involves unfreezing some layers of a frozen pre-trained base model and jointly training both frozen layer and the newly added classifier layers as the last layers on the base model.

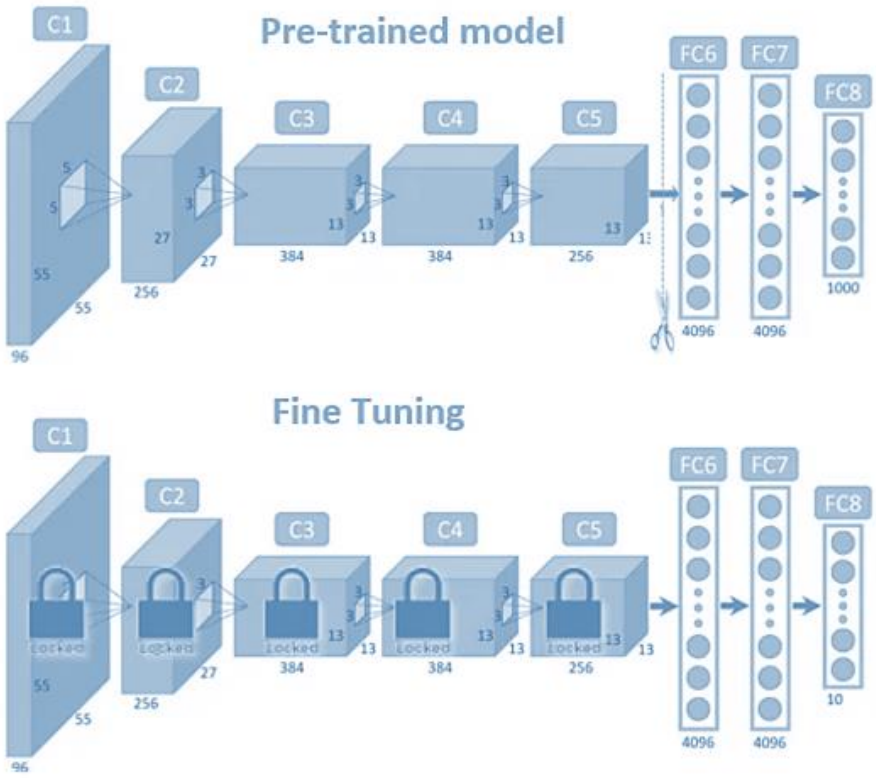

Fig.2: Fine Tuning with frozen layers of pre-trained model and newly added classifier layers

Example

```
# Train Keras model on extracted features from KNN
model = models.Sequential([
layers.InputLayer( input_shape = ( X_train_features.shape[ 1 ]  )),
layers.Dense( 64, activation='relu' ),
layers.Dense( 1, activation = 'sigmoid' ) ])

model.compile( optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'] )

model.fit( X_train_features, y_train, epochs = 10, batch_size = 32, validation_split = 0.2 )
```

Transfer learning is useful because it saves time and resources, improves performance, and helps in scenarios where data is limited.

Advantages

- Computational costs : reduce the amount of model training time, training data, processor units, and other computational resources.

- Dataset size : helps prevent difficulties that are involved in acquiring large datasets.

- Generalizability : can help to prevent overfitting, as the model will have already learned general features.

Disadvantages

- Domain mismatch : model can be applied to both learning tasks only if tasks are similar, where source and target datasets data distributions do not vary too greatly.

- Overfitting : model may learn task-specific features that do not generalize well to new data, if fine-tuned too much on the second task.

- Complexity : model fine-tuning process can be computationally expensive and may require specialized hardware.

Types of Transfer learning

1. Inductive Transfer : When the source and target tasks are different, regardless of any difference or similarity between the target and source domains, the source model is used to improve the performance of the target model and the data is often labeled.

2. Transductive Transfer : When the source and target tasks are the same, but the datasets (or domains) are different. An example is adapting a language model

trained on one language (e.g., Hindi) to work on different languages (e.g., Odia, Malyalam).

3. Unsupervised Learning : There is no labeled data available in neither the source nor the target domain. The target and source tasks are different. One common application of unsupervised learning is fraud detection.

Applications of Transfer Learning

- Computer Vision : like image recognition tasks, where models pre-trained on large image datasets are adapted to specific tasks such as **medical imaging**, facial recognition, and **object detection**.

- Natural Language Processing (NLP) : In NLP, models like BERT, GPT, are pre-trained on vast text-corpus and for specific tasks such as **sentiment analysis**, **machine translation**, and **question-answering**.

- Finance : assists in **fraud detection**, **risk assessment**, and **credit scoring** by transferring patterns learned from related financial datasets.

Conclusion

Transfer learning is a powerful technique in deep learning that enhances model performance by leveraging knowledge from previously trained models. By starting with pre-existing models and fine-tuning them for specific tasks, transfer learning saves time, improves accuracy, and enables effective learning even with limited data.