Deep Learning: A Comprehensive Guide

1. Introduction to Deep Learning

Deep Learning is a subset of Machine Learning that uses neural networks with multiple layers to learn complex patterns from data.
It is widely used in image recognition, natural language processing, and autonomous systems.

Key Characteristics:
- Uses artificial neural networks (ANNs)
- Requires a large amount of data
- Benefits from high computational power (GPUs, TPUs)
- Improves performance with deeper architectures

Example Python Code:

```
import tensorflow as tf
from tensorflow import keras

model = keras.Sequential([
    keras.layers.Dense(128, activation='relu', input_shape=(784,)),
    keras.layers.Dense(10, activation='softmax')
])

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

2. Neural Network Structure & Logic

A neural network consists of layers of interconnected neurons.

Layers in a Neural Network:
1. Input Layer: Receives input data
2. Hidden Layers: Processes and extracts features
3. Output Layer: Produces final predictions

Activation Functions:
- ReLU (Rectified Linear Unit): Used in hidden layers for non-linearity

- Sigmoid: Used for binary classification
- Softmax: Used for multi-class classification

Example of Forward Propagation:

```
def forward_propagation(X, W, b):
    return tf.nn.relu(tf.matmul(X, W) + b)
```

3. Convolutional Neural Networks (CNNs)

CNNs are deep learning models designed for image processing.

CNN Components:
- Convolutional Layer: Extracts features using filters
- Pooling Layer: Reduces dimensionality (Max/Average Pooling)
- Fully Connected Layer: Performs final classification

CNN Example Code:

```
model = keras.Sequential([
                    keras.layers.Conv2D(32, (3,3), activation='relu', input_shape=(28,28,1)),
    keras.layers.MaxPooling2D(2,2),
    keras.layers.Flatten(),
    keras.layers.Dense(128, activation='relu'),
    keras.layers.Dense(10, activation='softmax')
])
```

4. Recurrent Neural Networks (RNNs), LSTMs, & GRUs

RNNs process sequential data, making them useful for time-series and NLP tasks.

Key Differences:
- RNN: Maintains memory using hidden states
- LSTM (Long Short-Term Memory): Overcomes vanishing gradients using gates
- GRU (Gated Recurrent Unit): A simplified LSTM with fewer parameters

LSTM Example:

```
model = keras.Sequential([
```

```
    keras.layers.LSTM(50, return_sequences=True, input_shape=(100,1)),
    keras.layers.LSTM(50),
    keras.layers.Dense(1)
])
```

5. Transformers & Modern DL Models

Transformers are the foundation of modern NLP models like GPT and BERT.

Transformer Architecture:

- **Self-Attention Mechanism: Focuses on relevant input parts**
- **Positional Encoding: Maintains order information**
- **Multi-Head Attention: Enhances feature extraction**

Transformer Example:

```
from transformers import BertTokenizer, TFBertModel

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = TFBertModel.from_pretrained('bert-base-uncased')
```