Deep Learning Guide: Transformers, LSTM, and GRU

# 1. Transformers (Self-Attention Mechanism)

Transformers revolutionized NLP with self-attention. Introduced in “Attention is All You Need” by Vaswani et al. in 2017.  
  
Core Concepts:  
- No RNN/CNN: Works via attention mechanisms alone.  
- Scales well with large data.  
- Used in BERT, GPT, T5, etc.  
  
Key Components:  
1. Input Embeddings + Positional Encoding  
2. Multi-head Self-Attention  
3. Feed Forward Network (FFN)  
4. Add & Norm  
5. Encoder/Decoder Blocks  
  
Self-Attention Explained:  
- Input: Sequence of embeddings (Q, K, V)  
- Attention(Q, K, V) = softmax(QKᵀ / √dₖ) \* V  
  
Sample PyTorch Code using Hugging Face:  
from transformers import AutoTokenizer, AutoModel  
tokenizer = AutoTokenizer.from\_pretrained("bert-base-uncased")  
model = AutoModel.from\_pretrained("bert-base-uncased")  
inputs = tokenizer("Hello, Transformers!", return\_tensors="pt")  
outputs = model(\*\*inputs)  
  
Use Cases:  
- Text Classification, Question Answering, Translation, etc.

# 2. LSTM (Long Short-Term Memory)

LSTM is a type of RNN designed to capture long-term dependencies.  
  
Why LSTM?  
- Addresses vanishing gradient problem in RNNs  
- Maintains memory via gates (Forget, Input, Output)  
  
Cell Structure:  
1. Forget Gate: Decides what to throw away  
2. Input Gate: Decides what new info to add  
3. Output Gate: Decides what to output  
  
PyTorch Example:  
import torch  
import torch.nn as nn  
  
lstm = nn.LSTM(input\_size=10, hidden\_size=20, num\_layers=2)  
inputs = torch.randn(5, 3, 10) # (seq\_len, batch, input\_size)  
h0 = torch.randn(2, 3, 20)  
c0 = torch.randn(2, 3, 20)  
out, (hn, cn) = lstm(inputs, (h0, c0))  
  
Use Cases:  
- Text Generation, Sentiment Analysis, Time Series Forecasting

# 3. GRU (Gated Recurrent Unit)

GRU is a simplified version of LSTM, introduced in 2014.  
  
Advantages:  
- Fewer gates (only update and reset)  
- Faster to train, less complex  
  
GRU Mechanism:  
1. Update Gate: How much of the past to keep  
2. Reset Gate: How to combine new input with previous memory  
  
PyTorch Example:  
gru = nn.GRU(input\_size=10, hidden\_size=20, num\_layers=1)  
inputs = torch.randn(5, 3, 10)  
h0 = torch.randn(1, 3, 20)  
out, hn = gru(inputs, h0)  
  
Use Cases:  
- Similar to LSTM: good for speech, time series, NLP tasks  
  
Tip:  
LSTM and GRU perform similarly; GRU is preferred when faster training is needed.