Architecture Choice and Why

Deciding factors included:

1. **TCN vs RNN/LSTM:** TCN processes data in parallel, making training faster. Dilated convolutions prevent gradient vanishing issues that affect RNNs.
2. **TCN vs Transformer:** TCN works better for sensor data because it assumes nearby timesteps are related (local patterns). With only 108K parameters vs 300K+ for Transformers.
3. **Dilated Convolutions:** Using rates [1,2,4,8] lets the model see patterns at different time scales. Early layers detect quick movements like individual steps, while deeper layers recognize longer patterns like complete walking cycles. Residual connections help train the 4-layer network, avoiding gradient problems.

Preprocessing Strategy

- **Normalization:** Applied z-score normalization $(x - mean) / std$ separately for each channel. This is necessary because accelerometer readings (m/s^2) and gyroscope readings (rad/s) have different units and scales. Without it, larger values would dominate the learning process.
- **Input Projection:** A 1×1 convolution transforms 9 input channels to 64 features. This lets the model learn useful combinations of different sensors (like how acceleration relates to gyroscope readings) before processing time patterns. It uses fewer parameters than fully-connected layers.
- **30% Validation Split:** Held out 2,164 samples (30% of training data) to create a larger validation set for reliable overfitting detection. Better signal for early stopping decisions.

Challenges Encountered

- **Receptive Field Sizing:** Initial [1,2,4] dilation gave only a small receptive field. Expanded to [1,2,4,8] for 31 timesteps. Further expansion [1,2,4,8,16] caused overfitting without gains.
- **Overfitting:** Model reached ~97% training accuracy but only 89% validation accuracy, showing overfitting. Implemented early stopping when the difference becomes too large.

Trade-offs Made

- **Model Depth (4 vs 8):** 4 blocks cover ~50% sequence (sufficient for patterns) vs 8 blocks cover 0100% but double training time and risk of overfitting. accuracy saturates within 4 blocks.
- **Expansion [64 ,64, 128,128] vs [128 ,128 , 128,128]:** Progressive expansion uses 108K params vs 185K. Faster training and better generalization.
- **Kernel Size (3 vs 5):** k=3 has fewer parameters without any major consequence.
- **Global Avg Pool vs Flatten+FC:** Pooling uses 128 params vs ~1M. small accuracy difference.
- **Optimizer (Adam vs SGD):** Adam converges in 30 epochs vs 60+ for SGD. Time constraints made Adam essential despite SGD's potentially better final loss.

Potential Improvements

- **Architecture:** Multi-scale TCN with parallel branches (kernel sizes [3,5,7]). Squeeze-Excitation blocks for channel attention. lightweight self-attention after final TCN for long-range dependencies.
- **Adaptive Learning Rate:** to improve the gradient results further
- **Batch Size Tuning:** Try different configurations to find a decent batch for training, validating, and testing
- **Extended Work:** Transfer learning (pre-train on PAMAP2/OPPORTUNITY). Multi-task learning (activity + intensity). Ensemble approach (multiple models).

References

1. https://arxiv.org/pdf/2112.09293
2. Z-Score Normalization: Definition and Examples - GeeksforGeeks