1 Mixed Isometric Convolution (MIC) for Time Series Forecasting and Imputation

1.1 Introduction

Time series data exhibits intricate patterns at various temporal scales, from local fluctuations to long-term trends. Our model leverages Mixed Isometric Convolution (MIC) layers to capture these diverse dynamics effectively.

1.2 Model Architecture

Figure 1: Overview of the MIC-based time series model architecture.

Our model comprises:

1. **MIC Layers:** * Combine isometric, downsampling, and upsampling convolutions to extract multi-scale features. * Utilize residual connections and normalization for improved training stability.

Listing 1: Core of the MIC layer: 'conv$_t$rans$_c$onv' $function$

```
1  def conv_trans_conv_ours(self, input, conv1d, conv1d_trans,
       isometric):
```

```
2        batch_size, seq_len, channels = input.shape
3        x = input.permute(0, 2, 1)
4
5        # Downsampling convolution with residual connection
6        x_r = conv1d(x)
7        x_r = self.act(x_r)
8        x_r = self.drop(x_r)
9        x_iso = x_r
10       # ... (rest of the code)
```

2. **SeasonalPrediction Module:** * Stacks multiple MIC layers for deeper representation learning. * Projects the learned features onto the output space.

Listing 2: Seasonal prediction module definition

```
1   class SeasonalPrediction(nn.Module):
2       def __init__(self, embedding_size=512, n_heads=8, dropout=0.05,
            d_layers=1, decomp_kernel=[32], c_out=1,
3               conv_kernel=[2, 4], isometric_kernel=[18, 6],
                    device='cuda'):
4           super(SeasonalPrediction, self).__init__()
5
6           self.mic = nn.ModuleList([MIC(feature_size=embedding_size,
                n_heads=n_heads,
7                                   decomp_kernel=decomp_kernel,
                                        conv_kernel=conv_kernel,
8                                   isometric_kernel=
                                        isometric_kernel, device=
                                        device)
9                               for i in range(d_layers)])
10
11          # ... (rest of the code)
```

3. **Model Class (Forecasting/Imputation):** * Handles both long-term forecasting and time series imputation tasks. * Employs time series decomposition for forecasting. * Combines trend and seasonal predictions for a comprehensive output.

1.3   Key Features