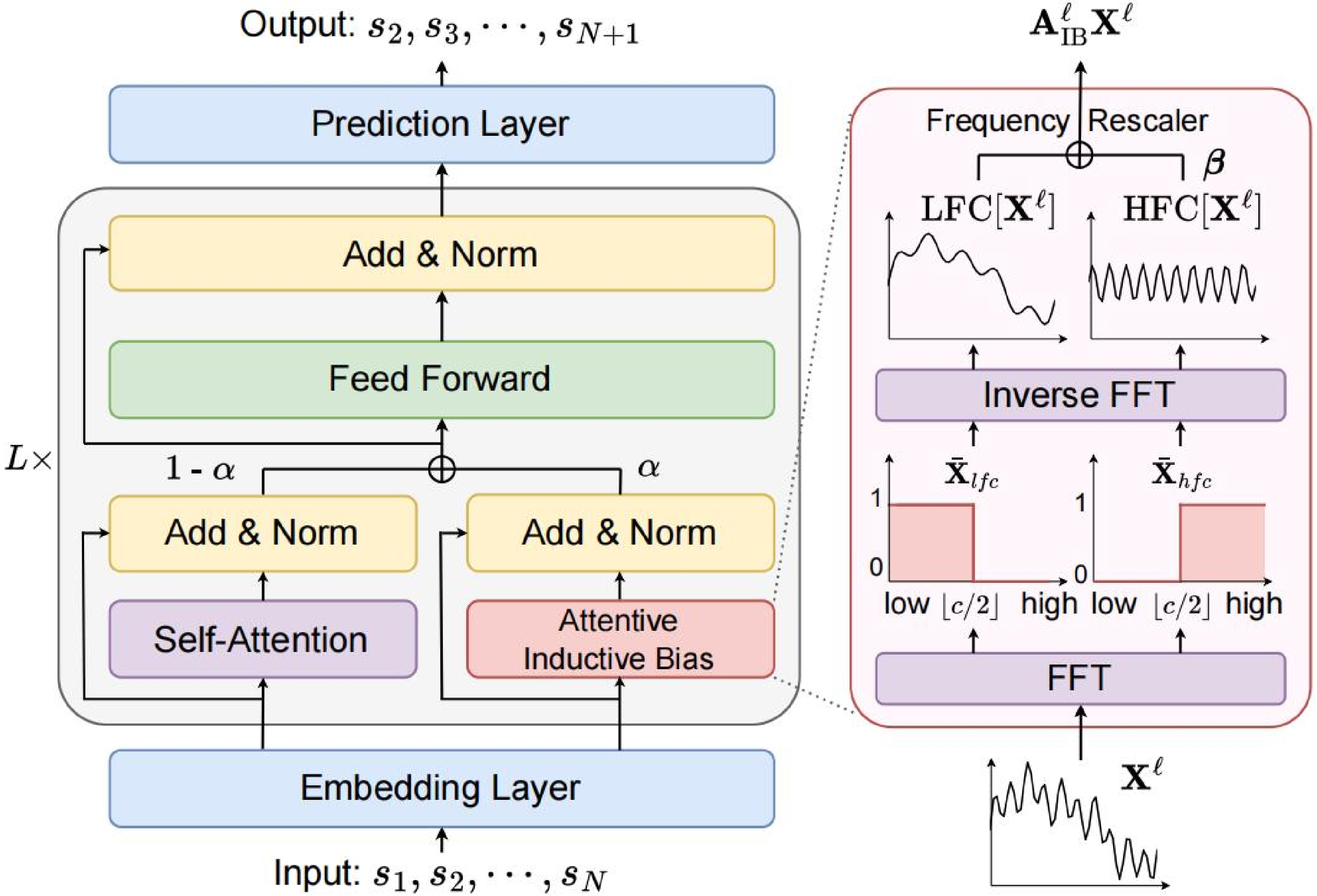
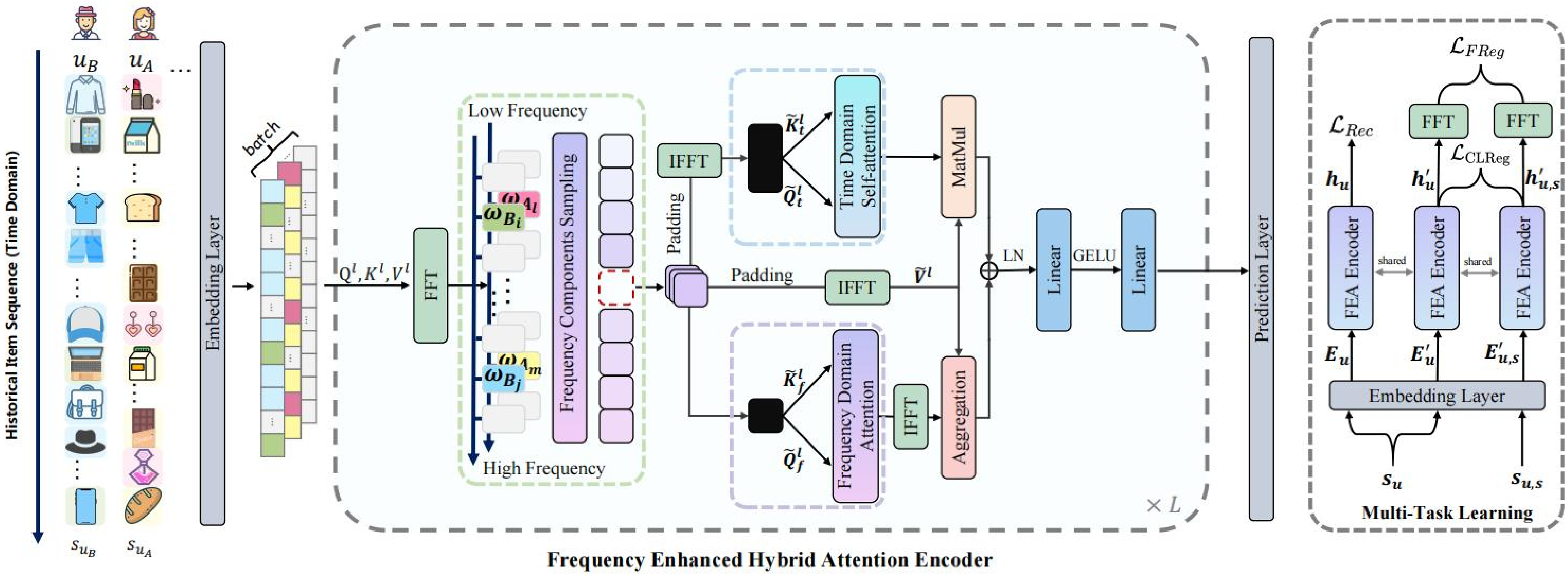
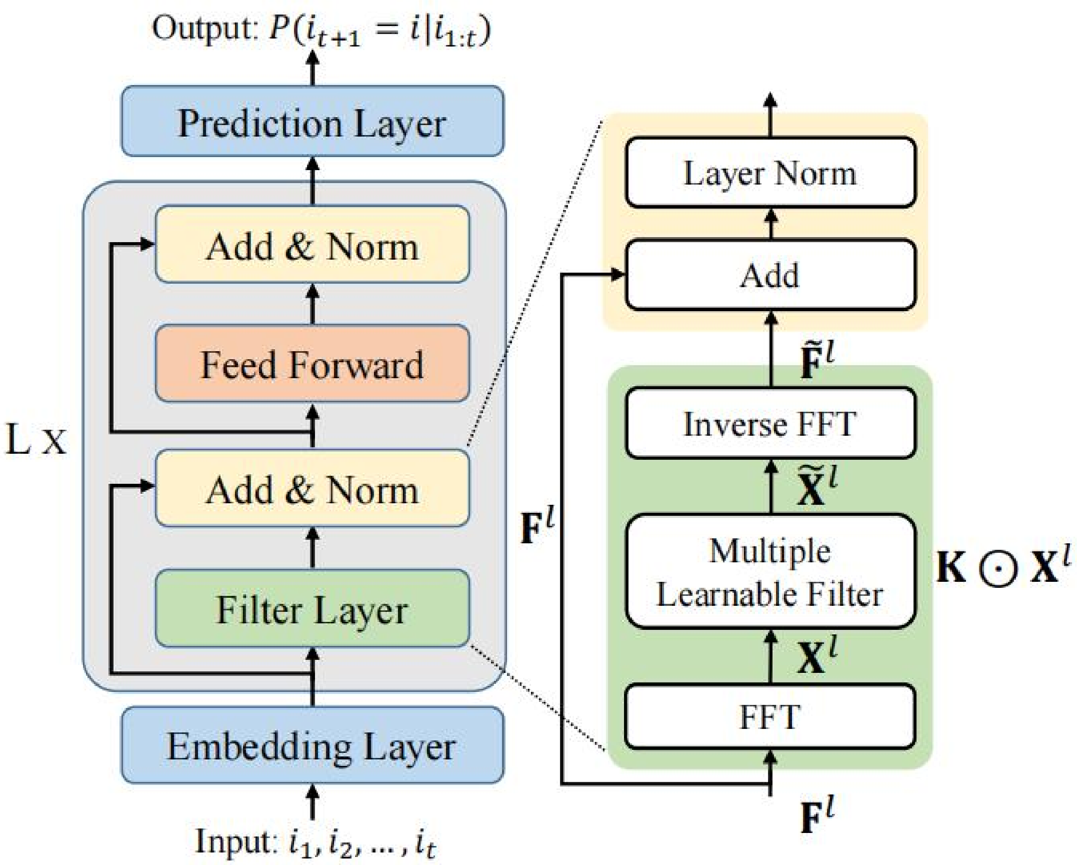


2024_AAAI_An Attentive Inductive Bias for Sequential Recommendation beyond the Self-Attention

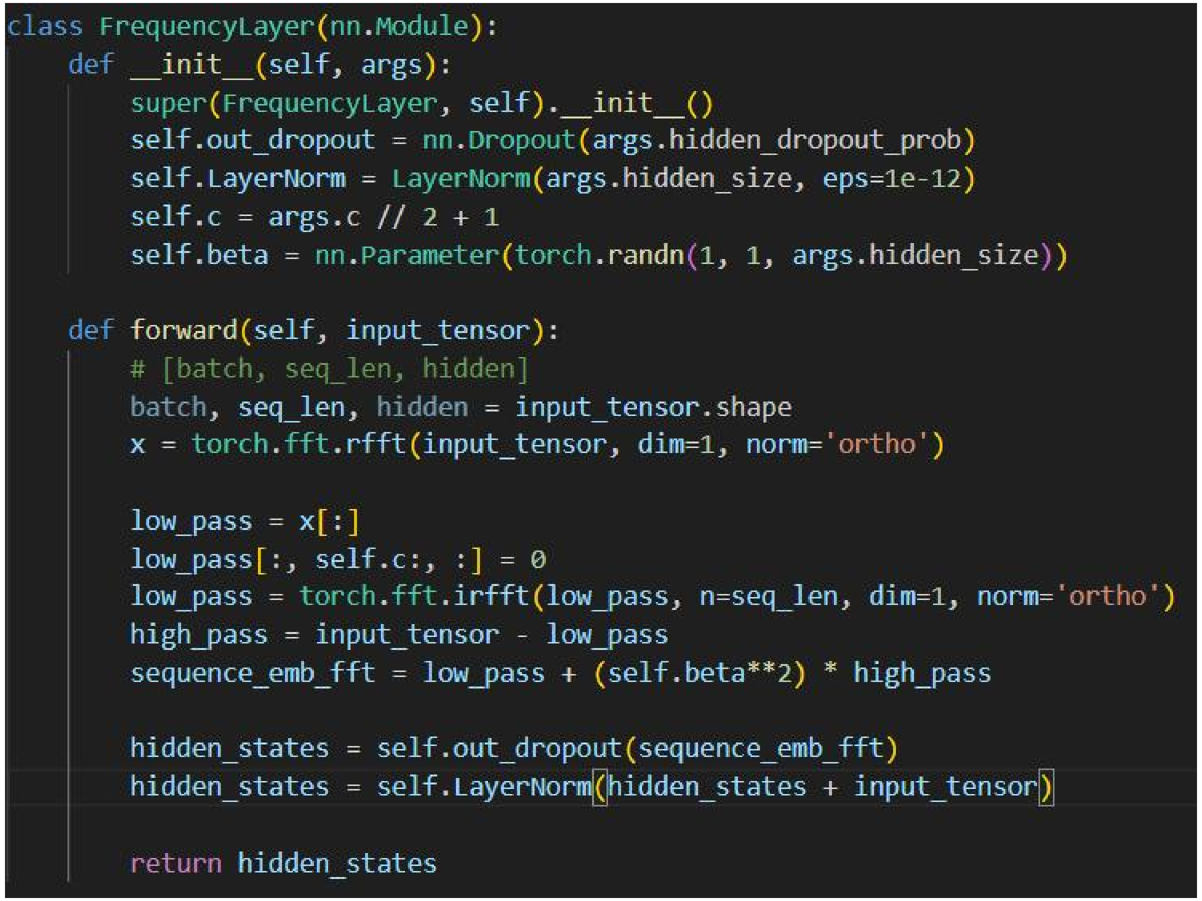


2023_Sigir_Frequency Enhanced Hybrid Attention Network for Sequential Recommendation



2022_WWW_Filter-enhanced MLP is All You Need for Sequential Recommendation

# 推荐系统——傅里叶变换代码
# （AAAI_2024）



```python
class FrequencyLayer(nn.Module):
    def __init__(self, args):
        super(FrequencyLayer, self).__init__()
        self.out_dropout = nn.Dropout(args.hidden_dropout_prob)
        self.LayerNorm = LayerNorm(args.hidden_size, eps=1e-12)
        self.c = args.c // 2 + 1
        self.beta = nn.Parameter(torch.randn(1, 1, args.hidden_size))

    def forward(self, input_tensor):
        # [batch, seq_len, hidden]
        batch, seq_len, hidden = input_tensor.shape
        x = torch.fft.rfft(input_tensor, dim=1, norm='ortho')

        low_pass = x[:]
        low_pass[:, self.c:, :] = 0
        low_pass = torch.fft.irfft(low_pass, n=seq_len, dim=1, norm='ortho')
        high_pass = input_tensor - low_pass
        sequence_emb_fft = low_pass + (self.beta**2) * high_pass

        hidden_states = self.out_dropout(sequence_emb_fft)
        hidden_states = self.LayerNorm(hidden_states + input_tensor)

        return hidden_states
```

输入:[batch_size,seq_len,hiden_size]
输出:[batch_size,seq_len,hiden_size]