TIME SERIES GENERATION WITH MASKED AUTOENCODER

Mengyue Zha; SiuTim Wong;Mengqi Liu;Tong Zhang; Kani Chen

摘要

本文研究表明，带外推器的掩码自编码器 (ExtraMAE) 是一种可扩展的自监督时间序列生成模型。ExtraMAE 随机地掩盖原始时间序列的一些片段，并通过恢复被掩盖的片段来学习时间动态。本文的方法有两个核心设计。第一，外部自我监督。允许 ExtraMAE 有效且高效地捕获原始时间序列的时间动态。其次，ExtraMAE 提出了一个外推器来解开解码器的两个工作：恢复潜在的表示并将其映射回特征空间。这些独特的模型设计使得 ExtraMAE 在时间序列生成中持续且显著地优于最先进的基准 (SoTA)。轻量级的架构也使得 ExtraMAE 具有快速和可扩展的特点。ExtraMAE 在时间序列分类、预测和插补等各种任务中表现出优异的性能。作为一种自监督的生成模型，ExtraMAE 允许对合成数据进行显式管理。我们希望本文将迎来一个新的具有自监督模型的时间序列生成时代。

关键词： ExtraMAE；时间序列；自监督生成模型

1 引言

自引入深度学习以来，时间序列模型在模型容量和能力上都有了飞速的增长。最新的模型 [1–4] 往往有数百万个可训练的参数。模型规模的爆炸式增长依赖于高质量数据的充足供应。然而，许多领域并没有提供足够的合格数据。数据的稀缺性导致研究其背后的机制几乎是不可能的。在深度学习时代，对数据的渴求已经成为时间序列模型的瓶颈。一个打破数据限制的想法是合成数据，良好的合成数据尊重原始数据的时间动态性，在实际应用中可以作为原始数据的替代。

2 相关工作

2.1 序列数据重构方法

无监督的生成对抗网络 (GAN) [5] 可以用来生成合成数据。GAN 学习一个从随机噪声到原始数据分布的映射。理论上，GAN 可以从随机噪声中采样生成任意数量的样本。一系列的工作集中在 GAN 用于时间序列生成。C-RNN-GAN [6] 是 GAN 生成合成时间序列的，首次尝试用 GAN 生成合成时间序列，C-RNN-GAN 通过循环神经实现生成器和鉴别器。RCGAN [7] 在 C-RNN-GAN 基础上增加附加条件信息，生成器和鉴别器都以辅助信息为条件。目前大量 GAN 变体生成特定领域的合成数据，如金融、传感器和决策等。序列数据的时间动态是数据

生成的核心，而上述 GAN 的变体模型都没有考虑时间序列数据的时间性质。为了更好地捕获时间序列数据中的时间依赖性，TimeGAN [8] 在潜在空间中设计了一个监督预测任务。目前已知 TimeGAN 是唯一一个成功保存原始数据时间动态的时间序列生成模型。尽管有监督训练在 TimeGAN 中取得了成功，但无监督模型仍然主导着时间序列生成。

2.2 无监督/有监督时间序列生成模型的区别

1. 训练难度不同。有监督生成模型很少受到无监督生成模型中运行问题的困扰。一些典型的问题是不收敛，梯度消失和模式崩溃。自然语言处理 (NLP) 和计算机视觉 (CV) 通过自监督预训练来解决这些问题。基于掩盖 [9–11] 的解决方案在概念上很简单：移除一部分数据，并学习对移除的内容进行填充。

2. 可操作性不同。无监督生成模型与无关的操作不兼容，其合成数据对下游任务是盲目的，不能管理合成数据的多样性，也无法估算缺失的价值；GAN 将随机噪声作为生成的种子，无法系统地管理生成。然而，监督生成模型允许我们对生成进行很好的控制，可以直接确定合成数据的多样性。

3 本文方法

3.1 本文方法概述

本文提出了带外推器的掩膜自动编码器 (ExtraMAE)，一种简单、有效、可扩展的时间序列生成监督模型，如图 1所示。训练是自监督的。在每次迭代中，ExtraMAE 随机掩盖输入时间序列中的部分片段，并从未掩盖的片段中外推缺失的片段。这种设计使得时间依赖关系的建模变得高效和有效。ExtraMAE 提出了一个外推器来重建潜在空间中缺失的块。外推器舍弃了掩码标记，从而避免了合成时间序列的不连续性。本文还对之前基于掩码的模型中的堆叠 Transformer 块进行了剪枝。用轻量级的 RNNs 替换了重量级的 Transformer 模块。ExtraMAE 的生成遵循类似交叉验证的方式。

编码器（Encoder）：由两层 RNN 网络构成，只对未遮盖的部分进行操作。

外推器（Extrapolator）：由两层全连接网络构成，恢复被遮盖部分的数据。

解码器（Decoder）：结构同 Encoder 部分一样，对经过外推器恢复的数据进行解码。

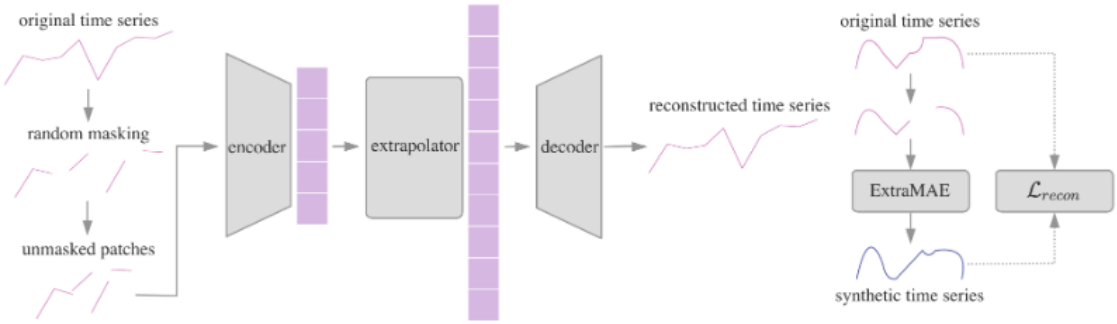


图 1. ExtraMAE 模型结构图

3.2 序列合成模块

ExtraMAE 以交叉验证的方式生成合成数据，如图 2所示。将一个原始的时间序列分成若干个折叠。每次 ExtraMAE 移除一个折叠，并从部分观察中重建。重复这个过程，直到每个折叠都有一个合成序列。我们将合成序列依次串联，进而得到完整的合成时间序列。

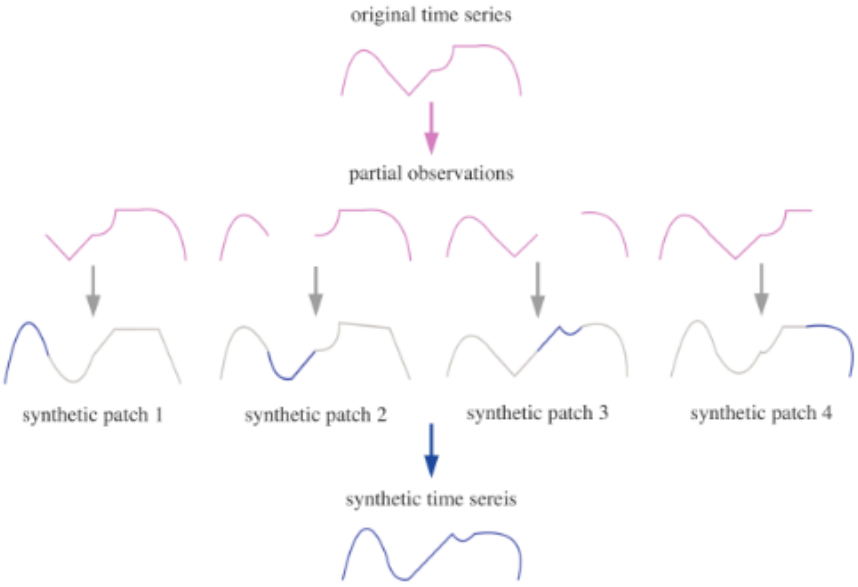


图 2. ExtraMAE 生成方案

3.3 损失函数定义

损失函数计算重构的时间序列与原始时间序列的均方误差 (MSE)，不同于只在掩膜的面片上计算损失，本文在所有的面片上计算损失，整体重构损失 L_{recon} 保证重构后的时间序列保持原时间序列 X 的连续性。损失函数公式如下:

$$L_{\text{recon}} = E_X \parallel X - \hat{X} \parallel^2$$

4 复现细节

4.1 与已有开源代码对比

本文可获取公开源码，本次复现模型结构采用论文提供的源码，其余部分新增了部分代码。由于代码项目代码过多，这里仅展示关键部分代码，如图 3 4 5所示。

本次复现，模型框架基于 Pytorch 搭建，Encoder 部分由两层 RNN 网络构成，只对未遮盖的部分进行操作。Extrapolator 部分由两层全连接网络构成，恢复被遮盖部分的数据。Decoder 部分结构同 Encoder 部分一样，对经过外推器恢复的数据进行解码。

```python
class Encoder(nn.Module):
    def __init__(self, args):
        super(Encoder, self).__init__()
        self.rnn = nn.RNN(input_size=args.z_dim,
                          hidden_size=args.hidden_dim,
                          num_layers=args.num_layer)
        self.fc = nn.Linear(in_features=args.hidden_dim,
                            out_features=args.hidden_dim)

    def forward(self, x):
        x_enc, _ = self.rnn(x)
        x_enc = self.fc(x_enc)
        return x_enc
```

图 3. 编码器代码

```python
class Decoder(nn.Module):
    def __init__(self, args):
        super(Decoder, self).__init__()
        self.rnn = nn.RNN(input_size=args.hidden_dim,
                          hidden_size=args.hidden_dim,
                          num_layers=args.num_layer)
        self.fc = nn.Linear(in_features=args.hidden_dim,
                            out_features=args.z_dim)

    def forward(self, x_enc):
        x_dec, _ = self.rnn(x_enc)
        x_dec = self.fc(x_dec)
        return x_dec
```

图 4. 解码器代码

```python
class Interpolator(nn.Module):
    def __init__(self, args):
        super(Interpolator, self).__init__()
        self.sequence_inter = nn.Linear(in_features=(args.ts_size - args.total_mask_size),
                                        out_features=args.ts_size)
        self.feature_inter = nn.Linear(in_features=args.hidden_dim,
                                       out_features=args.hidden_dim)

    def forward(self, x):
        x = rearrange(x, pattern: 'b l f -> b f l')
        x = self.sequence_inter(x)
        x = rearrange(x, pattern: 'b f l -> b l f')
        x = self.feature_inter(x)
        return x
```

图 5. 外推器代码

此外，主要添加 Argo 数据处理部分代码，下面展示获取拼接数据部分代码。

```python
def merge_data(sdata, simples_length):
    datas = []
    j, l, flag = 0, 0, 1
    for s in range(10):
        data = []
        simple_length = simples_length[s]
        if s == 0:
            for j in range(sdata[s].shape[0]):
                data.append(sdata[s][j])
            j = j + 1
        dl = len(data)
        if flag == 0:
            j = j + 24
        while dl < simple_length:
            for k in range(l, sdata[j].shape[0]):
                data.append(sdata[j][k])
                dl += 1
                if dl == simple_length:
                    if k < sdata[j].shape[0] - 1:
                        l = k + 1
                    else:
                        l = 0
                        flag = 0
                    break
            if dl < simple_length:
                l = 0
                j = j + 24
        data = np.array(data)
        datas.append(data)
    return datas
```

图 6. 数据拼接代码

4.2 实验环境搭建

本次复现在 Pytorch 中实现，使用 conda 创建代码所需的虚拟环境，在该虚拟环境下运行代码所需要的包：

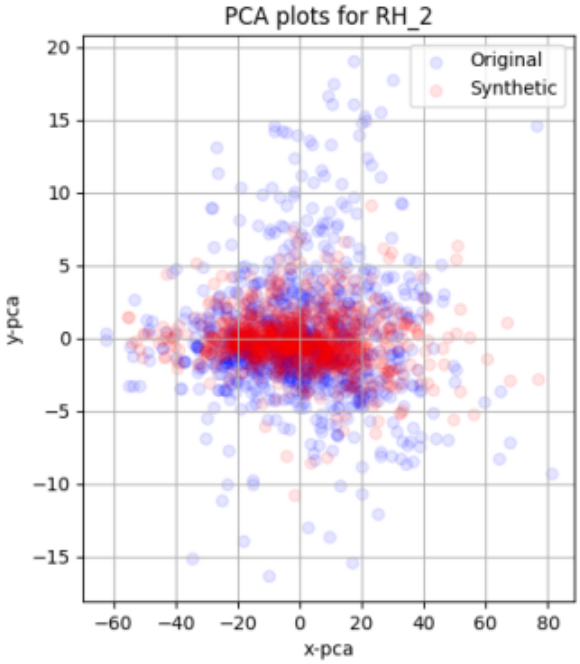
tqdm =4.62.3
numpy =1.19.5
tensorflow =2.0.0
scikit-learn =0.24.2
argparse =1.4.0
pandas =1.1.5
torch =1.10.0
matplotlib =3.3.4
einops =0.3.2
tf-slim
jupyterlab

4.3 创新点

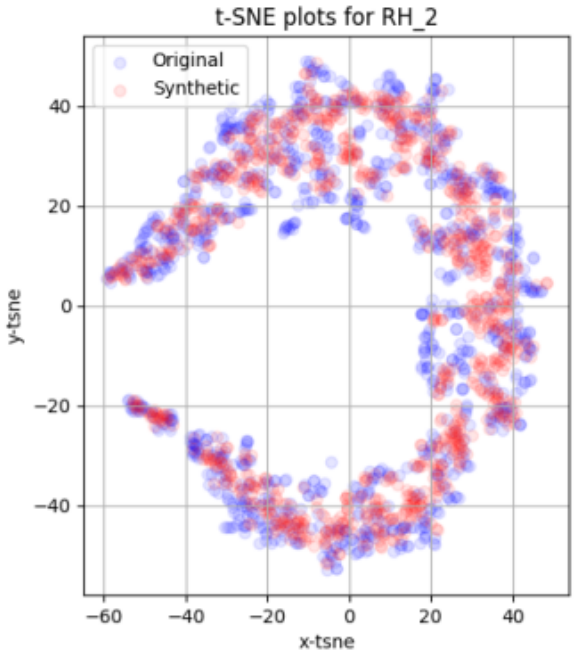
本文方法提出两个核心设计。第一，外部自我监督。监督允许 ExtraMAE 有效且高效地捕获原始时间序列的时间动态。其次，ExtraMAE 提出了一个外推器来解开解码器的两个工作：恢复潜在的表示并将其映射回特征空间。

5 实验结果分析

实验部分，本次复现将原始数据和合成数据进行可视化分析，以直观地评估保真度。使用 tsne 和 PCA 两种方法来显示合成数据的分布与原始数据的分布相似程度。对原论文中提供的股票、能源和正弦三个数据集进行实验，参数选用的是论文中最佳的效果参数，并与原论文中的结果进行比对。三种数据降维后的可视化效果图及测试分数如下：

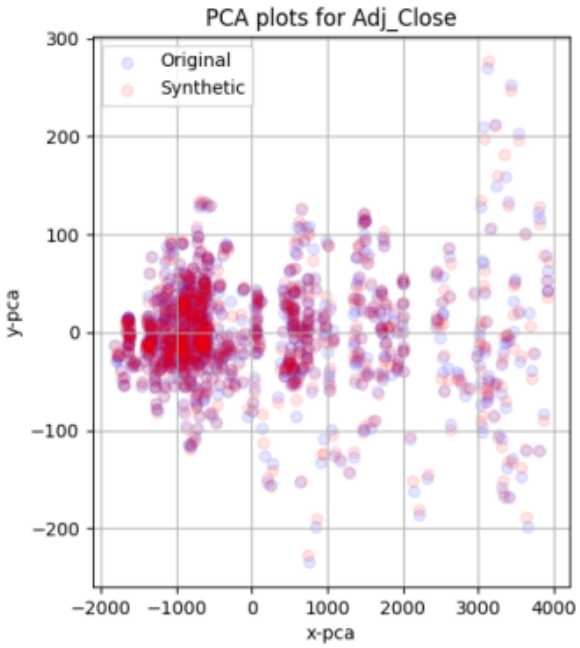


energy_PCA

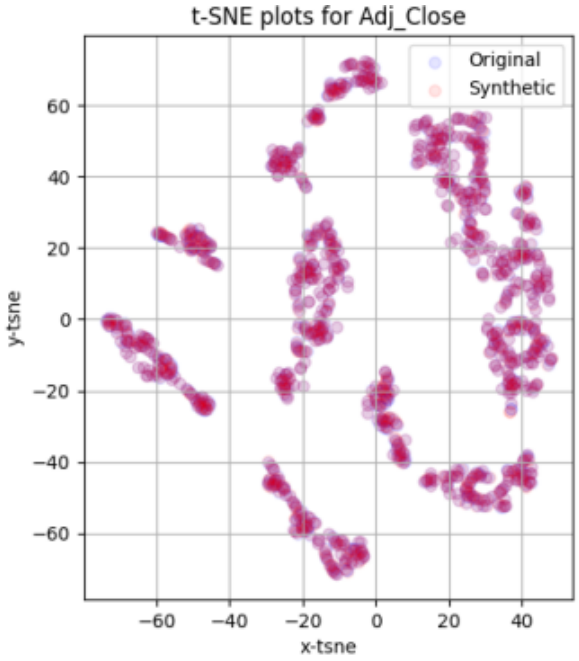


energy_tsne

图 7. 能源数据降维可视化效果

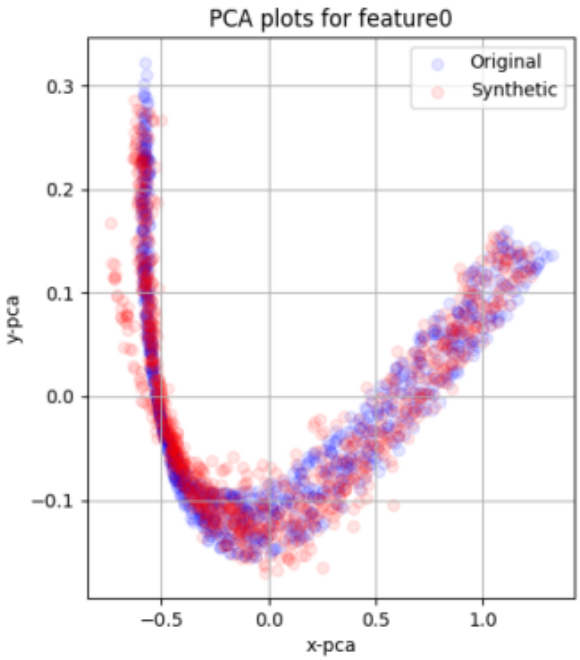


stock_PCA

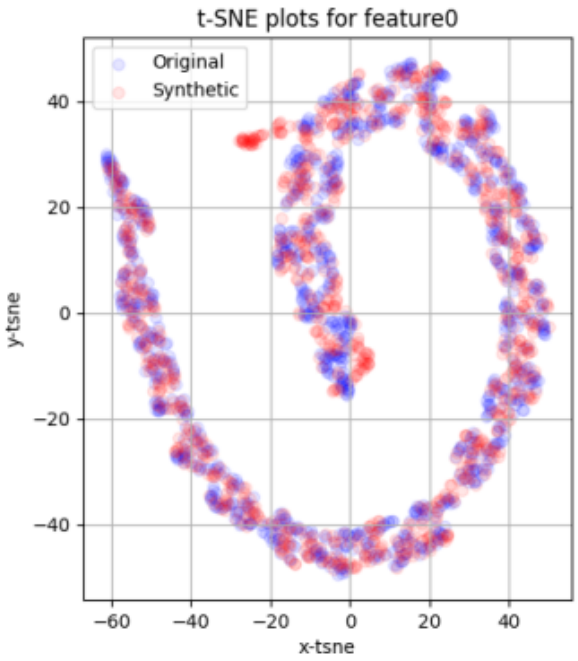


stock_tsne

图 8. 股票数据降维可视化效果



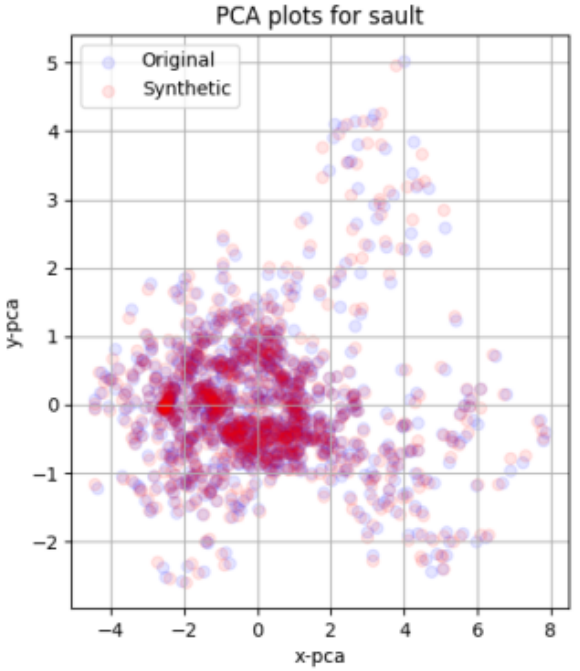
sine_PCA



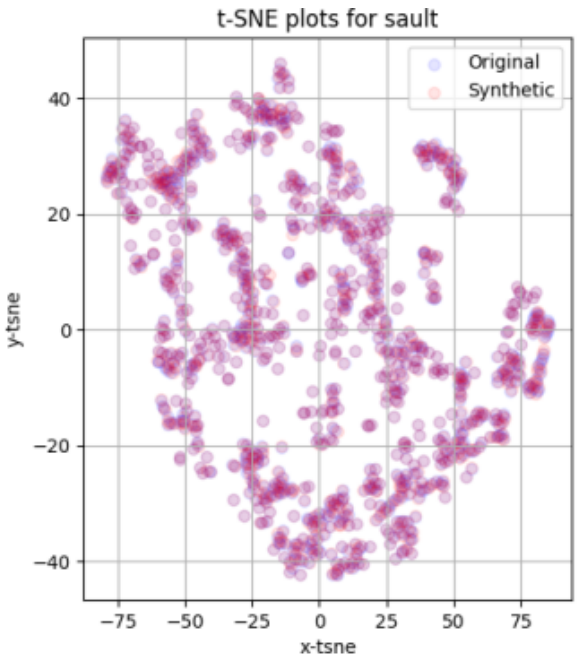
sine_tsne

图 9. 正弦数据降维可视化效果

此外，本文采用原论文中的框架，对 Argo 数据进行可视化展示，对比原始数据和合成数据的拟合程度，以评估模型的有效性，降维后的可视化效果图如下：



Argo_PCA



Argo_tsne

图 10. Argo 数据降维可视化效果

表 1. 预测分数

Predictive score	Stock	Energy	Sine	Argo
Random	0.037254	0.293746	0.188913	0.0196328
Random Average	0.037105	0.281628	0.187766	0.0194538
Cross concate	0.037181	0.257067	0.188169	0.0176249
Cross Average	0.037327	0.284203	0.187306	0.0186357
Paper Result	0.037±0.000	0.256±0.001	0.101±0.000	

表 2. 差异分数

Predictive score	Stock	Energy	Sine	Argo
Random	0.051160	0.500000	0.145775	0.043623
Random Average	0.499683	0.281628	0.076025	0.033256
Cross concate	0.499949	0.257067	0.061600	0.096354
Cross Average	0.499886	0.284203	0.022725	0.059638
Paper Result	0.410±0.000	0.256±0.001	0.019±0.010	

6 总结与展望

本次复现的论文提出了一种带有外推器的掩膜自动编码器 (ExtraMAE)，用于时间序列生成的自监督模型，ExtraMAE 提出了一个外推器来重构序列数据。论文选取股票、能源和正弦三种数据进行模型训练，并用随机、随机平均、拼接、拼接平均四种测试方法进行模型测试，此次所复现的结果与论文中所列结果大体一致。此外，本次复现还采用 Argo 数据进行模型的训练和测试，根据降维后的可视化效果显示，模型效果良好。本次复现我学习了序列数据重构的思路，虽然本文方法无法直接对应于此表层温盐重构，但在后续研究中将考虑是否可以结合本文方法深入研究。

参考文献

[1] Jiehui Xu, Jianmin Wang, Mingsheng Long, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34, 2021.

[2] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. 2021.

[3] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, 32, 2019.

[4] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. 2020.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. 27, 2014.

[6] Olof Mogren. C-rnn-gan: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*, 2016.

[7] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[8] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. 32, 2019.

[9] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. 2022.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2019.

[11] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.