Data Cleansing of Multiple Environmental Monitoring Time Series Using Spatio-Temporal Correlation

Ranier A. A. Moura, Domingos B. S. Santos, Daniel G. M. Lira, José E. B. Maia

[1]Universidade Estadual do Ceará – CCT-PPGCC-UECE
Av. Dr. Silas Munguba, 1700 – 60714-903 – Fortaleza, CE – Brasil

{ranier.moura,domingos.bruno,daniel.gleison}@aluno.uece.br

Abstract. *Computational applications based on sensor data is a reality, but the data collected and transmitted to the applications rarely arrives ready for use due to losses and noise of various types. In this work, an approach based on spatial temporal correlation is developed for cleaning data from multiple temporal series of sensors for noise, missing data and outliers. The method was tested on six publicly available real datasets and its performance was compared with a baseline method, an autoencoder denoising, and another published method. The results show that the proposed approach is competitive and requires less training data than competitors.*

Resumo. *Aplicações computacionais baseadas em dados de sensores são uma realidade, mas os dados coletados e transmitidos para as aplicações raramente chegam prontos para o uso devido a perdas e ruídos de vários tipos. Neste trabalho desenvolve-se uma abordagem baseada em correlação espaço temporal para limpeza de dados de múltiplas séries temporais de sensores quanto à ruído, dados ausentes e outliers. O método foi testato em seis conjuntos de dados reais publicamente disponíveis e o seu desempenho foi comparado com um método baseline, com um autoencoder denoising e com outro método publicado. Os resultados mostram que a abordagem proposta é competitiva e requer menos dados de treinamento do que os concorrentes.*

1. Introdução

Uma série tempotal (ST) S_t é uma sequência ordenada de valores indexada pelo índice $t \in \{0, 1, 2, ...\}$. Um exemplo de ST é a série histórica dos valores das temperaturas máximas anuais em um ponto determinado na superfície da terra. Dados de Séries Temporais (STs) adquiridos de sensores são o núcleo de um número cada vez maior de aplicações tão variadas quanto dispositivos vestíveis e *smart* [De Aquino et al. 2007], redes de sensores em geral [Oliveira and Rodrigues 2011], Internet das coisas industrial (IIoT) [Mois et al. 2017] e diagnóstico médico [Xiao et al. 2017] ou de máquinas [Sun et al. 2021].

Em uma classificação ampla, há dois modos de processamento de séries temporais, dependendo da aplicação. No primeiro, tem-se a série temporal (ST) armazenada em memória e o processamento é no modo *offline* [Fulcher 2017]. Neste caso não há

restrições de memória nem de tempo de processamento e o objetivo é construir um modelo para tarefas tais como predição de valores futuros ou pesquisa por eventos ocorridos.

No segundo, considerado neste trabalho, a série temporal deve ser processada no modo *data stream* [Read et al. 2020, Kong and Mamouras 2020], onde a aplicação utiliza os dados imediatamente quando eles chegam. As tarefas típicas neste caso são detecção de eventos *online* ou o processamento para extração e armazenamento da série comprimida via *features* de mais alto nível.

Entretanto, os dados originais fornecidos pelos sensores quase nunca estão prontos para uso devido a presença de ruídos, outliers ou dados ausentes, neste último caso devido a perdas ou erros de transmissão ou mesmo falha no sensoriamento. Sendo assim, a limpeza de dados de séries temporais é uma fase essencial para o sucesso das aplicações [Wang and Wang 2019]. O problema é como descrito na Figura 1. Busca-se desenvolver uma subcamada de processamento que recebe dados brutos (sujos) adquiridos diretamente pelos sensores e entrega dados limpos (pré-processados) para serem utilizados pelas aplicações.

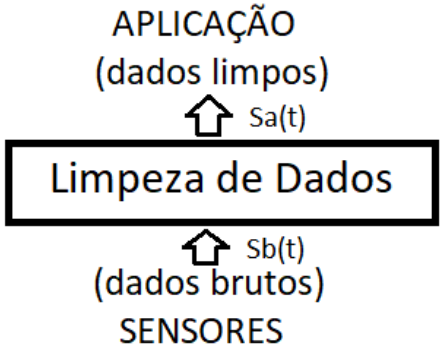

Figura 1. A limpeza de dados vista como subcamada de Internet das Coisas (IoT).

Neste trabalho são comparadas quatro abordagens para o projeto de uma subcamada de limpeza de dados em Séries Temporais de Redes de Sensores sem Fio (RSSF) de Monitoramento Ambiental [Mois et al. 2017, Oliveira and Rodrigues 2011]: Modelo de Regressão Linear Local (RLLAE), Modelo Gaussiano Espaço-Temporal (MGET), Redes Neurais Artificiais MLP (MLP - *multilayer perceptron*) e *AutoEncoder* (MLP-AE). No método RLLAE, cada série temporal é processada individualmente, já no método MGET o processamento de uma série é baseado em um pequeno grupo de séries altamente correlacionados, enquanto que nas abordagens integradas MLP e MLP-AE, um único modelo de rede neural de múltiplas entradas e múltiplas saídas é construido para todo o conjunto de séries temporais.

A contribuição deste trabalho em limpeza de dados de STs pode ser resumida em três pontos: primeiro, o trabalho propõe e avalia modelar os valores próximos no espaço e no tempo das séries de diferenças como uma distribuição gaussiana conjunta e aplica correlação espaço-temporal ao problema de limpeza de dados; segundo, comparar entre si métodos baseados em séries individuais e em grupos de séries, com algoritmos neurais baseados no conjunto de STs; e terceiro, avaliar comparativamente *AutoEncoder* MLP

denoising na tarefa de limpeza de dados de múltiplas STs na presença de ruídos, *outliers* e dados ausentes.

Em continuação, a Seção 2 apresenta uma breve revisão de trabalhos relacionados com esta pesquisa bem como aqueles utilizados na comparação de desempenho e a Seção 3 apresenta de forma sucinta, porém rigorosa, os fundamentos de cada método analisado. A Seção 4 é de resultados e discussão dos achados, e algumas conclusões são destacadas na Seção 5.

2. Trabalhos Relacionados

Para localizar este trabalho no contexto da pesquisa em limpeza de dados de STs, esta seção revisa alguns trabalhos mais diretamente relacionados. Artigos de revisão (*reviews*) que analisam o tema mais amplamente podem ser encontrados em [Wang and Wang 2019] e [Zhang et al. 2017].

Em [Tan et al. 2005] as performances de Filtro de Kalman e Regressão são comparadas como solução para o problemas dos dados incompletos, ruidosos e de baixa confiabilidade de múltiplas STs recebidas de sensores. Os resultados dos experimentos mostram que regressão polinomial de alta ordem e Filtro de Kalman funcionam igualmente bem na maioria dos casos, mas que o Filtro de Kalman supera a regressão quando os dados possuem alta variabilidade. Utilizando um dos *datasets* de temperatura que será também utilizado neste artigo, os autores encontraram MAE de 2,32 para o Filtro de Kalman e 2,08 para Regressão Linear Multivariada. Este resultados serão utilizados para comparação na Seção de Resultados. Predição em múltiplas STs também foi feita em [Nunes et al. 2020] utilizando inferência fuzzy.

Sendo o sensoriamento de máquinas um componente chave da indústria inteligente, o paper [Ding et al. 2019] propõe um sistema denominado Cleanits para limpeza de STs industriais. O sistema procura atender a três tipos de erros: valores ausentes, valores inconsistentes com o contexto e valores e subsequências anômalas. A abordagem é uma composição de técnicas estatísticas com conhecimento do domínio. O sistema é demonstrado com STs de dois cenários de aplicação industrial mas ele não apresenta comparação de performance.

Em certos ambientes de monitoramento de edifícios comerciais e residenciais o efeito que predomina nas STs é a perda de dados. Para atacar este problema o trabalho [Liguori et al. 2021] apresenta uma abordagem *data-driven* para preencher estes segmentos ausentes. O trabalho avalia três diferentes redes neurais *AutoEncoder* para reconstruir dados de STs de curto prazo. O desempenho é comparado com métodos existente utilizando a medida de desempenho RMSE (*root mean squared error*) em *data streams* de temperatura, umidade relativa e CO2 *indoor* com performance superior ao dos trabalhos anteriores utilizados na comparação.

Séries temporais do monitoramento de turbinas eólicas marinhas sofrem com muitos dados ausentes devido as condições ambientais nas quais os dados são sensoriados e transmitidos. Correlação espaço-temporal e correlação de atributos foram utilizadas conjuntamente em [Sun et al. 2021] para imputação de dados ausentes neste cenário. A ideia é aprender um modelo de correlação espaço-temporal entre as falhas e outras funcionalidades das turbinas eólicas para imputar os dados de forma atrasada. O método teve desempenho estado da arte nos testes relatados.

Neste parágrafo faz-se um cotejamento dos trabalhos revisados nesta seção com a proposta desta pesquisa. Predição com regressão linear é utilizada em [Tan et al. 2005] e [Ding et al. 2019] para detectar *outliers* baseada em limiar (*threshold*) de erro da mesma forma que é feita neste trabalho. Nos métodos neurais, o artigo [Liguori et al. 2021] descreve a aplicação de *AutoEncoder* em limpeza de STs mas o método difere do que é feito neste trabalho. Por fim, embora os autores em [Sun et al. 2021] também apliquem correlação espaço-temporal para limpeza de dados de STs, a abordagem utilizada é diferente da que é proposta neste trabalho. Não foram encontrados trabalhos publicados aplicando correlação espaço-temporal do modo que aqui é feito.

3. Métodos

Esta seção descreve os métodos deste trabalho, que são: Regressão Linear com Alisamento Exponencial Simples (RLLAE) na Seção 3.1, Modelo Gaussiano Espaço-Temporal (MGET) na na Seção 3.2 e Redes Neurais Artificiais (RNAs) MLP e *autoencoder* na Seção 3.3.

Uma matriz de dados de STs $\mathbf{S}(t)$ é uma matriz de dimensão NxM com a coluna i armazenando N valores de $S_i(t)$, a i-ésima série temporal, e em que cada linha é um vetor $\mathbf{s}_j(t)$ com os valores das M STs no tempo t. Além disso, $T = t_i - t_{i-1}$ é intervalo de tempo entre amostras, e $j \in \{1, ..., M\}$ indexa cada série em um conjunto de múltiplas séries temporais. M é número de séries no conjunto e N é o número de pontos em cada série, aqui suposto ser o mesmo para todas as séries, sem perda de generalidade. Adicionalmente, define-se também a série temporal de diferenças $Sd_i(t)$ da i-ésima série, dada por:

$$Sd_i(0) = 0, \qquad Sd_i(t) = S_i(t) - S_i(t-1). \tag{1}$$

Tomando a Figura 1 como referência, denote por $\mathbf{s}b(t)$ o vetor de dados brutos com os valores das séries temporais no tempo t, sendo $Sb(t)$ uma série genérica e $Sb_i(t)$ a i-ésima série temporal do vetor $\mathbf{s}b(t)$. Similarmente, denote por $\mathbf{s}a(t)$ o vetor de dados limpos das séries temporais no tempo t, sendo $Sa(t)$ uma série genérica e $Sa_i(t)$ a i-ésima série temporal do vetor $\mathbf{s}a(t)$. Assume-se também que foi coletado um conjunto de dados de treinamento representado pela matriz de dados $\mathbf{S}(t)$. Onde necessário, esta notação será utilizada nas próximas três subseções para descrever os algoritmos utilizados neste trabalho. Em todos os procedimentos supõe-se dispor de um conjunto de treinamento livre de *outliers*. O modo de processamento é *data stream*: cada amostra é processada ao chegar.

3.1. Regressão Linear Local com Alisamento Exponencial Simples (RLLAE)

Neste método cada série temporal é tratada individualmente. O procedimento é dado no pseudocódigo do Algoritmo 1. Para dado ausente a regra é imputar o último valor válido (filtrado) da ST. A ideia neste método é estimar o próximo valor da série utilizando um modelo de regressão linear construído com base nas últimas $N = 3$ amostras após um alisamento exponencial simples (AES). Se o valor recebido fica além de um limiar absoluto relativo a esta predição, ele é considerado outlier. O outlier é enviado para um procedimento a parte e esta posição na ST recebe o tratamento de dado ausente. RLLAE será tomado como método *baseline* de performance nos experimentos por ser um modelo simples e individual por série. AES é aplicado localmente para evitar erro acumulado.

Algorithm 1: Pseudocódigo para o algoritmo Regressão Linear Local com Alisamento Exponencial (RLLAE).

Data: Os últimos 3 pontos limpos da Série Temporal $Sa(t-3)$, $Sa(t-2)$, $Sa(t-1)$ e um ponto de entrada sujo $Sb(t)$.
Result: O próximo ponto da série temporal filtrado $Sa(t)$.
Ler $Sb(t)$;
if $Sb(t) = ausente$ **then**
  $Sb(t) = Sa(t-1)$;
end
$Sb(t) = alisamentoExpS(Sb(t))$;
Calcula a predição linear $\hat{s}(t)$ baseada em RLL;
if $abs(\hat{s}(t) - Sb(t)) > 2\sigma$ **then**
  $outlier = Sb(t)$;
  $Sb((t) = \hat{s}(t)$;
end
$Sa(t) = alisamentoExpS(Sb(t))$;
$out = trataOutlier(outlier)$; # se existir ;
return $Sa(t), out$;

Seguindo o pseudocódigo, após ler uma nova entrada da série $Si(t)$, o Algoritmo 1 imputa o valor anterior caso a leitura resulte em dado ausente. Se um valor é obtido, ele é comparado com um valor predito pela RLL e é considerado *outlier* se o desvio absoluto em relação à leitura anterior for maior que 5 desvios padrões da série de diferenças dos dados de treinamento. Em qualquer caso, o dado bruto $Sb(t)$ recebe alisamento exponencial simples gerando o dado limpo $Sa(t)$. Os métodos auxiliares são a Regressão Linear Local (RLL) e o Alisamento Exponencial Simples. A RLL pedida no Algoritmo 1 estima o valor da série no tempo t, $\hat{S}(t)$, utilizando os valores passados $S(t-1)$, $S(t-2)$ e $S(t-3)$. As equações em variáveis genéricas para isto são dadas por [Morettin and Toloi 2006],

$$\hat{y}(t_i) = \alpha_0 + \alpha_1(t_i - t_{i-N}) \tag{2}$$

onde α_0 e α_1 são dados por:

$$\alpha_1 = \frac{\sum_{i=1}^{N}(\bar{x} - x_i)(\bar{y} - y_i)}{\sum_{i=1}^{N}(\bar{x} - x_i)^2} \tag{3}$$

$$\alpha_0 = \bar{y} - \alpha_1\bar{x}, \tag{4}$$

onde $\{x_i, y_i\}$ são os dados para a regressão e $\bar{x}$ e $\bar{y}$ são as médias de x_i e y_i.

O Alisamento Exponencial Simples utiliza a fórmula [Morettin and Toloi 2006]

$$\hat{S}a(t) = \alpha Sa(t-1) + (1-\alpha)Sb(t). \tag{5}$$

O parâmetro α do alisamento exponencial é dependente do problema e pode ser determinado a partir dos dados de treinamento por um procedimento de otimização, no qual, variando α, busca-se minimizar o erro de predição dos valores da série um passo à frente, ou seja,

$$\alpha^* = min_\alpha(\hat{S}a(t) - Sa(t))^2. \tag{6}$$

3.2. Modelo Gaussiano de Correlação Espaço-Temporal (MGET)

MGET é o método proposto neste trabalho. A hipótese nestes métodos é que as séries temporais de diferenças $sd_i(t)$ de sensores correlacionados pode ser aproximadas como tendo sido geradas por um vetor de distribuições gaussianas conjunto $\mathbf{s}d(t)$. As contribuições de métodos puramente temporal e puramente espacial também são estudados nesta seção. A determinação do número p de séries correlacionadas com a ST i a considerar no modelo é um problema experimental de seleção de modelos. Experimentos preliminares baseados em erros de predição nos conjuntos de treinamento dos *datasets* indicaram que $p = 2$ é um bom compromisso.

Assuma que está disponível uma matriz de dados de séries de diferenças centradas e padronizadas ($\sigma = 1.0$) para treinamento $\mathbf{S}d(t)$. Para encontrar as $p = 2$ STs mais correlacionadas com cada ST, primeiro obtem-se a matriz de correlação dos dados $\mathbf{R}d$:

$$\mathbf{R}d = \mathbf{S}d^T\mathbf{S}d, \tag{7}$$

onde $()^T$ indica o transposto de uma matriz. Nesta matriz as colunas dos dois maiores valores na linha i fora da diagonal são os índices das duas ST mais correlacionadas com $S_i(t)$. Sejam j e k estes índices.

Assim, para o Método Gaussiano Temporal (MGT), o vetor de variáveis aleatórias normais para a ST i é formado por $s_i = [sd_i(t), sd_i(t-1), sd_i(t-2)]^T$ e para o Método Gaussiano Espacial (MGE) tem-se $s_i = [sd_i(t), sd_j(t), sd_k(t)]^T$. Já para o Método Gaussiano Espaço–Tempporal (MGET) este vetor fica dado por $s_i = [sd_i(t), sd_i(t-1), sd_i(t-2), sd_j(t), sd_k(t)]^T$. Sejam $\mathbf{m}$ e $\mathbf{\Sigma}$ o vetor de médias e a matriz de covariâncias dos dados para um dos métodos qualquer. Então a estimativa linear para $sd_i(t)$ é dada por [Yates and Goodman 2014]

$$\hat{s}d_i(t) = m_i + \sigma_i\Sigma_i^{-1}(\mathbf{s}_i - \mathbf{m}_i), \tag{8}$$

onde m_i é a média de sd_i, $\mathbf{m}_i$ é o subvetor de médias extraído de $\mathbf{m}$ sem a componente m_i, σ_i é o vetor de covariâncias de sd_i com as demais STs, e $\mathbf{\Sigma}_i$ é a submatriz de $\mathbf{\Sigma}$ quando a linha e a coluna i são suprimidas. O pseudocódigo para estes métodos é semelhante ao do RLLAE no Algoritmo 1 substituindo a predição RLL pela predição baseada em Correlação Espaço-Temporal dada na Equação acima.

3.3. Redes Neurais Artificiais - MLP e *AutoEncoder* (MLP, MLP-AE)

Uma alternativa aos modelo anteriores são o modelos neurais. Enquanto no método RLLAE o processamento de cada ST é independente das demais, e no algoritmo MGET a limpeza de uma ST apoia-se no comportamento de algumas outras STs altamente correlacionadas com esta, nos modelos neurais [Shrestha and Mahmood 2019] MLP e MLP-AE o vetor de STs completo é utilizado para treinar a RNA (Rede Neural Artificial) no modo vetor-a-vetor. Para um conjunto com M STs a RNA terá M entradas e M saídas. Dois modelos são avaliados: uma RNA MLP rasa, com uma camada escondida, e um *Auto-Encoder* MLP com 3 camadas escondidas. Estas estruturas estão mostradas na Figura 2. Busca em grade para minimizar R^2, entre a metade e o dobro do número de entradas, foi utilizada para determinar o número de neurônios em cada camada. Para o *autoencoder* o

número de neurônios no gragalo foi fixado em metade do número de STs ficando com um grau de liberdade na camada intermediária.

Por limitação de espaço esta seção não apresenta conceitos teóricos das redes neurais utilizadas. O leitor interessado deve consultar as referências indicadas. Apenas do ponto de vista de estrutura, a Figura 2 mostra uma comparação entre MLP e *Auto-Encoder*. Note que no *AutoEncoder* há um estreitamento na representação interna e essa propriedade pode ser utilizada para filtragem de ruído (*denoising*).

Para descrever a estratégia de treinamento, seja $\mathbf{S}tr$ uma matriz de dados de treinamento de STs com cada coluna armazenando uma ST. O procedimento consiste em treinar a RNA com $\mathbf{S}tr$ como saída (variável dependente) e uma versão 'suja' de $\mathbf{S}tr$, $\mathbf{S}trs = \mathbf{S}tr + Ruído$ como a entrada da RNA. A ideia é treinar as RNAs como regressores vetoriais robustos a ruídos e *outliers* nos dados de entrada. Ao final do treinamento a RNA minimiza a função de perdas

$$J(\mathbf{S}tr, \theta) = \sum \sum ||\mathbf{S}tr - \hat{\mathbf{S}}tr||^2, \tag{9}$$

onde θ é o vetor de parâmetros da RNA e $\hat{\mathbf{S}}tr$ é a saída gerada pela RNA.

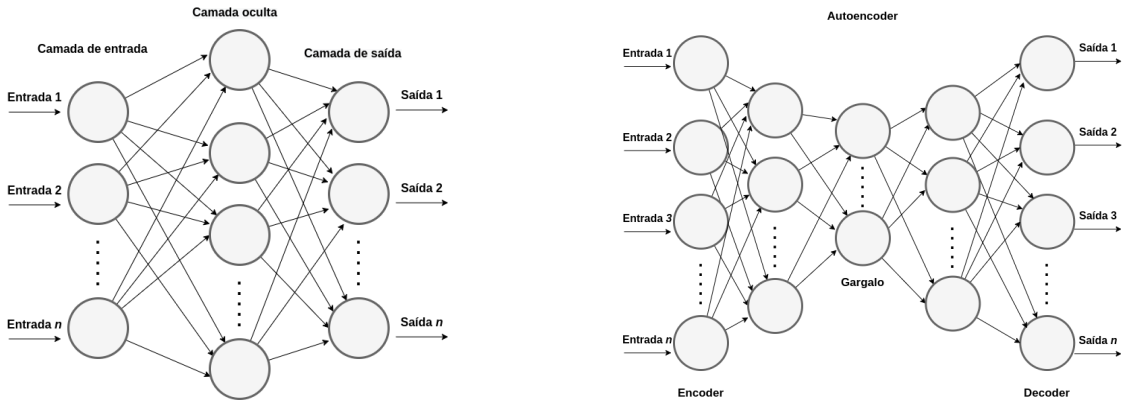

Figura 2. As topologias das redes neurais utilizadas MLP (esquerda) e MLP-AE (direita).

4. Resultados e Discussão

Para avaliar a performance do método proposto foram realizados experimentos com 6 conjuntos de dados. Em todos os casos, os dados de teste foram contaminados com 10% de valores ausentes e 10% de *outliers* em locais aleatoriamente escolhidos de cada ST. Um segundo teste foi realizado nas condições descritas em [Tan et al. 2005] de modo a poder comparar o desempenho com o de outro trabalho publicado. Todos os algoritmos foram codificados em Python 3 utilizando as bibliotecas *numpy* e *scikit-learn* [Pedregosa et al. 2011]. A execução não exigiu hardware especial.

4.1. *Datasets*

Os experimentos foram realizados com conjuntos de dados (*datasets*) publicamente disponíveis. Os autores em [Le Borgne et al. 2007] preprocessaram e publicaram um *dataset*

[1] derivado do Intel-Lab-Data *dataset* [Bodik et al. 2004], o qual contem 14.400 medições de temperatura de 52 sensores correspondendo a 5 dias de medição com um intervalo de 30 s entre medições. Este conjunto de dados será denotado S14K neste trabalho. Do *dataset* S14K derivou-se dois outros conjuntos de dados mais desafiadores para a tarefa de limpeza de dados: S120 e S60. S120 registra temperaturas horárias, contem 120 pontos e foi obtido de S14K tomando uma amostra a cada 120 pontos, para os 52 sensores, resultando em medições de temperatura em intervalos de uma hora. Já S60 tem 60 pontos apenas, registrando as medições a cada duas horas.

O segundo conjunto de dados utilizado nos experimento foi Appliances energy prediction [Candanedo and Feldheim 2016], que será denotado por S19K, disponível no repositório UCI[2]. O dataset contém 19.735 pontos de 29 atributos dos quais 4 são data e hora e 2 são ruídos aleatórios artificialmente incluídos. Nos experimentos foram utilizados 18 séries, colunas 4 à 21, que são STs de variáveis ambientais como temperaturas e umidades coletadas em intervalos de 10 min para uma pesquisa em projetos de prédios de baixa energia. Assim como foi feito para S14K, de S19K foram derivados dois outros *datasets*. S4K com medições horárias, o qual pega um ponto a cada 6 e S2K que pega um ponto a cada 12 pontos de S19K, registrando medições a cada duas horas.

A motivação para gerar estes *datasets* derivados foi obter STs nas quais a correlação temporal cai mais rapidamente e com maior variação de amplitude por passo. A Figura 3 mostra aspectos qualitativos típicos das STs nestes *datasets*. Esta figura mostra os primeiros 40% da ST-05 de cada *dataset* com S19K na primeira coluna e S14K na segunda coluna. As Figuras 3a e 3b mostram as ST originais e as demais mostram as STs de diferenças. As faixas de variações de amplitude das séries de diferenças mostram o desafio crescente quando se passa de S19K para S4K (3b para 3d) e de S14K para S120 (3c para 3e). Por outro lado, de S120 para S60 e de S4K para S2K não cresce a variação de amplitude mas reduz-se muito os dados disponíveis para treinamento de modelos.

4.2. Índices de desempenho

Para avaliar e comparar as performances dos métodos adotou-se o Erro Médio Absoluto (MAE) [Morettin and Toloi 2006] sobre todas as séries e o Coeficiente de Determinação Ajustado (R^2)[Morettin and Toloi 2006]. O índice MAE em variáveis genéricas y e $\hat{y}$ é dado por

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y - \hat{y}|, \tag{10}$$

onde y refere-se ao valor observado e $\hat{y}$ refere-se ao valor predito, e os índices R^2 e $R^2 - ajustado$ são calculados por

$$R^2 = \{\frac{\sum_1^N (y-\bar{y})(\hat{y}-\hat{\bar{y}})}{\sqrt{\sum_1^N (y-\bar{y})}\sqrt{\sum_1^N (\hat{y}-\hat{\bar{y}})}}\}^2 \tag{11}$$

e

$$R^2 - ajustado = 1 - \frac{N-1}{N-(k+1)}(1-R^2), \tag{12}$$

[1]http://www.ulb.ac.be/di/labo/code/PCAgExpe.zip

[2]https://archive.ics.uci.edu/ml/index.php

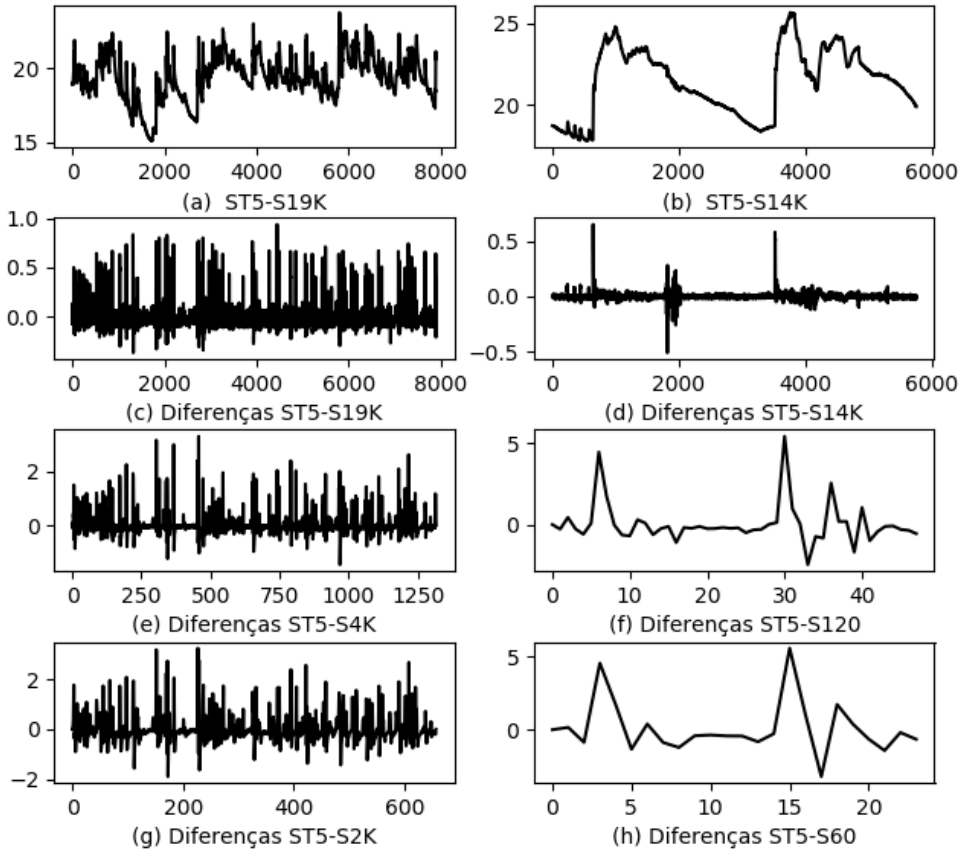

Figura 3. Aspectos qualitativos de STs de diferenças típicas dos *datasets* utilizados. Mostrada a ST-5 de (a) S19K, (b) S14K, (c) S19K diferenças, (d) S14K diferenças, (e) S4K, (f) S120, (g) S2K e (h) S60.

onde y e $\hat{y}$ são como antes e $\bar{y}$ e $\hat{\bar{y}}$ são os valores médios e k é o número de variáveis independentes, neste caso o número de STs.

$RMSE$ (*root mean squared error*) [Morettin and Toloi 2006] é outra medida de performance de modelos frequentemente utilizada. $RMSE$ dá maior peso aos maiores desvios e a vantagem de MAE sobre $RMSE$ neste contexto é porque supõe-se que os *outliers* serão eliminados. R^2, por sua vez é uma medida adimensional com valores entre 0 e 1, que mede o quão bem o modelo de fato se ajusta ao fenômeno que está sendo representado. $R^2 - ajustado$ compensa um viés existente em R^2 quando cresce o número de variáveis independentes.

4.3. Resultados

Os processos de otimização descritos na Seção 3 foram aplicados para determinar α do alisamento exponencial e o número de neurônios da MLP e do *AutoEncoder*. Para o alisamento exponencial adotou-se um só α para cada *dataset*: 0,97 para S19K, S4K e S2K, e 0,99 para S14K, S120 e S60. Para as Redes Neurais, o número de neurônios na camada

escondida da MLP resultou em 72 neurônios para S19K, S4K e S2K, e 30 neurônios para S14K, S120 e S60. Para o modelo *AutoEncoder* com 5 camadas os números de neurônios resultaram em (52-40-26-40-52) para S19K, S4K e S2K e (18-14-9-14-18) para S14K, S120 e S60.

Em cada experimento, o *dataset* foi dividido em três subconjuntos: 40% para treinamento, 20% para validação e 40% para teste. Note que para os *datasets* S120 e S60 resultou em um número muito pequeno de pontos para treinamento. Isto torna-se um fator crítico para métodos dependentes de grandes conjuntos de treinamento como as RNAs. Em cada caso, o conjunto de treinamento foi utilizado para construir o modelo com o ajuste dos hiperparâmetros (seleção de modelo) realizado sobre o conjunto de validação. O modelo final, treinado com treinamento + validação é avaliado sobre o conjunto de teste. Os resultados estão registrados na Tabela 1. Os valores de R^2 na tabela refere-se a $R^2 - ajustado$.

Tabela 1. Comparativo de desempenho dos métodos de limpeza de dados de séries temporais múltiplas avaliados: 10% de dados ausentes mais 10% de *outliers*.

Dataset	Método											
-	RLLAE		MGT		MGE		MGET		MLP		MLP-AE	
métrica	MAE	R2	MAE	R2	MAE	R2	MAE	R2	MAE	R2	MAE	R2
S14K	0,72	0,89	0,015	0,99	0,063	0.98	0,033	0,99	0,29	0.99	0,32	0,99
S120	1,90	0,65	0,068	0,98	0,057	0,98	0,055	0,99	1,01	0,90	0,92	0,91
S60	2,41	0,61	0,110	0,97	0,048	0,99	0,049	0,99	1,83	0,78	1,33	0,62
S19K	0,81	0,91	0,012	0,99	0,733	0,26	0,014	0,99	0,34	0,99	0,21	0,99
S4K	1,79	0,78	0,079	0,99	0,974	0,401	0,108	0,99	0,54	0,96	0,51	0,96
S2K	2,23	0,71	0,132	0,97	0,920	0,46	0,191	0,96	0,62	0,94	0,64	0,94

Discussão: Destaque-se que para consjuntos de dados pequenos, como o S60, no qual apenas 20 pontos estão disponíveis para treinamento, MGET trabalhou melhor que os demais. Note também que, como seria razoável esperar, RLLAE e MGT, ambos baseados apenas em informação temporal, apresentaram desempenhos semelhantes, mas melhor para MGT. Os métodos de Redes Neurais são sempre dependentes de extensos conjuntos de treinamento e por isso trabalharam ruim no menor *dataset*.

Comparação com outro trabalho: Um dos *datasets* utilizados em [Tan et al. 2005] foi o de temperaturas horárias, equivalente ao S120 utilizado neste trabalho, ambos derivados do Intel Lab Data, de forma que permite uma comparação justa. Aquele artigo registra os seguintes resultados de MAE para algoritmos de limpeza de dados: 2,32 para o Filtro de Kalman, 2,08 para Regressão Linear Multivariada, 0,90 para regressão quadrática e 0,12 para regressão multi-quádrica. Comparativamente, os valores de MAE correspondentes na Tabela 1 são: 0,055 para MGET, 1,01 para MLP e 0,92 para MLP-AE. Pode-se notar que MGET trabalha melhor que todos os demais, sendo porém conceitualmente mais simples e exigindo muito menos dados de treinamento.

5. Conclusão

Neste trabalho foram propostas e avaliadas soluções para limpeza de dados de conjuntos com múltiplas STs de monitoramento ambiental. Na técnica RLLAE, a limpeza de uma

ST $S_i(t)$ é baseada apenas em informação da própria $S_i(t)$. No algoritmo MGET, a limpeza de uma ST $S_i(t)$ considera, além de informação da própria $S_i(t)$, informação de um pequeno grupo de STs altamente correlacionadas com $S_i(t)$. Finalmente, nas abordagens neurais MLP e AE-MLP, todo o conjunto de STs é tomado de uma vez para treinar RNAs vetor-à-vetor, com M entradas e M saídas, para um conjunto de M STs.

A performance foi comparada utilizando o Erro Absoluto Médio (MAE) e o Coeficiente de Determinação (R2) para contaminação com 10% de dados ausentes e 10% de *outliers*. Em todos os casos, aos dados ausentes são imputados os últimos valores válidos da ST. Os valores imputados, tal qual uma leitura de um novo valor, são submetidos ao procedimento de limpeza.

Em conclusão, MGET foi o único método que trabalhou no melhor nível quando variando o tamanho do *dataset* e o grau de dificuldade em termos de amplitude das séries de diferenças. A principal limitação de MGET é que os testes realizados se limitaram a STs de monitoramento ambiental e não podendo os resultados serem generalizados para STs em geral. Em todos os procedimentos supõe-se dispor de um conjunto de treinamento limpo.

Em continuação a este trabalho pretende-se ampliar a comparação com outros trabalhos, diversificar os *datasets* de teste e avaliar a performance do ponto de vista de algumas aplicações.

Agradecimentos: Ranier Moura é bolsista mestrado FUNCAP e Domingos Santos é bolsistas de mestrado CAPES. José E.B. Maia - ORCID: 0000-0002-4983-1724.

Proceedings of **ENIAC 2021** - 18th National Meeting on Artificial and Computational Intelligence - Online, November 29 to December 3, 2021, Brazil.

Referências

Bodik, P., Hong, W., Guestrin, C., Madden, S., Paskin, M., and Thibaux, R. (2004). Intel lab data. *Online dataset*.

Candanedo, L. M. and Feldheim, V. (2016). Accurate occupancy detection of an office room from light, temperature, humidity and co2 measurements using statistical learning models. *Energy and Buildings*, 112:28–39.

De Aquino, A. L., Figueiredo, C. M., Nakamura, E. F., Buriol, L. S., Loureiro, A. A., Fernandes, A. O., and Claudionor Jr, J. (2007). Data stream based algorithms for wireless sensor network applications. In *21st International Conference on Advanced Information Networking and Applications (AINA'07)*, pages 869–876. IEEE.

Ding, X., Wang, H., Su, J., Li, Z., Li, J., and Gao, H. (2019). Cleanits: a data cleaning system for industrial time series. *Proceedings of the VLDB Endowment*, 12(12):1786–1789.

Fulcher, B. D. (2017). Feature-based time-series analysis. *arXiv preprint arXiv:1709.08055*.

Kong, L. and Mamouras, K. (2020). Streamql: a query language for processing streaming time series. *Proceedings of the ACM on Programming Languages*, 4(OOPSLA):1–32.

Le Borgne, Y.-A., Dricot, J.-M., and Bontempi, G. (2007). Principal component aggregation for energy efficient information extraction in wireless sensor networks. *Knowledge Discovery from Sensor Data*.

Liguori, A., Markovic, R., Dam, T. T. H., Frisch, J., van Treeck, C., and Causone, F. (2021). Indoor environment data time-series reconstruction using autoencoder neural networks. *Building and Environment*, 191:107623.

Mois, G., Folea, S., and Sanislav, T. (2017). Analysis of three iot-based wireless sensors for environmental monitoring. *IEEE Transactions on Instrumentation and Measurement*, 66(8):2056–2064.

Morettin, P. A. and Toloi, C. (2006). *Análise de séries temporais*. Editora Blucher.

Nunes, F. R., Macêdo, C. d. S., Soares, J. d. N., Cavalcante, H. G., Brilhante, M. Q., and Maia, J. E. (2020). Fuzzy-probabilistic approach for dense wireless sensor network. In *International Conference on Intelligent Systems Design and Applications*, pages 1018–1027. Springer.

Oliveira, L. M. and Rodrigues, J. J. (2011). Wireless sensor networks: a survey on environmental monitoring. *JOURNAL OF COMMUNICATIONS*, 6(2):143.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Read, J., Rios, R. A., Nogueira, T., and de Mello, R. F. (2020). Data streams are time series: Challenging assumptions. In *Brazilian Conference on Intelligent Systems*, pages 529–543. Springer.

Shrestha, A. and Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE Access*, 7:53040–53065.

Sun, C., Chen, Y., and Cheng, C. (2021). Imputation of missing data from offshore wind farms using spatio-temporal correlation and feature correlation. *Energy*, 229:120777.

Tan, Y. L., Sehgal, V., and Shahri, H. H. (2005). Sensoclean: Handling noisy and incomplete data in sensor networks using modeling. *Main*, pages 1–18.

Wang, X. and Wang, C. (2019). Time series data cleaning: A survey. *IEEE Access*, 8:1866–1881.

Xiao, H., Lu, C., and Ogai, H. (2017). A new low-power wireless sensor network for real-time bridge health diagnosis system. In *Society of Instrument and Control Engineers of Japan (SICE), 2017 56th Annual Conference of the*, pages 1565–1568. IEEE.

Yates, R. D. and Goodman, D. J. (2014). *Probability and stochastic processes: a friendly introduction for electrical and computer engineers*. John Wiley & Sons.

Zhang, A., Song, S., Wang, J., and Yu, P. S. (2017). Time series data cleaning: From anomaly detection to anomaly repairing. *Proceedings of the VLDB Endowment*, 10(10):1046–1057.