Utilização de RNNs para Predição de mudança de estado em compressor industrial

Fernanda Mickosz Villa Verde

CPE727 – Deep Learning

6 de dezembro de 2025

Sumário

Motivação e Objetivos do trabalho

Motivação:

- Compressores industriais são ativos críticos (segurança, disponibilidade, custo).
- Mudanças de regime operacional (estados) afetam: Desempenho energético; Risco de falha; Planejamento de manutenção.
- Predizer **quando** ocorrerá a próxima mudança de estado é útil para:
  - Detecção precoce de desvios;
  - Replanejamento operacional;
  - Suporte à decisão para operadores.

Objetivos do trabalho:

- Modelar séries temporais industriais (compressores) com amostragem irregular, dados faltantes e alta dimensionalidade utilizando arquiteturas recorrentes.
- Predição dos **estados** em janelas futuras;
- Predição do **tempo até a próxima mudança de estado**.

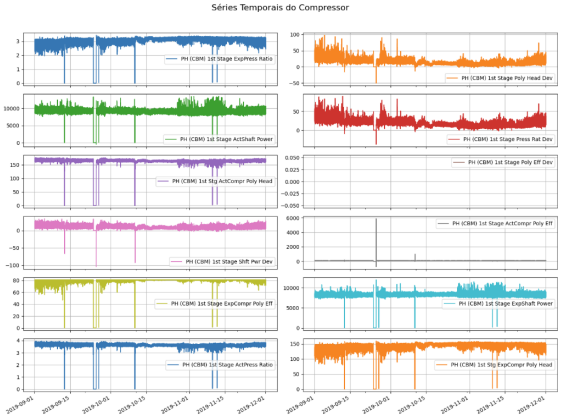
Comparar arquiteturas:

- Modelos DEEP, LSTM, GRU, BiLSTM e BiGRU com GRU fuser, todos com otimização Adam;
- Variações de treinamento (warmup, scheduler, RAdam, AdamW, variational dropout, difusão, log-likelihood e layernorm).

Base de dados e pré-processamento

Base de dados

Cognite — sinais reais de sensores de um compressor industrial offshore.



Pré-processamento

- Estados operacionais:
  - Estados originais: 6 (0 a 5);
  - Estados finais: 3 (normal, alerta, falha).
- Normalização robusta;
- Amostragem a cada 5 minutos;
- Janelas temporais:
  - $L = 40$ amostras ($\approx$ 3h20min);
  - Deslocamento de 40 amostras (janelas não sobrepostas).

Definição do problema

- Série multivariada: $x_t \in \mathbb{R}^d$, $t = 1, \ldots, T$, com $d = 11$ sensores.
- Estados discretos: $s_t \in \{0, 1, 4\}$ (após fusão dos 6 estados iniciais).
- Para cada janela $X = (x_{t-L+1}, \ldots, x_t)$, queremos:
  - Predizer a distribuição sobre estados futuros:

$$p(s_{t+\Delta} \mid X)$$

  - Predizer o tempo até a próxima mudança de estado:

$$\hat{\tau} = f_\theta(X)$$

- Problema multi-tarefa:
  - Classificação de estados (estado futuro) + regressão de tempo (tempo até a mudança).

A partir de janelas de histórico de sensores, o modelo aprende simultaneamente a prever o próximo regime operacional e o tempo restante até a próxima mudança de estado.

Função de custo

- Função de custo multi-tarefa:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{rec}} + \lambda_2 \mathcal{L}_{\text{diff}} + \lambda_3 \mathcal{L}_{\text{time}} + \lambda_4 \mathcal{L}_{\text{mask}} + \lambda_5 \mathcal{L}_{\text{VAE-}x} + \boxed{\lambda_6 \mathcal{L}_{\text{VAE-}\tau}}$$

  - $\mathcal{L}_{\text{rec}}$: erro de reconstrução da série observada (ajuste aos sinais reais).
  - $\mathcal{L}_{\text{diff}}$: perda de difusão, regularizando a predição de ruído (robustez a incerteza).
  - $\mathcal{L}_{\text{time}}$: erro na predição do tempo até a próxima mudança de estado.
  - $\mathcal{L}_{\text{mask}}$: aprendizado explícito do padrão de dados ausentes.
  - $\mathcal{L}_{\text{VAE-}x}$: regularização variacional da reconstrução dos sinais.
  - $\boxed{\mathcal{L}_{\text{VAE-}\tau}}$: **modelagem probabilística do tempo até falha**, penalizando mudanças próximas e calibrando incerteza.

$$\mathcal{L}_{\text{VAE-}\tau} = \mathbb{E}_{q(z_\tau|\mathbf{X})}\big[-\log p(\tau \mid z_\tau)\big] + \beta\,\mathrm{KL}\big(q(z_\tau)\,\|\,\mathcal{N}(0,I)\big)$$

- Peso maior atribuído a instantes próximos à mudança de estado:

$$\log p(\tau \mid z_\tau) \;\propto\; w_t \cdot \|\hat{\tau} - \tau\|^2, \quad w_t \gg 1$$

- Permite estimar **incerteza**, **intervalos de confiança** e **cobertura probabilística**.

Arquiteturas avaliadas

- **TSDF_DEEP**:
  - Modelo feedforward sobre janelas (baseline não-recorrente).
- **TSDF_LSTM**:
  - RNN LSTM unidirecional;
  - Versão **BiLSTM** (`bi_lstm=True`).
- **TSDF_GRU**:
  - RNN GRU unidirecional;
  - Versão **BiGRU** (`bi_gru=True`).

Hiperparâmetros principais

- `in_channels = 11`
- `hidden_dim = 11 * 32`
- `status_dim = 3`
- Janela: `window_size = 40`, `window_step = 40`

Arquitetura baseline: TSDF_DEEP (feedforward)

- Entrada: janela temporal $X = (x_1, \ldots, x_L)$, com $x_t \in \mathbb{R}^{11}$ e máscara $m_t \in \{0,1\}^{11}$.
- Codificação inicial por MLP (por timestep):

$$e_t = \phi\left(W_e[x_t \,\|\, m_t] + b_e\right), \quad t = 1, \ldots, L$$

  onde $\phi(\cdot)$ é uma ativação não linear (ReLU/GELU).
- Incorporação explícita do tempo:

$$e'_t = \mathrm{MLP}\left([e_t \,\|\, t_t]\right)$$

  com t_t o timestamp normalizado do instante t.
- Representação latente da janela:
  - Obtida independentemente por timestep (sem estados recorrentes);
  - Integra informação temporal apenas via timestamps explícitos.
- Cabeças de saída (multi-tarefa):

$$\hat{x}_t = f_\theta(e'_t) \quad \text{(reconstrução / difusão)}$$
$$\hat{\tau} = g_\theta(e'_t) \quad \text{(tempo até mudança de estado)}$$

- Treinamento:
  - Otimizador Adam;
  - Mesmas funções de perda usadas nos modelos recorrentes.

Arquitetura 1: LSTM

- Entrada: janela temporal $X = (x_1, \ldots, x_L)$, $x_t \in \mathbb{R}^{11}$.
- Codificação temporal com LSTM unidirecional:

$$(h_t, c_t) = \mathrm{LSTM}(x_t, h_{t-1}, c_{t-1}), \quad t = 1, \ldots, L$$

- Representação da janela:

$$z = h_L \in \mathbb{R}^H$$

- Cabeças de saída (multi-tarefa):

$$\hat{y}_{\text{state}} = \mathrm{softmax}(W_s z + b_s)$$
$$\hat{\tau} = W_\tau z + b_\tau$$

- Otimização:
  - Otimizador Adam, minimizando perda conjunta (classificação de estado + regressão de tempo).

Arquitetura 2: GRU

- Mesma entrada: $X = (x_1, \ldots, x_L)$, $x_t \in \mathbb{R}^{11}$.
- Codificação temporal com GRU unidirecional:

$$h_t = \mathrm{GRU}(x_t, h_{t-1}), \quad t = 1, \ldots, L$$

- Representação da janela:

$$z = h_L \in \mathbb{R}^H$$

- Cabeças de saída:

$$\hat{y}_{\text{state}} = \mathrm{softmax}(W_s z + b_s)$$
$$\hat{\tau} = W_\tau z + b_\tau$$

- Otimização:
  - Otimizador Adam, com os mesmos hiperparâmetros do baseline Deep (learning rate, batch, etc.).

Arquitetura 3: BiGRU + GRU fuser

- Codificação da janela com BiGRU:

$$h_t^{\rightarrow}, h_t^{\leftarrow} = \mathrm{BiGRU}(x_t, h_{t-1}^{\rightarrow}, h_{t+1}^{\leftarrow})$$

$$z_k = [h_L^{\rightarrow} \parallel h_1^{\leftarrow}] \in \mathbb{R}^{2H}$$

  onde z_k é o embedding da k-ésima janela.
- Fusão temporal entre janelas com GRU fuser:

$$m_k = \mathrm{GRU}_{\mathsf{fuser}}(z_k, m_{k-1}), \quad k = 1, \ldots, K$$

  $m_K \in \mathbb{R}^{H_f}$ resume a história recente de janelas.
- Cabeças de saída a partir de m_K:

$$\hat{y}_{\mathsf{state}} = \mathrm{softmax}(W_s m_K + b_s)$$
$$\hat{\tau} = W_\tau m_K + b_\tau$$

- Otimização:
  - Adam sobre todos os parâmetros (BiGRU + GRU fuser + cabeças).

Arquitetura 4: BiLSTM + GRU fuser

- Codificação da janela com BiLSTM:

$$(h_t^{\rightarrow}, c_t^{\rightarrow}), (h_t^{\leftarrow}, c_t^{\leftarrow}) = \mathrm{BiLSTM}(x_t, \dots)$$

$$z_k = [h_L^{\rightarrow} \parallel h_1^{\leftarrow}] \in \mathbb{R}^{2H}$$

- Fusão temporal entre janelas com GRU fuser:

$$m_k = \mathrm{GRU}_{\mathsf{fuser}}(z_k, m_{k-1}), \quad k = 1, \dots, K$$

- Cabeças de saída:

$$\hat{y}_{\mathsf{state}} = \mathrm{softmax}(W_s m_K + b_s)$$
$$\hat{\tau} = W_\tau m_K + b_\tau$$

- Otimização:
  - Treino conjunto com Adam, mesma função de perda multi-tarefa.

Variações de treinamento - Otimização

- **BiLSTM (GRU Fuser) + Warmup & Scheduler**
  - Aumenta gradualmente a taxa de aprendizado no início do treino, evitando instabilidades e melhorando a convergência inicial.
  - `warmup_steps = 50`, `min_lr_factor = 0.01`
- **BiLSTM (GRU Fuser)+ RAdam**
  - Reduz a variância adaptativa do Adam nos primeiros passos, tornando o treinamento mais estável.
  - `optimizer_name = 'radam'`

Variações de treinamento — Regularização

- **BiLSTM (GRU Fuser) + Variational Dropout**
  - Regularização explícita no tempo, reduzindo overfitting sem quebrar dependências temporais.
  - `variational_dropout = 0.2`
- **BiLSTM (GRU Fuser) + Difusão (missingness)**
  - Regularização implícita via objetivos probabilísticos, forçando robustez a dados ausentes.
  - `lam = [0., 0.0, 0.0, 0.05, 0.0, 0.95]`, `rebuild=True`
- **BiLSTM (GRU Fuser) + LayerNorm**
  - Estabiliza as ativações internas, com efeito regularizante indireto.
  - `use_layernorm = True`
- **BiLSTM (GRU Fuser) + AdamW (Weight Decay)**
  - Regularização explícita dos pesos, melhorando generalização.
  - `optimizer_name = 'adamw'`, `weight_decay = 1e-4`

Configuração Experimental

- Divisão treino/validação/teste: 60% / 20% / 20%.
- Treinamento por 500 épocas, batch size 256.
- Paciênica de 50 épocas para early stopping.
- Função de perda multi-tarefa conforme descrito.
- Otimizador: Adam (exceto variações).
- Taxa de aprendizado inicial: 1×10^{-3}.
- Early stopping baseado em MSE de validação.
- Avaliação final no conjunto de teste.

Métricas utilizadas

- **ELBO** (Evidence Lower Bound):
  - Função objetivo usada no treinamento de modelos variacionais;
  - Equilibra qualidade de reconstrução e regularização via divergência KL.
- **MSE** (Mean Squared Error):
  - Mede o erro médio quadrático na predição do tempo até a próxima mudança;
  - Avalia precisão pontual das estimativas.
- **NLL** (Negative Log-Likelihood):
  - Avalia a qualidade probabilística das previsões;
  - Penaliza incertezas mal calibradas (sub ou superestimação).
- **Cobertura 90%** (`cov_90`):
  - Fração de observações reais contidas no intervalo preditivo de 90%;
  - Mede a calibração do modelo.
- **Largura do intervalo 90%** (`width_90`):
  - Largura média dos intervalos preditivos de 90%;
  - Reflete o compromisso entre precisão e incerteza (sharpness).

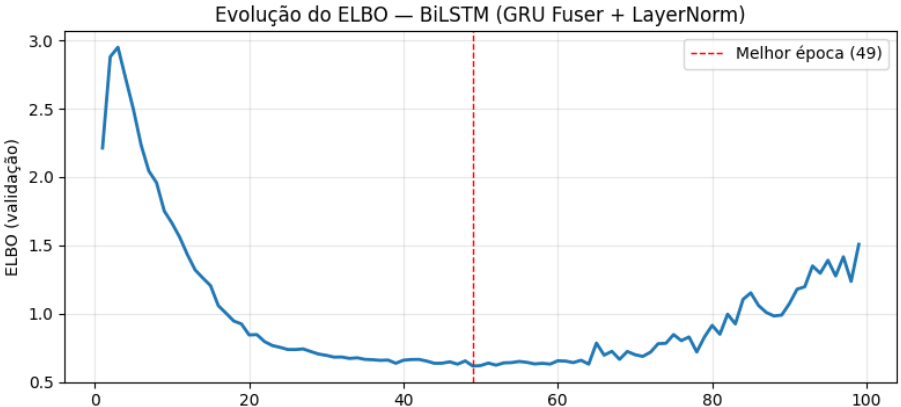
Resultados quantitativos

Modelo	ELBO ↑	MSE ↓	NLL ↓	Cov90 ↑	Width90 ↓
DEEP	1.217	0.693 ± 0.209	1.164	0.916	2.785
BiLSTM (GRU Fuser) LayerNorm	0.617	0.342 ± 0.140	0.514	0.907	2.281
BiLSTM	0.609	0.337 ± 0.133	0.527	0.901	2.269
BiLSTM (GRU Fuser) AdamW	0.601	0.341 ± 0.141	0.506	0.900	2.224
BiLSTM (GRU Fuser) Warmup/Sched.	0.598	0.354 ± 0.146	0.504	0.903	2.251
BiLSTM (GRU Fuser) RAdam	0.597	0.341 ± 0.139	0.511	0.898	2.234
BiLSTM (GRU Fuser) VarDrop 0.2	0.595	0.317 ± 0.119	0.506	0.898	2.240
BiGRU	0.589	0.335 ± 0.136	0.499	0.909	2.267
LSTM	0.588	0.332 ± 0.129	0.513	0.900	2.251
GRU	0.558	0.300 ± 0.108	0.498	0.905	2.252
BiLSTM (GRU Fuser) Difusão	0.518	0.243 ± 0.067	0.462	0.902	2.200

Tabela: Comparação de desempenho entre modelos e variações.

Curva de treinamento — Modelo campeão

- Evolução do ELBO ao longo das épocas para o modelo **BiLSTM (GRU Fuser + LayerNorm)**.
- Observa-se:
  - Convergência mais rápida;
  - Menor oscilação durante o treinamento;
  - Melhor estabilidade em comparação aos modelos sem normalização.



Discussão dos resultados

- Comparar quais modelos:
  - Obtiveram menor MSE e NLL;
  - Mantiveram boa cobertura com intervalos estreitos (trade-off).
- Avaliar impacto de:
  - Bidirecionalidade (BiLSTM/BiGRU);
  - Warmup + scheduler;
  - Otimizadores (Adam vs. RAdam vs. AdamW);
  - Variational dropout e difusão para missingness;
  - LayerNorm.
- Conectar com a operação:
  - O modelo é suficientemente responsivo antes de mudanças de estado?

Conclusões

- Implementação de um pipeline completo:
  - DataLoader, segmentação em estados, criação de janelas e labels.
- RNNs (especialmente BiLSTM/BiGRU) capturam bem a dinâmica do compressor.
- Técnicas de regularização e otimização (dropout, difusão, AdamW) influenciam precisão e calibração.
- Abre caminho para:
-
  - Integração com gêmeo digital;
  - Alarmes mais inteligentes de mudança de regime;
  - Estudos de degradação e manutenção preditiva.