

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

Fernanda Mickosz Villa Verde

CPE727 – Deep Learning

8 de dezembro de 2025

Sumário

- 1 Motivação e Objetivos
- 2 Base de dados e pré-processamento
- 3 Formulação do Problema
- 4 Arquiteturas e Variações
- 5 Configuração Experimental
- 6 Resultados
- 7 Discussão e Conclusões

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.
- Aprender uma **modelagem probabilística do tempo até a próxima mudança de estado**, estimando distribuições preditivas condicionadas ao histórico observado.

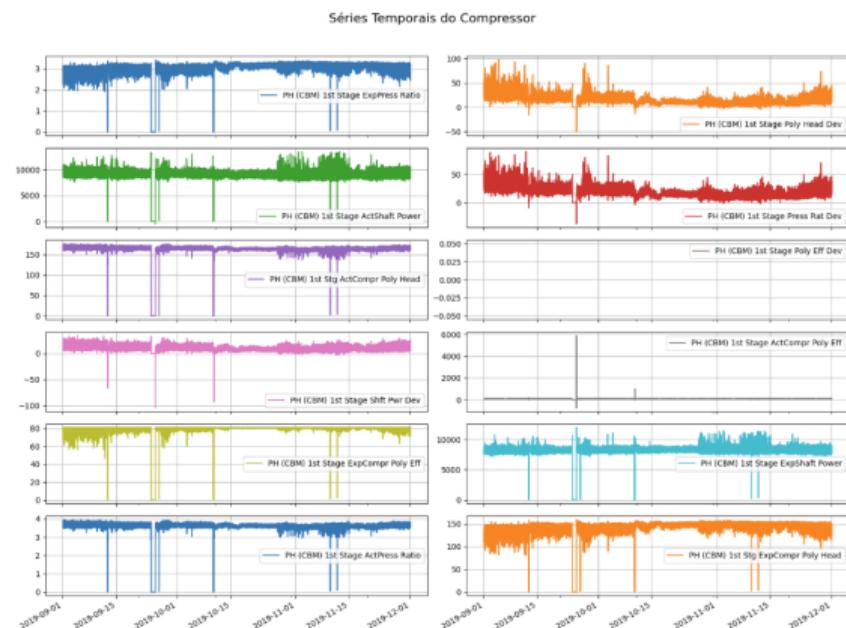
Comparar arquiteturas:

- 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 temporal multivariada: $x_t \in \mathbb{R}^d$, $t = 1, \dots, T$, com $d = 11$ sensores.
- Estados discretos: $s_t \in \{0, 1, 4\}$ (após fusão dos 6 estados iniciais).
- Definimos o **tempo até a próxima mudança de estado**:

$$\tau_t = \min\{\Delta > 0 : s_{t+\Delta} \neq s_t\}.$$

- Para cada janela de histórico $X = (x_{t-L+1}, \dots, x_t)$, condicionada ao estado atual s_t , queremos modelar a distribuição preditiva:

$$p(\tau | X, s_t).$$

- Na prática, o modelo produz, para cada estado $s \in \{0, 1, 4\}$, parâmetros $\mu_s(X)$ e $\sigma_s^2(X)$ que definem as **PDFs do tempo até mudança de estado**.

Assim, tratamos o problema como um **forecast probabilístico de tempo até evento**.

Função de custo

- Função de custo multi-tarefa:

$$\mathcal{L} = \lambda_m \mathcal{L}_{\text{miss}} + \lambda_v \mathcal{L}_{\text{vae}}$$

$$\mathcal{L}_{\text{miss}} = \text{BCE}(m, \hat{m}) \quad \text{com} \quad \hat{m} = \sigma(f_{\text{miss}}(h))$$

$\mathcal{L}_{\text{miss}}$: aprendizado explícito do padrão de dados ausentes.

$$\mathcal{L}_{\text{vae}} = \mathbb{E}_{q(z_\tau | x)} [-\log p(\tau | z_\tau)] + \beta \text{KL}(q(z_\tau) \| \mathcal{N}(0, I))$$

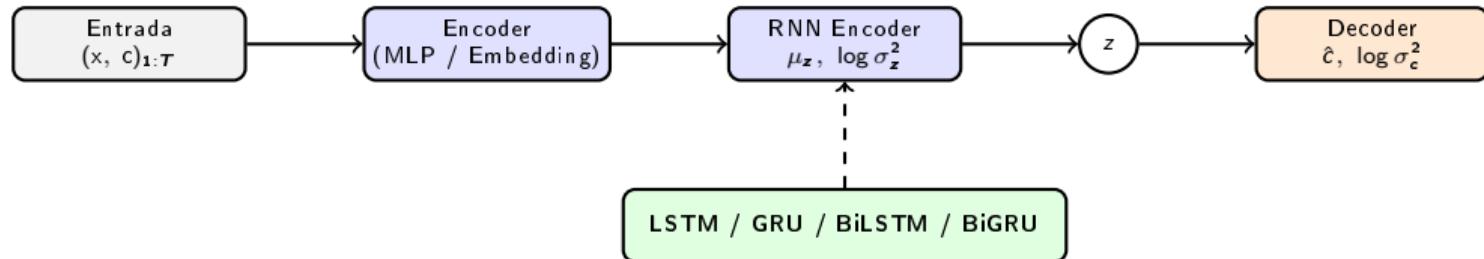
\mathcal{L}_{vae} : **modelagem probabilística do tempo até falha**, penalizando mudanças próximas e calibrando incerteza.

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

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

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

Arquitetura base e arquiteturas avaliadas



- **Arquitetura base:** Encoder temporal recorrente acoplado a um VAE latente.
- A entrada $X = (x, c)_{1:T}$ é codificada e processada pelo **RNN encoder**, que parametriza a distribuição latente $(\mu_z, \log \sigma_z^2)$.
- A partir da amostragem $z \sim q(z | X)$, o **decoder** estima a distribuição sobre a mudança futura, produzindo \hat{c} e sua incerteza associada $\log \sigma_c^2$.
- **Arquiteturas avaliadas:**
 - **TSDF_DEEP:** baseline feedforward (sem recorrência);
 - **TSDF_LSTM:** LSTM unidirecional e BiLSTM;
 - **TSDF_GRU:** GRU unidirecional e BiGRU.

Variações de treinamento em relação à arquitetura base

Arquitetura base fixa: Encoder temporal BiLSTM + VAE latente.

1. Variações de otimização

- **BiLSTM + Warmup & Scheduler:** Aumenta gradualmente a taxa de aprendizado no início e depois aplica um *scheduler* de decaimento, reduzindo instabilidades nas primeiras épocas e melhorando a convergência inicial.
- **BiLSTM + RAdam:** Substitui o Adam por **RAdam**, que corrige a variância adaptativa nos primeiros passos, tornando o treinamento mais estável.

2. Variações de regularização

- **BiLSTM + Variational Dropout:** Aplica *variational dropout* ($p = 0,2$) compartilhado no tempo, reduzindo overfitting sem quebrar dependências temporais.
- **BiLSTM + Difusão (missingness):** Acrescenta um objetivo probabilístico extra $\mathcal{L} = \lambda_m \mathcal{L}_{miss} + \lambda_v \mathcal{L}_{vae}$, forçando o modelo a ser robusto a *missing data* e a calibrar melhor incerteza.
- **BiLSTM + LayerNorm:** Aplica **LayerNorm** às ativações da BiLSTM, estabilizando as distribuições internas com efeito regularizante indireto.
- **BiLSTM + AdamW (L2/Weight Decay):** Usa **AdamW** com *weight decay* desacoplado ($\lambda = 10^{-4}$), acrescentando regularização L2 explícita nos pesos e melhorando generalização.

Configuração Experimental

- Divisão treino/teste: 80% / 20%.
- A quantidade de neurônios já foi validada anteriormente em outra disciplina
- Treinamento por 500 épocas, batch size 256.
- Paciência de 50 épocas para early stopping.
- Função de perda multi-tarefa conforme descrito.
- Otimização e Regularização tratadas como diferentes modelos.
- Otimizador: Adam (exceto variações).
- Taxa de aprendizado inicial: 3×10^{-4} .
- Early stopping baseado no NELBO do teste.
- Avaliação final no conjunto de teste.

- **NELBO** (Negative Evidence Lower Bound):

- Loss probabilística minimizada no treinamento;
- Maximiza implicitamente a verossimilhança (ELBO);
- Balanceia reconstrução e regularização latente (KL).
- $NELBO = \mathbb{E}_{q(z|x)}[-\log p(x | z)] + KL(q(z | x) \| p(z))$

- **NLL** (Negative Log-Likelihood):

- Generaliza o MSE ao modelar explicitamente a variância da distribuição predita;
- Penaliza erros grandes e variâncias mal calibradas (super ou subestimação de incerteza).

$$\text{NLL} = -\log p(x | \mu, \sigma^2) = \frac{(x - \mu)^2}{2\sigma^2} + \frac{1}{2} \log \sigma^2 + \text{cte}$$

Figuras de Mérito utilizadas

- **MSE** (Mean Squared Error):

- Erro médio de reconstrução das séries;
- Mede fidelidade gerativa.

- **Cobertura 90%** (cov_90):

- Mede a fração de amostras reais que caem dentro do intervalo preditivo teórico [5%, 95%];
- Avalia se o desvio padrão estimado produz uma **calibração probabilística consistente** com a cobertura nominal de 90%.

- **Largura do intervalo 90%** (width_90):

- Corresponde à largura teórica do intervalo [5%, 95%] derivado da distribuição predita;
- Quantifica a **sharpness** do modelo: intervalos menores indicam maior confiança, desde que a cobertura permaneça bem calibrada.

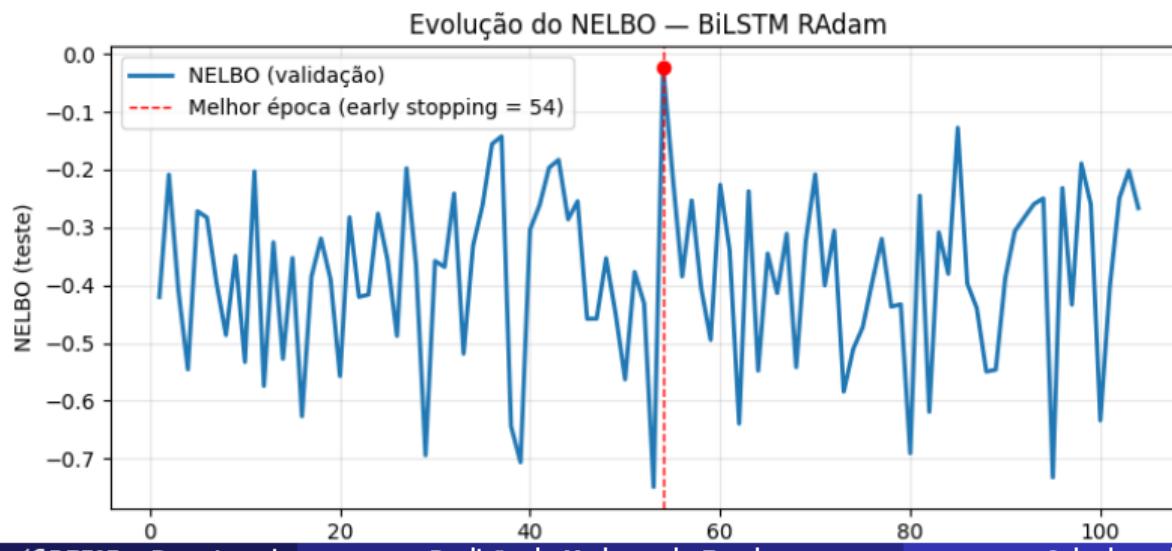
Resultados quantitativos

Modelo	NELBO ↓	MSE (micro) ↓	NLL ↓	Cov90 → 0.90	Width90 ↓
BiLSTM RAdam	0.023	0.002 ± 0.000	0.023	0.946	0.270
BiLSTM Warmup/Sched.	0.033	0.002 ± 0.000	0.033	0.946	0.270
LSTM	0.064	0.002 ± 0.000	0.063	0.937	0.270
BiLSTM	0.150	0.002 ± 0.000	0.150	0.920	0.270
BiGRU	0.203	0.001 ± 0.000	0.202	0.875	0.270
BiLSTM AdamW	0.230	0.001 ± 0.001	0.229	0.866	0.270
BiLSTM Difusão	0.329	0.002 ± 0.001	0.328	0.839	0.270
BiLSTM LayerNorm	0.383	0.054 ± 0.001	0.332	0.839	0.270
GRU	0.464	0.003 ± 0.001	0.463	0.812	0.270
BiLSTM VarDrop 0.2	0.992	0.002 ± 0.001	0.991	0.705	0.270
DEEP	8016.760	0.819 ± 0.011	33.711	0.250	0.270

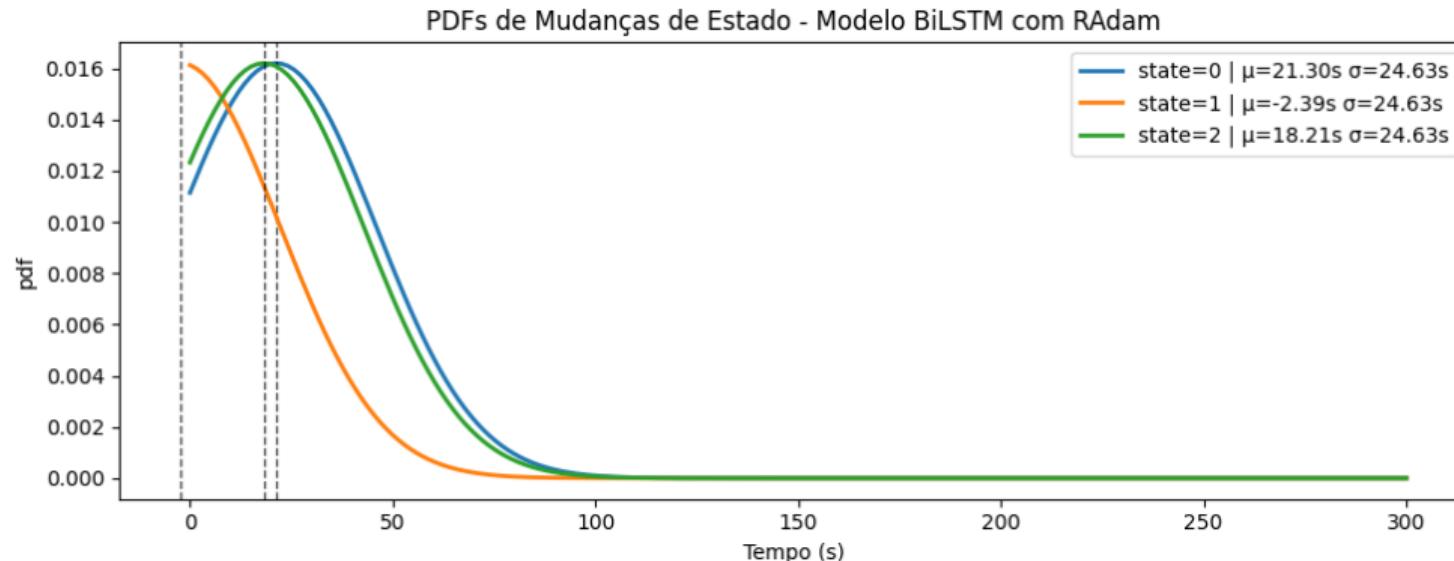
Tabela: Comparação de desempenho entre modelos e variações.

Curva de treinamento — Modelo BiLSTM RAdam

- Curva de NELBO bastante oscilatória, refletindo o desbalanceamento entre grupos e a raridade de mudanças de estado;
- Após as primeiras dezenas de épocas, o modelo passa a operar em um patamar de NELBO baixo, com mínimos recorrentes;
- A melhor época (**epoch 54**) atinge o menor NELBO entre as épocas avaliadas, sendo o melhor compromisso entre ajuste e complexidade para a tarefa de mudança de estado.



PDFs do tempo até mudança de estado



Distribuições preditivas do tempo até mudança de estado condicionadas ao histórico observado, ilustrando a modelagem probabilística e a assimilação de incerteza pelo modelo.

- PDFs do tempo até a mudança, condicionadas ao histórico $X_{1:t}$.
- Curvas por estado futuro, com médias distintas e incerteza associada.

Discussão dos resultados — Mudança de estado

- **Qualidade probabilística:**

- O BiLSTM RAdam apresenta o menor NELBO, indicando melhor ajuste probabilístico global;
- Bons valores de NLL e MSE micro confirmam alta precisão na predição de mudanças de estado raras.

- **Calibração das incertezas:**

- Cobertura próxima de 0.90 nos melhores modelos indica boa calibração probabilística;
- Width90 constante sugere intervalos estáveis e comparáveis entre arquiteturas.

- **Impacto arquitetural:**

- Modelos bidirecionais superam LSTM/GRU unidirecionais;
- Em cenários fortemente desbalanceados, o principal desafio é a **otimização sob gradientes raros**, e não o overfitting;
- Otimizadores adaptativos (RAdam, warmup) estabilizam o treinamento e facilitam a captura de eventos de mudança raros, enquanto regularização excessiva tende a diluir esses sinais.

Conclusões — Mudança de estado

- A predição de mudanças de estado é um problema fortemente desbalanceado e sensível à calibração probabilística.
- Arquiteturas recorrentes probabilísticas são essenciais:
 - Capturam dependências temporais longas;
 - Modelam explicitamente incertezas associadas ao evento de transição.
- O **BiLSTM RAdam** é o modelo mais adequado:
 - Menor NELBO entre os modelos avaliados;
 - Boa calibração ($\text{Cov90} \approx 0.95$) e baixo erro micro.
- Esses resultados indicam potencial para:
 - Detecção antecipada de mudanças de regime;
 - Suporte a sistemas de monitoramento e alerta industrial.

Referências

- Kingma, D. P.; Welling, M. *Auto-Encoding Variational Bayes*. ICLR, 2014.
- Hochreiter, S.; Schmidhuber, J. *Long Short-Term Memory*. Neural Computation, 1997.
- Cho et al. *Learning Phrase Representations using RNN Encoder–Decoder*. EMNLP, 2014.
- Ho, J.; Jain, A.; Abbeel, P. *Denoising Diffusion Probabilistic Models*. NeurIPS, 2020.
- Che et al. *Recurrent Neural Networks for Multivariate Time Series with Missing Values*. Sci Rep, 2018.