UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Sistema de Detecção e Categorização de Falhas de Infraestrutura

Diogo Moura Santos

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

Diogo Moura Santos

Sistema de Detecção e Categorização de Falhas de Infraestrutura

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Kelton Costa

Versão original

São Carlos

2024

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Santos, Diogo Moura
S237s Sistema de Detecção e Categorização de Falhas de Infraestrutura / Diogo Moura Santos; orientador Kelton Costa. -- São Carlos, 2024.
35 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2024.

1. BIG DATA. 2. PROCESSAMENTO DE TEXTO. 3. PROCESSAMENTO DE LINGUAGEM NATURAL . I. Costa, Kelton, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Diogo Moura Santos

Infrastructure Failure Detection and Categorization System

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Kelton Costa

Original version

São Carlos

2024

Este trabalho é dedicado aos alunos da USP, como uma contribuição das Bibliotecas do Campus USP de São Carlos para o desenvolvimento e disseminação da pesquisa científica da Universidade.

AGRADECIMENTOS

Agradeço ao meu orientador, Prof. Kelton Costa, pelo apoio e orientação ao longo deste projeto. Agradeço também à minha família e amigos pelo incentivo constante e à Universidade de São Paulo por proporcionar os recursos necessários para a realização deste trabalho.

"O estudo, a busca da verdade e da beleza são domínios em que nos é consentido sermos crianças por toda a vida."

Albert Einstein

RESUMO

SANTOS, D. **Sistema de Detecção e Categorização de Falhas de Infraestrutura**. 2024. 39 p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

Devido a uma crescente necessidade de desenvolver sistemas **eficientes** para a **detecção e categorização** de falhas, **supervisão e monitoramento** em infraestruturas de TI ao longo das últimas décadas. Isso se deve à grande e complexa constituição dos **ambientes de rede**, com a distribuição dos sistemas de computadores. **Falhas** podem causar grandes **perdas** em **recursos** e prejudicam muito, a reputação das organizações. Neste contexto, o **monitoramento tradicional** muitas vezes **não é sensível o suficiente** para captar as **sutilezas e inter-relações** entre diferentes tipos de falhas, apontando para a necessidade de abordagens mais sofisticadas e ativas para garantir a resiliência e eficiência dos sistemas. As **tecnologias de Processamento de Linguagem Natural** (PLN) na análise de logs visam **reduzir os tempos de inatividade** e aumentar a disponibilidade dos serviços de TI por meio de um sistema automatizado para detecção e classificação de falhas nos servidores. O **PLN** tem aplicações para **derivar padrões e anomalias dos registros de logs de maneira automatizada**, de modo que as informações necessárias possam ser facilmente extraídas e, assim, beneficiar ainda mais o processo de modelagem preditiva e tornar o fluxo de trabalho de detecção de falhas e sua categorização **por meio de aprendizado de máquina eficiente**. É um grande avanço no campo de pesquisas recentes, que enfatizou a importância das técnicas de PLN e dos algoritmos de aprendizado de máquina na **prevenção e detecção precoce** de **incidentes** nos sistemas de TI.
Palavras-chave: Classe USPSC. Tese. Dissertação. Trabalho de conclusão de curso (TCC). Relatório. eficientes, detecção, categorização, supervisão, monitoramento, ambientes, rede, Falhas, perdas, recursos, tradicional, sutilezas, inter-relações, tecnologias, Processamento, Linguagem, Natural, reduzir, tempos, inatividade, PLN, derivar, padrões, anomalias, por, meio, aprendizado, máquina, eficiente, prevenção, detecção, precoce, incidentes.

ABSTRACT

SANTOS, D. **Infrastructure Failure Detection and Categorization System**. 2024. 39 p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

There has been a growing need to develop **efficient** systems for **detection and categorization** of faults, **observance and monitoring** in IT infrastructures over the last few decades. This is due to the large and complex constitution of **network environments**, with the distribution of computer systems. **Failures** can cause large **losses** both in **resources** and in the organizations' reputation. In this context, **traditional monitoring** is often **not precise enough** to capture the **subtleties and interrelationships** between different types of failures, pointing to the need for more sophisticated and active approaches to ensure the resilience and efficiency of systems. **Natural Language Processing** (NLP) technologies in log analysis aim to **reduce downtime** and increase the availability of IT services through an automated system for detecting and classifying server failures . **PLN** has applications to **derive patterns and anomalies from log records in an automated manner** so that the necessary information can be easily extracted and thus further benefit the predictive modeling process and make the fault detection workflow and its categorization **through efficient machine learning**. It is a major advance in the field of recent research, which has emphasized the importance of NLP techniques and machine learning algorithms in **prevention and early detection** of **incidents** in IT systems.

Keywords: USPSC class. Thesis. Dissertation. Course completion work (TCC). Report. Efficient, detection, categorization, observance, monitoring, environments, network, Failures, losses, resources, traditional monitoring, subtleties, inter-relationships, technologies, Natural Language Processing, reduce, downtime, NLP, derive, patterns, anomalies, through, machine learning, efficient, prevention, early detection, incidents.

LISTA DE FIGURAS

Figura 1 – Diagrama Abstrato do Processo 30
Figura 2 – Arquitetura LogBERT (Guo, 2021), adaptado pelo autor. 31
Figura 3 – Mapa de Calor de Correlação 33
Figura 4 – Dispersão entre o Tamanho do Log e o Número de Tokens 34
Figura 5 – Dispersão entre o Tamanho do Log e o Número de Tokens 35
Figura 6 – Logs do processo de Treinamento 36

LISTA DE TABELAS

Tabela 1 – Resultados experimentais com diferentes métodos em logs de Windows. 36
Tabela 2 – Estatísticas do conjunto de dados para Windows 37

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
IBGE	Instituto Brasileiro de Geografia e Estatística
USP	Universidade de São Paulo
USPSC	Campus USP de São Carlos
TI	Tecnologia da Informação
PNL	Processamento de Linguagem Natural
GRU	Gated Recurrent Unit
RNN	Recurrent Neural Network
BERT	Bidirectional Encoder Representations from Transformers
CPU	Central Processing Unit
GPU	Graphics Processing Unit
VRAM	Video Random Access Memory
TF-IDF	Term Frequency-Inverse Document Frequency
NLTK	Natural Language Toolkit
SO	Sistema Operacional
NLP	Natural Language Processing

SUMÁRIO

1 INTRODUÇÃO 27
1.1 Introdução e Contextualização 27
1.2 Justificativa e Motivação 27
1.3 Objetivos 27

2 FUNDAMENTAÇÃO TEÓRICA 28

3 METODOLOGIA 29
3.1 Coleta de Dados 29
3.2 Preparação e filtragem dos Dados 29
3.3 Pré-processamento de Dados 29
3.4 Treinamento do Modelo 29
3.5 Teste com validação cruzada 10 fold; Feedback 30
3.6 Diagrama do Processo 30
3.7 BERT e Logs 30
3.8 Exemplo de Estrutura LogBERT 31

4 AVALIAÇÃO EXPERIMENTAL 33

5 CONCLUSÃO 38

Referências 39

1 INTRODUÇÃO

1.1 Introdução e Contextualização

No contexto atual, os ambientes em rede e as infraestruturas informáticas distribuídas assumiram proporções e complexidade tão grandes que o estabelecimento de tais sistemas, permitindo o seu bom funcionamento e manutenção eficaz, representa um desafio absolutamente substancial na era digital. Tais interrupções, que podem surgir de falhas na rede, software e hardware, podem resultar em efeitos de longo alcance. Estas podem atingir níveis como a paralisia de serviços críticos, impactos financeiros massivos e perda de reputação corporativa.

1.2 Justificativa e Motivação

As abordagens convencionais de monitoramento, muitas vezes ficam aquém da capacidade de detectar padrões e correlações entre vários tipos de falhas com maior nuance, o que poderia comprometer os pilares de disponibilidade, integridade e confidencialidade dos dados. A crítica proativa à identificação surge neste contexto em servidores suscetíveis a falhas e em ambientes altamente críticos, conferindo resiliência e eficiência às infraestruturas de tecnologias de informação.

Nesta perspetiva, a identificação proativa de servidores vulneráveis a falhas ou ambientes de alta prioridade onde a resiliência e a eficiência devem ser garantidas nas infraestruturas de TI. O principal objetivo deste projeto é desenvolver e propor uma nova solução através do desenvolvimento de uma abordagem inovadora que possibilite a categorização e identificação de falhas na infraestrutura de TI com base nos logs do servidor através de Processamento de Linguagem Natural (PNL).

Considerando que se pretende maximizar o processo de automatização, neste sentido, a iniciativa deverá visar a automatização de todo o processo descrito. Para maior identificação e classificação de erros dentro dos servidores, deve-se contar com o fato de serem processos lentos, manuais e que necessitam de automação para terem maior agilidade e eficácia no processo de resolução de problemas, e assim evitar paradas ou diminuí-las.

1.3 Objetivos

Atualmente, uma das melhores ferramentas para detecção de anomalias tem sido a PNL, com o uso de Redes Neurais Transformers, proporcionando melhorias nas áreas de segurança e a resiliência de infraestruturas computacionais. Um exemplo disso, tem sido a aplicação do modelo de Machine Learning BERT, que tem excelentes resultados. O trabalho de Guo; Yuan; Wu, 2021 por exemplo, apresentou o LogBERT, que utiliza a

arquitetura do BERT para identificar padrões em sequências de logs e detectar desvios. Esse modelo superou métodos tradicionais, como SVM e PCA, destacando o poder dos transformadores em capturar a natureza sequencial e contextual dos dados de logs, essencial para identificar falhas em TI de maneira mais rápida e precisa.

Além disso, o estudo de Ryciak; Wasielewska; Janicki, 2022 elucida como as arquiteturas de memória usando redes neurais recorrentes (RNN), aplicando-as com sucesso na detecção de anomalias, detectando padrões recorrentes. As técnicas de PLN não somente melhoram a detecção de anomalias, mas também aumentam a eficiência na categorização de falhas. Por isso, este trabalho visa implementar uma solução baseada no modelo BERT, que utilize menos recursos de hardware e tempo tanto para treino quanto para inferência, para ser instrumentalizada para a detecção e categorização de falhas de infraestruturas de TI, com o intuito de superar os modelos tradicionais e realizar uma proposta mais efetiva para certificar a segurança de sistemas informacionais.

2 FUNDAMENTAÇÃO TEÓRICA

É de substancial importância identificar proativamente, falhas de sistemas técnicos informacionais, garantindo os pilares de integridade, disponibilidade e confidencialidade dos dados em uma infraestrutura informacional. Métodos de detecção preditivos possibilitam o uso de dados históricos e aprendizado de máquina, para gerar regras e inferências que previnem falhas. Acelerando até mesmo o processo de remediação, evitando assim, danos mais abrangentes. A combinação desses fatores e inteligência, gera resiliência estrutural contra uma parcela substancial de ataques conhecidos e falhas operacionais comuns, aumentando a eficiência dos sistemas analisados por tal ferramenta.

2.1 Benefício de análise de logs utilizando PNL: Essa abordagem contribui para uma detecção mais eficiente e precisa de falhas de infraestrutura de TI. Além disso, a análise de logs com PNL abre espaço para a criação de modelos preditivos, dando lugar a técnicas de aprendizado de máquina que permitirão melhoria na detecção e espaço para aprendizado das melhores formas de classificação de falhas que poderão levar a uma análise mais ágil e precisa dos dados, apontando para soluções contextuais e automações escaláveis.

2.2 Estado a Arte:

Por exemplo uma pesquisa recente publicada no Journal of Optical Communications and Networking discute como algoritmos de aprendizado de máquina têm sido aplicados à previsão de falhas de rede e propõe um sistema de última geração para detecção precoce de padrões anômalos Rafique; Velasco, 2018 Além disso, o artigo da Optics Express discute como as técnicas de PNL podem ser usadas para explorar melhor os logs do servidor para a detecção de eventos mais relevantes para a segurança Ryciak; Wasielewska; Janicki, 2022 Estes trabalhos elucidam que a eficácia preditiva Du *et al.*, 2017a previsão de falhas e eventos adversos que podem ocorrer em sistemas de TI também é alcançável graças aos métodos de Data Science. Yogatama *et al.*, 2018

O presente projeto, portanto, tenta contribuir para esta área específica de estudo com um sistema avançado de identificação e categorização de falhas dentro de infraestruturas de TI. O método faz o uso de técnicas de aprendizado de máquina e também emprega a validação cruzada (10-fold cross-validation), onde o empenho de diferentes modelos é avaliado para determinar a melhor abordagem em termos de predições e análises.

3 METODOLOGIA

A metodologia envolveu a coleta de logs em formato bruto de um repositório público pensado especificamente para alimentar projetos que fazem o uso de Machine learning. Os dados foram limitados a um espaço temporal de 1 mês de coleta para redução da amostra, os logs foram filtrados, normalizados e em seguida tokenizados para serem transformados em vetores numéricos usando a técnica transformers BERT. O treino foi avaliado com métricas de precisão, recall, e f1-score, utilizando uma infraestrutura híbrida de CPUS e GPUs.

3.1 Coleta de Dados

Fonte de Dados: Open-source logs datasets estão disponíveis em repositórios de publicação como Loghub (https://github.com/logpai/loghub), proejto pelo User LOGPAI (https://github.com/logpai) no Github o escopo do treinamento incial foram os logs do SO para servidores (Linux e Windows). Os datasets foram truncados para 1 mês de logs, para que pudessem se enquadrar nos recursos computacionais disponíveis. O arquivo .log tem uma dimensão de 6.37GB, com 12,277,449 linhas de logs apenas pré filtrados.

3.2 Preparação e filtragem dos Dados

Formatação dos Logs brutos em um formato que adequado para análise; Normalização e remoção e limpeza de Dados; Eliminação de ruído e inconsistências e redundâncias (linhas redundantes/repetidas/vazias;

3.3 Pré-processamento de Dados

Tokenização: Dividir os logs em tokens significativos, como palavras ou frases ou componentes sintáticos de palavras; Normalização dos tokens: converter pra uma forma padrão, lowercase, sem pontuação; Vectorização: Foram testados os métodos TF-IDF, Word2Vec e embeddings via BERT (Bidirectional Encoder Representations from Transformers);

3.4 Treinamento do Modelo

Técnicas de embedding para transformar valores textuais em vetores numéricos. Usados para treinar o modelo transformers. Este método resolveu muitos problemas que eram comuns do GRU RNN, com eficiência computacional muito boa para o tema de identificação de padrões contextuais que envolvem relações bidirecionais e recorrentes, como

é o caso de análise de logs. Ferramentas/dependências como o Scikit-learn, TensorFlow, NLTK, Pandas foram usadas para desenvolver e avaliar o modelo.

3.5 Teste com validação cruzada 10 fold; Feedback

A validação cruzada 10-Fold foi aplicada para testar a robustez do modelo de classificação. Neste método, os dados foram divididos em 10 partes (folds), onde o modelo foi treinado em 9 dessas partes e testado na parte restante. Esse processo foi repetido de forma rotativa para cobrir todos os folds. Essa abordagem permitiu uma avaliação mais completa do modelo, minimizando o viés nas métricas de desempenho, como precisão, recall e F1-Score.

3.6 Diagrama do Processo

A captação do fluxo de trabalho completo, da coleção à PNL até ated feedback do DB. Cada etapa é essencial para a eficácia do sistema.

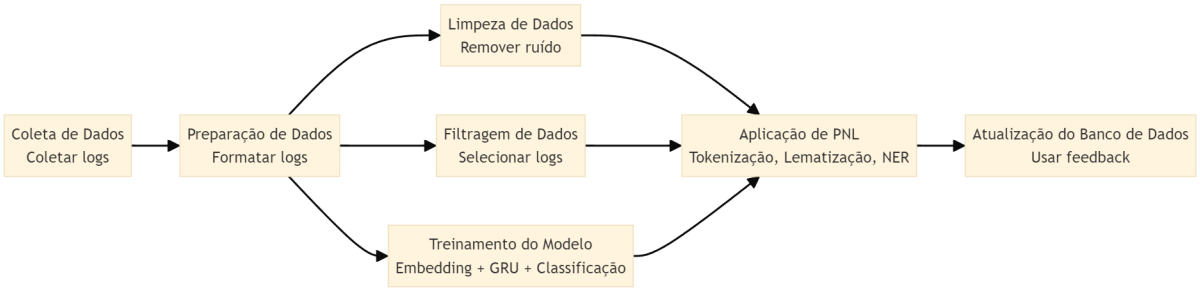

Figura 1 – Diagrama Abstrato do Processo

Conforme a Figura 1, o processo começa com a coleta de dados, seguida pela preparação e filtragem dos dados para garantir a qualidade. A preparação é segmentada em três etapas: limpeza de dados, normalização e tokenização. Essas etapas são fundamentais para transformar os logs brutos em uma forma que possa ser utilizada pelo modelo de aprendizado de máquina para detecção e categorização de falhas.

3.7 BERT e Logs

O BERT foi desenvolvido por Devlin *et al.*, 2019 um modelo de linguagem que faz o uso de transformers para capturar o contexto bidirecional de palavras em uma sentença, o que é adaptado por LogBERT para analisar sequencias de Logs.

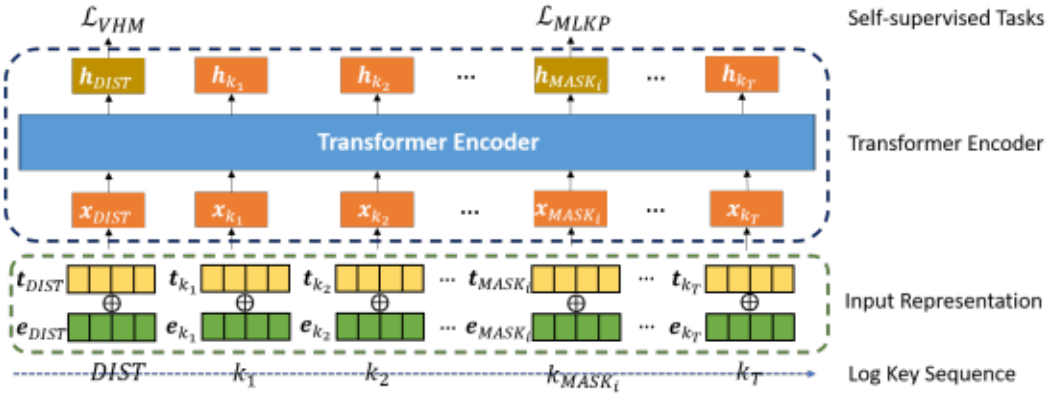

Figura 2 – Arquitetura LogBERT (Guo, 2021), adaptado pelo autor.

3.8 Exemplo de Estrutura LogBERT

Conforme ilustrado na Figura 2, a arquitetura LogBERT Guo; Yuan; Wu, 2021 utiliza técnicas de BERT para capturar o contexto das sequências de logs, facilitando a detecção de anomalias.

1. **Predição de Chaves de Logs Mascaradas:** Essa Tarefa consiste em gerar embeddings aleatórias, com o intuito de prever as sequências. Isso força o modelo a entender o contexto completo das sequências;

   Fórmula 1: Softmax para Predição de Chave de Log Mascarada

   $$y_{\text{pred}} = \text{Softmax}(W \cdot h + b)$$

   - **Softmax**: Uma função que transforma números em probabilidades.
   - **W**: Pesos que o modelo ajusta para melhorar suas predições.
   - **h**: Representação da chave de log mascarada (informação que o modelo tem sobre ela).
   - **b**: Vieses que ajudam a ajustar a predição.

2. **Minimização do Volume da Hiperesfera (VHM):** Baseada no Deep SVDD, esse módulo concentra sequências normais em um espaço de embedding, para treinar o modelo, minimizando o volume da hiperesfera que as contém, reduzindo os parâmetros, e assim, facilitando a distinção entre embeddings normais e anômalos;

Fórmula 2: Função de Perda para Predição de Chave de Log Mascarada

$$L_{\text{MLKP}} = -\frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{M} y_{\text{real}} \cdot \log(y_{\text{pred}})$$

- **LMLKP**: A função de perda que o modelo tenta minimizar. Quanto menor, melhor o modelo está performando.
- **N**: Número total de sequências de log usadas para treinamento.
- **M**: Número total de chaves de log mascaradas em cada sequência.
- **y real**: A chave de log real que queremos prever.
- **log(y_{pred})**: O logaritmo da probabilidade prevista pelo modelo para a chave de log real. O logaritmo penaliza mais fortemente os erros.

4 AVALIAÇÃO EXPERIMENTAL

Na análise exploratória realizada, existem três gráficos que nos fornecem uma visão detalhada sobre os dados de logs utilizados.

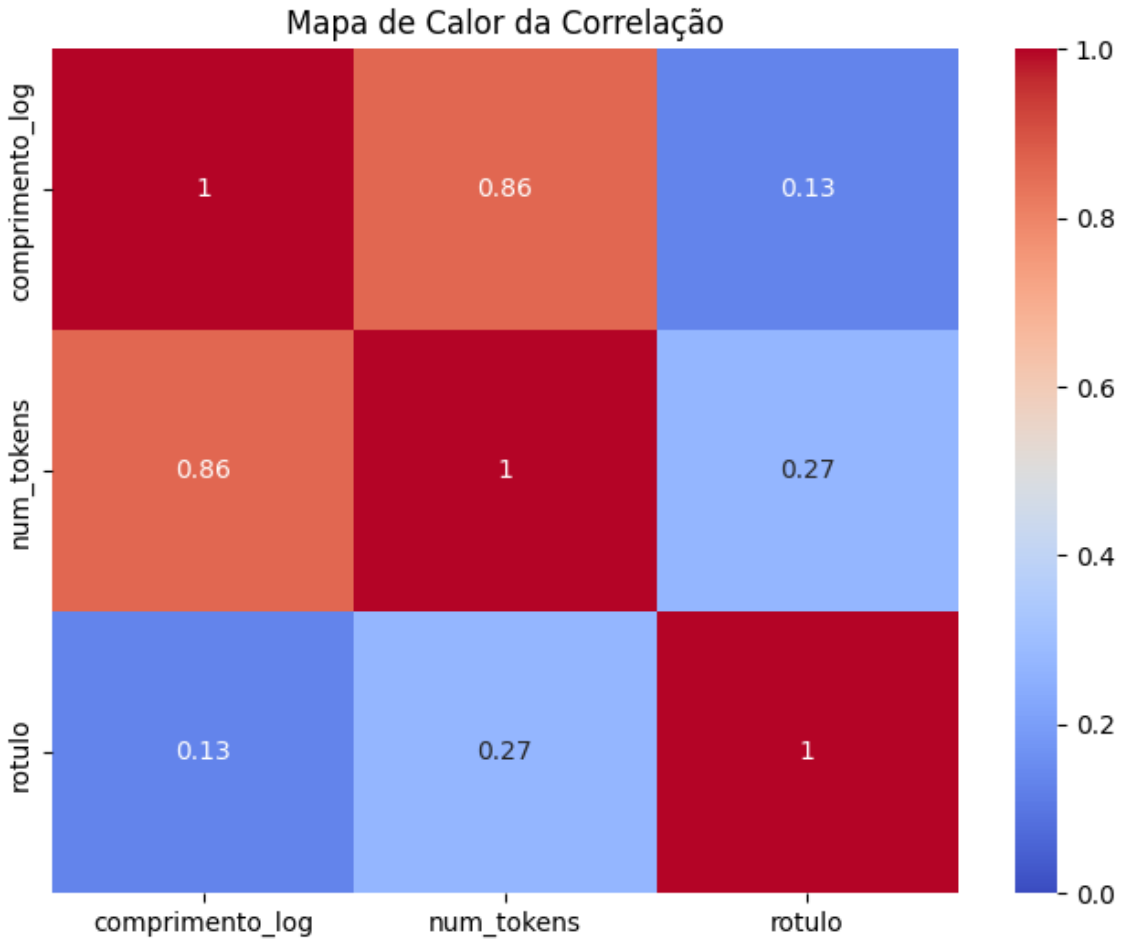

Figura 3 – Mapa de Calor de Correlação

A Figura 3: **Mapa de Calor de Correlação** mostra uma correlação forte (0,86) entre o comprimento dos logs e o número de tokens, como esperado, tendo em vista que são diretamente proporcionais. À medida que o comprimento dos logs aumenta, o número de tokens também cresce. E, todavia, é invalidada a hipótese que correlaciona o tamanho do log a erro/anomalia não há correlação significativa entre esses atributos e o rótulo de anomalia com 0 indicando normal e 1 indicando anomalia, sugerindo que o tamanho do log e o número de tokens não influenciam diretamente a detecção de anomalias.

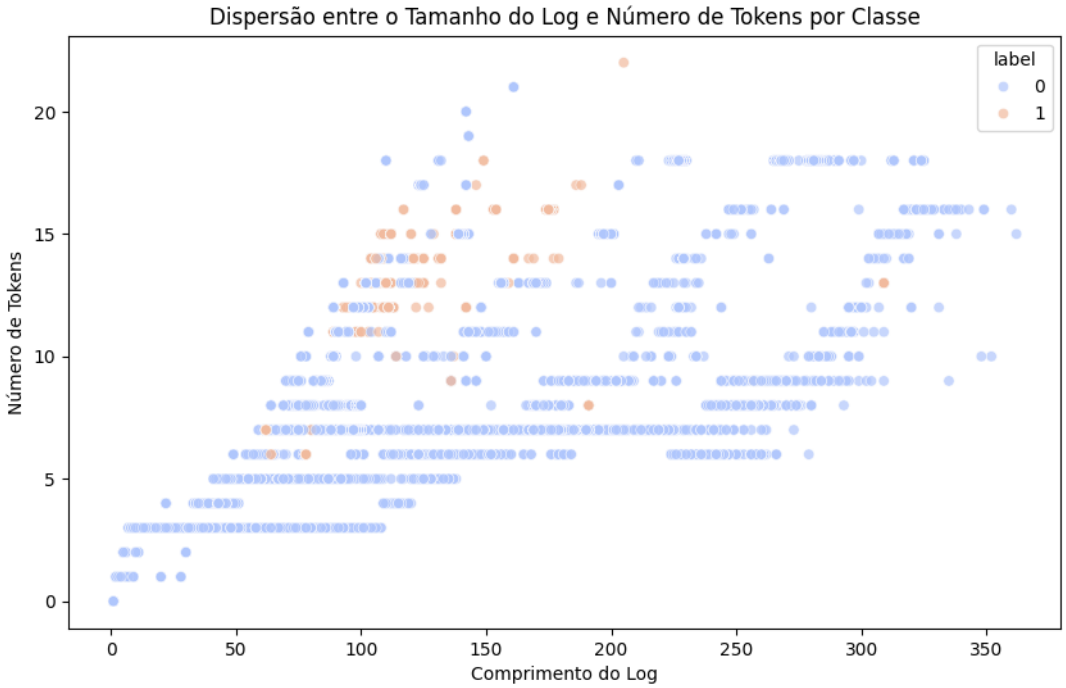

Figura 4 – Dispersão entre o Tamanho do Log e o Número de Tokens

A Figura 4 mostra a dispersão entre o tamanho do log e o número de tokens, diferenciando os logs normais e anômalos. Notamos que, embora haja alguma sobreposição, logs anômalos (em laranja) tendem a ter mais tokens e comprimentos maiores em relação aos logs normais (em azul). Isso indica que anomalias podem estar associadas a logs mais verbosos, mas não exclusivamente, reforçando a necessidade de técnicas mais sofisticadas, como o uso de modelos de linguagem para capturar esses detalhes.

Fórmula 1: Softmax para Predição de Chave de Log Mascarada

$$y_{\text{pred}} = \text{Softmax}(W \cdot h + b)$$

- **Softmax**: Uma função que transforma números em probabilidades.
- **W**: Pesos que o modelo ajusta para melhorar suas predições.
- **h**: Representação da chave de log mascarada (informação que o modelo tem sobre ela).
- **b**: Vieses que ajudam a ajustar a predição.

E além disso, a função softmax foi aplicada aos logits (resultados da execução do modelo BERT) para que o sistema de predição ficasse mais acurado, e ajudasse a ajustar os hiperparâmetros também, para melhor performance do modelo, sem muito custo

computacional adicionado envolvido. Os hiperparâmetros associados à Softmax incluem os pesos W e o viés b, ajustados durante o treinamento do modelo. Esses valores impactam diretamente a predição final e são otimizados para minimizar a função de perda.

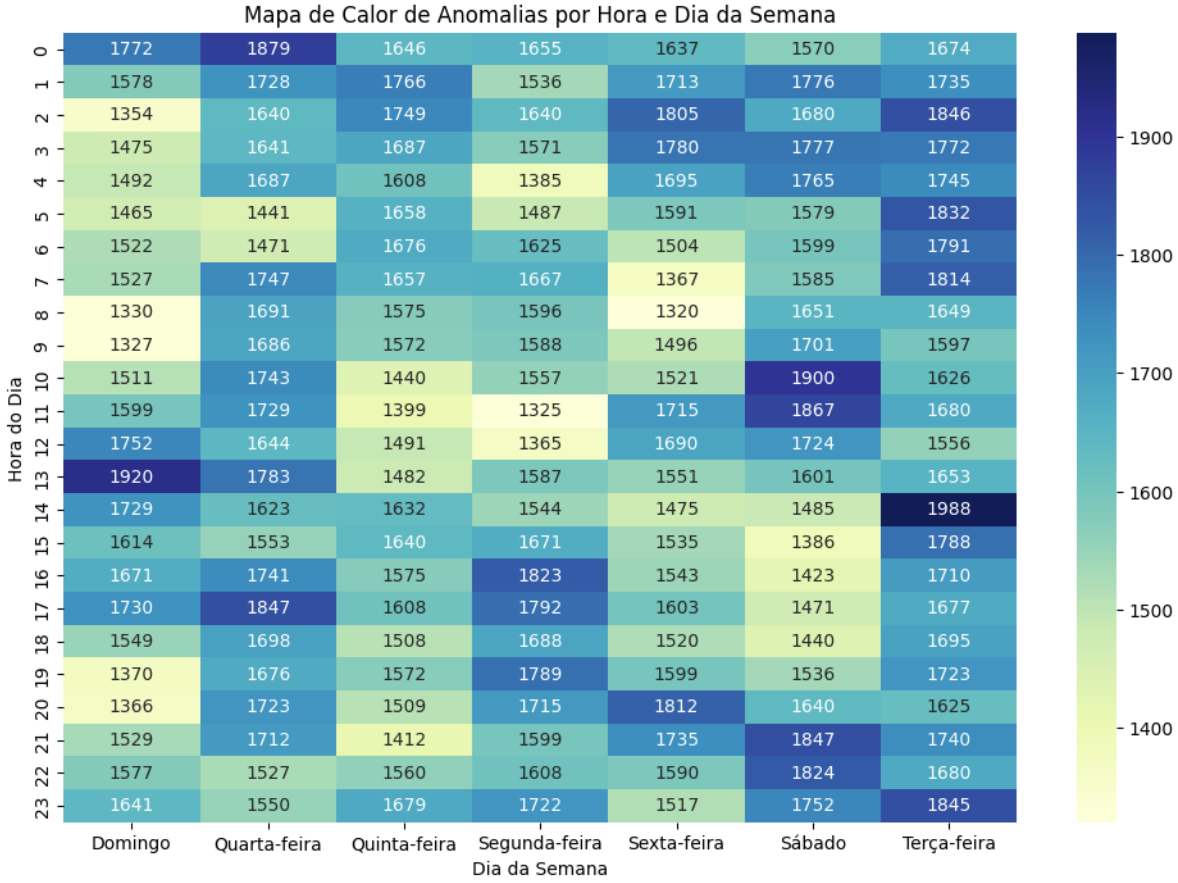

Figura 5 – Dispersão entre o Tamanho do Log e o Número de Tokens

Na Figura 5: Com a predição, podemos usar a contagem de anomalias e consolidar **Mapa de Calor de Correlação por Hora e Dia da Semana** revela padrões interessantes, com um pico significativo de anomalias às 14h de terça-feira (1988 ocorrências) e às 10h de sábado (1900 ocorrências). Esses horários específicos podem indicar eventos críticos ou falhas recorrentes no ambiente de TI, sugerindo a necessidade de uma investigação mais aprofundada.

Além disso, observa-se uma tendência geral de aumento nas anomalias durante o horário comercial, especialmente nas tardes de segunda e terça-feira, enquanto há uma redução durante a madrugada. Esses padrões indicam que o sistema pode estar sob maior estresse nesses períodos, o que reforça a importância de um monitoramento cuidadoso e ajustes na infraestrutura durante esses horários críticos.

Esses gráficos e análises mostram a relevância de se utilizar uma abordagem combinada entre pré-processamento avançado (como tokenização) e modelagem de linguagem

para detecção de anomalias em logs, uma vez que a simples análise de correlação não captura toda a complexidade dos dados.

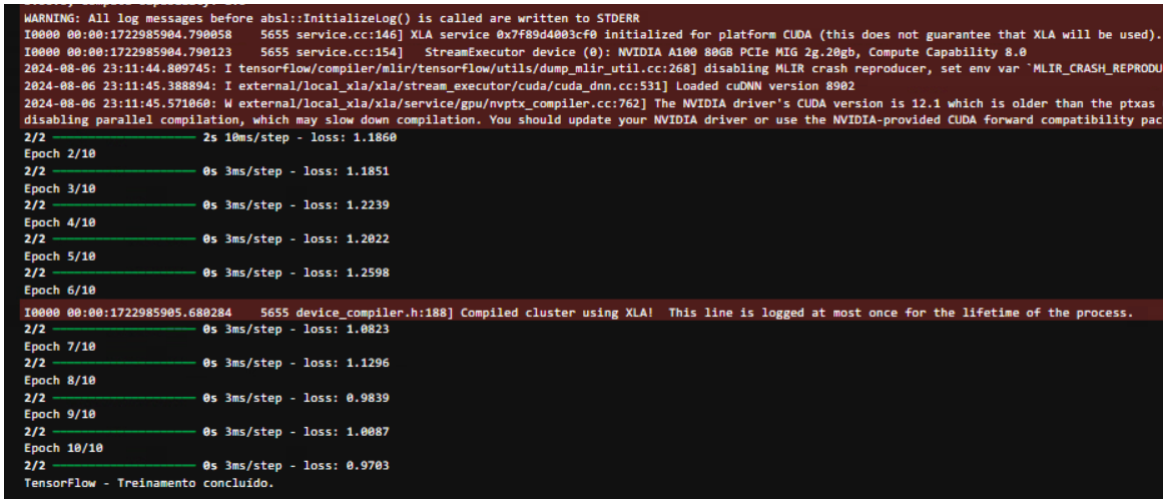

Figura 6 – Logs do processo de Treinamento

A figura 6 mostra alguns logs extraídos depois do checkpoint ser treinado por 10 epochs para o 10-Fold.

O treinamento do modelo aconteceu durante cerca de 2 meses levando aproximadamente 548 horas, usando uma infraestrutura mista entre CPU (Intel(R) Xeon(R) CPU E5-2680 v4) para tarefas/algoritmos que fazem uso intensivo de GPU, e usando aproximadamente 32GB de RAM. O treinamento e inferência dos modelos/checkpoints que utilizam mais recursos de GPU foi feito utilizando cloud GPU, com 4 x NVIDIA A100 totalizando 20GB de VRAM.

Na Tabela 1 estão os resultados de experimentos utilizando diferentes métodos. Os parâmetros avaliados são, Recall e F1-Score. O método LogBERT apresenta o melhor desempenho geral, com um F1-Score de 82.33, destacando-se.

Tabela 1 – Resultados experimentais com diferentes métodos em logs de Windows.

Método	**Precisão**	**Recall**	**F-1 score**
SVM	2.54	100.00	4.95
PCA	5.89	100.00	11.12
Random Forest	53.60	69.41	60.55
Detecção de Falhas	61.20	69.41	**65.04**
DeepLog	85.44	69.49	76.60
LogBERT	87.02	78.10	82.33

O modelo do Sistema de Detecção de falhas, com apenas 12,277,449 entradas e 10 folds (uma para validação e outras 9 para treinamento) obteve um F1-Score de 65.04, o que é significativo, devido ao pouco tempo de treino e aos recursos computacionais limitados.

Tabela de Log para Windows

Tabela 2 – Estatísticas do conjunto de dados para Windows

Dataset	**# Linhas**	**# Anômalos**	**# Logs Únicos**
Windows	12,277,449	65,950	57

A Tabela 2 apresenta as estatísticas de um conjunto de dados de logs para o sistema Windows. O conjunto de dados contém 12.277.449 mensagens de log, das quais 65.950 são anomalias. Há 57 chaves de log únicas; As chaves de log únicas representam a diversidade dos eventos nos logs; Quanto maior a variedade dessas chaves, maior a complexidade do sistema monitorado, o que é essencial para a detecção de anomalias.

A presença de 57 chaves de log únicas (variedade nos milhões de linhas) no dataset indica que o sistema captura uma ampla gama de eventos, treinando de maneira mais ampla o BERT. A diversidade de chaves de log também influencia diretamente as métricas de desempenho do modelo, como por exemplo o F1-Score. Se o modelo consegue manter um alto desempenho mesmo com a variedade de chaves de log, isso indica que ele é eficiente em capturar anomalias, independentemente da raridade dos eventos. Portanto, essas chaves de log únicas são fundamentais, pois refletem a complexidade dos ambientes da vida real (fora das condições de treino).

5 CONCLUSÃO

Neste trabalho, desenvolveu-se um método automatizado e contínuo para a detecção e classificação de falhas em infraestruturas de TI por meio de análise e classificação de logs de fonte Open Source empregando a Rede Neural baseada em Transformers BERT. Identificou-se um problema real, definiu-se o que era importante classificar utilizando técnicas de análise exploratória. Priorizou-se a eficiência do processo de treinamento e e inferência do modelo, empregando menos recursos de hardware, menor tempo de treinamento e uma quantidade reduzida de dados. Este trabalho contribui com o aumento da eficiência técnica do treinamento do modelo.

O processo resulta em um modelo bom o suficiente para ser usado em ambientes com limitações de hardware e infraestrutura (para treino e inferência), tempo e dados, ampliando sua aplicabilidade em ambientes que não dispõem desses em abundância. Assim, aumentando a resiliência e segurança dos contextos nos quais é aplicado.

REFERÊNCIAS

DEVLIN, J. *et al.* Bert: Pre-training of deep bidirectional transformers for language understanding. *In*: **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186. Disponível em: https://aclanthology.org/N19-1423.

DU, M. *et al.* Deeplog: Anomaly detection and diagnosis from system logs through deep learning. *In*: **Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security**. New York, NY, USA: Association for Computing Machinery, 2017. (CCS '17), p. 1285–1298. ISBN 9781450349468. Disponível em: https://doi.org/10.1145/3133956.3134015.

GUO, H.; YUAN, S.; WU, X. **LogBERT: Log Anomaly Detection via BERT**. 2021. Disponível em: https://arxiv.org/abs/2103.04475.

RAFIQUE, D.; VELASCO, L. Machine learning for network automation: Overview, architecture, and applications [invited tutorial]. **Journal of Optical Communications and Networking**, Optica Publishing Group, v. 10, n. 10, p. D126–D143, October 2018. Disponível em: https://opg.optica.org/jocn/abstract.cfm?URI=jocn-10-10-D126.

RYCIAK, P.; WASIELEWSKA, K.; JANICKI, A. Anomaly detection in log files using selected natural language processing methods. **Applied Sciences**, v. 12, n. 10, p. 5089, 2022. ISSN 2076-3417. Disponível em: https://www.mdpi.com/2076-3417/12/10/5089.

YOGATAMA, D. *et al.* Memory architectures in recurrent neural network language models. 2018. Disponível em: https://openreview.net/forum?id=SkFqf0lAZ.