-

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel de Castro Teixeira Pinheiro

Redução de Dimensionalidade com Descida de Gradiente: Uma Alternativa ao PCA para Preservação de Distâncias

Uberlândia, Brasil

2024

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Gabriel de Castro Teixeira Pinheiro

Redução de Dimensionalidade com Descida de Gradiente: Uma Alternativa ao PCA para Preservação de Distâncias

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como requisito exigido parcial à obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Bruno Augusto Nassif Travençolo

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Ciência da Computação

Uberlândia, Brasil

2024

Resumo

Este trabalho apresenta uma abordagem para a redução de dimensionalidade, utilizando descida de gradiente para reduzir as dimensões preservando melhor as distâncias entre os vetores originais. Em muitas aplicações, como no uso de *embeddings* gerados por LLMs (*Large Language Models*), a distância entre os vetores é uma métrica fundamental para representar a similaridade semântica entre os textos vetorizados. No entanto, esses vetores frequentemente possuem milhares de dimensões, o que torna inviável a aplicação de técnicas de visualização de dados ou reconhecimento de padrões sem uma redução dimensional. Métodos tradicionais, como o PCA (*Principal Component Analysis*), são amplamente utilizados para esse propósito, mas tendem a distorcer as distâncias entre os vetores durante o processo de redução. A técnica proposta neste trabalho visa minimizar essa distorção, oferecendo uma alternativa para reduzir as dimensões preservando as relações de proximidade entre os vetores no espaço reduzido. Foram realizados experimentos com comentários deixados publicamente na *Google Play Store* por usuários dos aplicativos analisados. Os comentários foram vetorizados pelo modelo *text-embedding-3-small* da *OpenAI* e, posteriormente, as 1536 dimensões de saída do modelo foram reduzidas para apenas duas utilizando PCA e o método proposto. Comparando os resultados dos dois algoritmos, pôde-se observar que ambos conseguiram manter uma separação clara entre comentários positivos e negativos, o que indica que preservaram bem a relação semântica original dos dados. O PCA demonstrou melhor desempenho na identificação e separação de grupos semânticos dentro dos comentários, preservando a topologia dos dados originais e facilitando a análise de *clusters*. Por outro lado, o método proposto mostrou-se mais eficaz na detecção de comentários anômalos, ou *outliers*, destacando aqueles que se distanciam significativamente dos demais e preservando as distâncias reais entre vetores no espaço reduzido. Com os experimentos realizados, foi demonstrado que tanto o PCA quanto o método proposto têm pontos positivos e negativos. Como conclusão, o método proposto é uma alternativa para a redução de dimensionalidade quando o objetivo é preservar melhor as distâncias entre os vetores, como na detecção de anomalias. No entanto, assim como o PCA, ele não é uma solução definitiva. Os dois métodos têm suas limitações e aplicabilidades específicas, a escolha entre eles deve ser guiada pelas necessidades de cada análise, reconhecendo que nenhum deles é uma solução definitiva, mas sim ferramentas complementares.

Palavras-chave: *Redução de Dimensionalidade*, *Descida de Gradiente*, *Preservaçao de Distâncias*, *PCA*, *LLM*, *Similaridade Semântica*, *Visualização de Dados Textuais*, *Preservação*.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Este trabalho apresenta uma abordagem para a redução de dimensionalidade, utilizando descida de gradiente para reduzir as dimensões preservando melhor as distâncias entre os vetores. Em muitas aplicações, como no uso de *embeddings* gerados por LLMs (*Large Language Models*), a distância entre os vetores é uma métrica fundamental para representar a similaridade semântica entre os textos vetorizados (REIMERS; GUREVYCH, 2019). No entanto, esses vetores frequentemente possuem milhares de dimensões, o que torna inviável a aplicação de técnicas de visualização de dados ou reconhecimento de padrões sem redução dimensional.

Métodos tradicionais, como o PCA (*Principal Component Analysis*), são amplamente utilizados para esse propósito (ETHAYARAJH, 2019), mas tendem a distorcer as distâncias entre os vetores durante o processo de redução (MAATEN; POSTMA; HERIK, 2009a). A técnica proposta neste trabalho visa minimizar essa distorção, oferecendo uma alternativa mais eficaz para reduzir as dimensões, preservando as relações de proximidade entre os vetores no espaço reduzido.

Neste trabalho, os vetores originais serão gerados pelo modelo *text-embedding-3-small* da OpenAI (OpenAI, 2024), que são representados em 1536 dimensões no espaço original. No espaço reduzido, dado o foco na visualização desses dados, eles serão transformados para duas dimensões.

1.1 Objetivo e Motivação

Com o avanço e popularização dos *LLMs* (Large Language Models), como os GPTs (*Generative Pre-trained Transformers*) da OpenAI (OpenAI, 2023), os *embeddings* textuais têm sido amplamente utilizados para representar textos em aplicações de busca semântica (Hugging Face, 2023), classificação de texto (Google AI, 2023a), sistemas de recomendação (Google AI, 2023c), detecção de anomalias (Google AI, 2023b), entre outras. Apesar de sua eficácia, esses *embeddings* frequentemente possuem milhares de dimensões, o que torna impraticável a aplicação direta de técnicas de visualização de dados, dificultando a análise visual e a interpretação de padrões contidos nesses vetores.

Neste trabalho, a motivação central está na necessidade de reduzir esses vetores para duas dimensões, para permitir sua visualização em gráficos bidimensionais. A aplicação prática deste estudo foca em dados textuais provenientes de comentários de usuários em aplicativos. Comentários de usuários serão vetorizados com o modelo *text-embedding-3-small*, e os vetores resultantes serão reduzidos para duas dimensões utilizando o PCA

e a metodologia proposta. Na visualização desses vetores em um gráfico, os comentários mais relacionados estarão mais próximos entre si, sendo possível entender os temas mais abordados nos comentários e como eles se relacionam.

Uma das vantagens do PCA é sua eficiência computacional, principalmente em comparação com a técnica que será explorada neste trabalho. No entanto, uma de suas limitações é que ele pode não capturar algumas relações semânticas entre os valores por assumir que as relações dos dados são lineares (MAATEN; POSTMA; HERIK, 2009a). Para ilustrar melhor essa limitação, podemos considerar um exemplo onde se deseja reduzir de duas para uma dimensão os pontos distribuídos como na Figura 1.

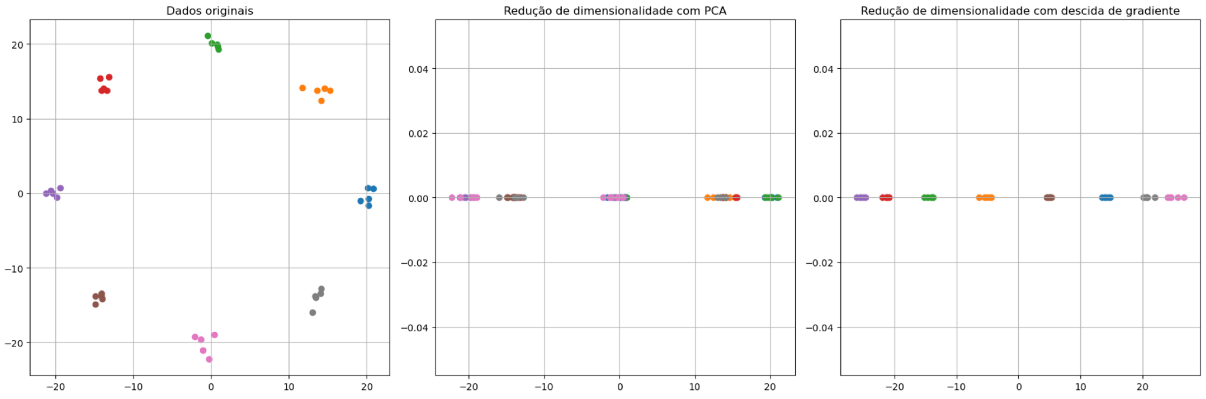


Figura 1 – Comparação de redução de dimensionalidade com PCA e descida de gradiente.

Ao realizar a redução de duas para uma dimensão, o PCA projeta os pontos em uma linha escolhida como direção principal. No entanto, observa-se que, independentemente da linha selecionada, pontos de diferentes grupos acabam se sobrepondo na projeção, como demonstrado no gráfico do meio. Dessa forma, as distâncias entre pontos de grupos distintos podem se tornar tão pequenas quanto entre pontos de um mesmo grupo. Em contraste, ao aplicar o método proposto neste trabalho, os grupos permanecem separados, refletindo melhor no espaço reduzido as relações de distância entre os pontos do espaço original, como demonstrado no gráfico da direita.

1.2 Justificativa

Métodos tradicionais, como o PCA (*Principal Component Analysis*), são amplamente utilizados para esse propósito, mas tendem a distorcer as distâncias entre os vetores durante o processo de redução (MAATEN; POSTMA; HERIK, 2009a). A técnica proposta neste trabalho visa minimizar essa distorção, oferecendo uma alternativa mais eficaz para reduzir as dimensões preservando as relações de proximidade entre os vetores no espaço reduzido.

Aplicações em que a similaridade semântica da entrada é o foco podem se beneficiar de uma técnica que distorce menos as distâncias (REIMERS; GUREVYCH, 2019), pois a similaridade semântica é mapeada para a distância entre os vetores originais. A redução de dimensionalidade preferível para esses casos deve manter a distância entre os vetores reduzidos.

2 Referencial Teórico

2.1 Conceitos Fundamentais

Nesta seção, são descritos os principais conceitos que formam a base para a melhor compreensão do tema e do trabalho.

2.1.1 Distância entre Vetores

Existem várias formas de calcular a distância entre vetores, neste trabalho serão abordadas a distância Euclidiana e a distância de cosseno. Isso porque são as métricas frequentemente utilizadas para medir a similaridade entre os dados originais utilizados para gerar os vetores de *embedding* (REIMERS; GUREVYCH, 2019).

2.1.1.1 Distância Euclidiana

A distância Euclidiana é a distância geométrica entre dois pontos em um espaço Euclidiano. Ela é calculada pela raiz quadrada da soma dos quadrados das diferenças entre as coordenadas dos pontos. Ela será utilizada comumente no espaço reduzido por ser uma distância mais intuitiva e fácil de visualizar.

Dados dois vetores $\mathbf{p} = (p_1, p_2, ..., p_n)$ e $\mathbf{q} = (q_1, q_2, ..., q_n)$, a distância Euclidiana entre eles é dada pela fórmula:

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{2.1}$$

2.1.1.2 Distância de Cosseno

A distância de cosseno é uma medida de distância entre dois vetores, calculada como o cosseno do ângulo entre eles. Visto que vetores de *embedding* originalmente possuem magnitudes muito diferentes, a distância de cosseno é uma métrica mais robusta para medir a distância entre eles, por não ser sensível à magnitude dos vetores (REIMERS; GUREVYCH, 2019).

Dados dois vetores $\mathbf{p}$ e $\mathbf{q}$, a distância de cosseno entre eles é dada pela fórmula:

$$d(\mathbf{p}, \mathbf{q}) = 1 - \frac{\mathbf{p} \cdot \mathbf{q}}{||\mathbf{p}|| \cdot ||\mathbf{q}||} \tag{2.2}$$

Dado a diferença de magnitude entre os vetores, é comum que os modelos de embeddings normalizem os vetores para terem magnitude unitária (OpenAI, 2024). Com isso, o denominador da fórmula acima é sempre 1, simplificando o cálculo da distância de cosseno, neste trabalho, para:

$$d(\mathbf{p}, \mathbf{q}) = 1 - \mathbf{p} \cdot \mathbf{q} \tag{2.3}$$

A distância de cosseno será utilizada no espaço original, antes da redução de dimensionalidade, para calcular a distância de cosseno entre os vetores originais.

2.1.2 Descida de Gradiente

Descida de gradiente é um método de otimização amplamente utilizado para encontrar o mínimo de uma função. O método consiste em iterativamente ajustar os parâmetros da função na direção oposta ao gradiente da função, até que se atinja um mínimo local ou global.

No método apresentado, será criada uma métrica de *loss* (perda) que mede o quão diferente as distâncias entre os vetores no espaço reduzido são das distâncias entre os vetores no espaço original. A perda será minimizada utilizando descida de gradiente, reduzindo a diferença entre as distâncias dos vetores no espaço original e no espaço reduzido.

A função de perda será calculada utilizando o erro quadrático médio (MSE, *Mean Squared Error*) (TERVEN; OUTROS, 2023), que corresponde à média dos quadrados das diferenças entre as distâncias de cada par de vetores no espaço original e no espaço reduzido. As diferenças são elevadas ao quadrado para garantir que qualquer desvio, seja ele positivo ou negativo, contribua igualmente para o cálculo do erro, com o objetivo de aproximá-lo de zero. A descida de gradiente será então aplicada para minimizar essa função de perda, ajustando os parâmetros de modo que as distâncias no espaço reduzido se tornem cada vez mais próximas das distâncias no espaço original.

A descida de gradiente é baseada na ideia de mover os parâmetros, no nosso caso as coordenadas dos vetores do espaço reduzido, na direção que mais rapidamente reduz o valor da função de perda. Esta direção é indicada pelo gradiente, que aponta para o aumento mais acentuado da função. Ao seguir o sentido contrário ao do gradiente, o algoritmo converge para um ponto onde a perda é minimizada (KINGMA; BA, 2015).

O algoritmo Adam (KINGMA; BA, 2014) será utilizado, pois ele apresentou os melhores resultados ao incorporar um fator de inércia no processo de descida de gradiente. Isso permite que os vetores atravessem regiões densas de outros vetores para alcançar uma posição ideal, mesmo que isso ocasionalmente aumente temporariamente o valor da perda durante o ajuste.

Adam (*Adaptive Moment Estimation*) é um método de otimização muito utilizado em aprendizado de máquina por aplicar momento por meio da acumulação de gradientes passados, o que ajuda a suavizar e direcionar o processo de descida. Ele calcula médias móveis do gradiente e de seu quadrado, permitindo que o algoritmo avance de forma estável mesmo em funções complexas e em espaços de parâmetros complicados. A inércia aplicada pelo Adam ajuda a contornar mínimos locais e regiões de baixa inclinação, mantendo uma direção de descida consistente, o que facilita a busca por um mínimo global, mesmo em regiões instáveis (KINGMA; BA, 2015).

2.1.3 PCA (*Principal Component Analysis*)

Redução de dimensionalidade é uma técnica fundamental em diversas áreas do aprendizado de máquina, estatística e ciência de dados. Seu principal objetivo é transformar dados de alta dimensionalidade em um espaço de menor dimensão, preservando ao máximo as características relevantes dos dados originais.

O PCA é uma técnica de redução de dimensionalidade que transforma os dados originais em um novo espaço de menor dimensão, mantendo a maior variância dos dados (SHLENS, 2014). O PCA é amplamente utilizado para visualização de dados e reconhecimento de padrões, mas tende a distorcer as distâncias entre os vetores durante o processo de redução (MAATEN; POSTMA; HERIK, 2009a).

O PCA foi escolhido como base de comparação para a técnica proposta neste trabalho, pois é a técnica mais utilizada para visualização de *embeddings* de texto, sendo sugerido pela *OpenAI* como uma técnica eficaz para reduzir a dimensionalidade dos *embeddings* gerados por seus modelos (OpenAI, 2022).

O processo de PCA envolve a decomposição dos dados em componentes principais, que são direções ortogonais ao longo das quais a variância dos dados é maximizada. Cada componente principal representa uma combinação linear das variáveis originais, e o número de componentes selecionados define a nova dimensionalidade do espaço transformado. Esses componentes são ordenados pela variância que cada um captura, de modo que os primeiros componentes principais são aqueles que mais retêm a informação relevante dos dados (SHLENS, 2014).

2.1.4 *Embeddings*

Embeddings são representações vetoriais de dados que capturam as relações semânticas entre eles. Por exemplo, no trabalho seminal de Mikolov et al. (2013), foi demonstrado que operações aritméticas simples em vetores podem capturar relações semânticas entre palavras. Um exemplo clássico é a operação $\text{rei} - \text{homem} + \text{mulher} \approx \text{rainha}$, onde o vetor resultante está próximo ao vetor associado à palavra “rainha”. Isso demonstra como os

embeddings capturam tanto a semântica quanto as relações entre conceitos. Os *embeddings* têm ganhado destaque com o avanço e popularização dos LLMs (*Large Language Models*), como os modelos de linguagem GPT (OpenAI, 2023), LLaMA (TOUVRON et al., 2023) e outros.

Os *embeddings* textuais são gerados por LLMs de embeddings, a distância entre os vetores gerados a partir de textos representa a similaridade semântica entre eles. Essa propriedade é comumente usada para busca semântica (Hugging Face, 2023), classificação de texto (Google AI, 2023a), sistemas de recomendação (Google AI, 2023c), detecção de anomalias (Google AI, 2023b), entre outras aplicações.

2.2 Trabalhos Relacionados

Existem diversas abordagens para redução de dimensionalidade, cada uma com características específicas. Nesta seção, serão discutidos trabalhos relevantes que contextualizam a metodologia proposta.

Maaten e Hinton (2008) introduziram o t-SNE (*t-Distributed Stochastic Neighbor Embedding*), um método de redução de dimensionalidade projetado para preservar a estrutura local dos dados. Embora muito eficiente para manter próximos no espaço reduzido os vetores que estavam próximos no espaço original, ele pode distorcer as distâncias entre grupos ou *clusters* de vetores.

Hinton e Salakhutdinov (2006) exploraram a utilização de *autoencoders*, uma arquitetura de rede neural projetada para aprender representações compactas e úteis dos dados, como uma abordagem para redução de dimensionalidade. Embora também usem descida de gradiente, a abordagem é diferente da proposta neste trabalho. Hinton e Salakhutdinov utilizaram descida de gradiente para treinar um *autoencoder* que reduz as dimensões dos dados, ao invés de utilizarem a descida de gradiente diretamente nos dados.

Maaten, Postma e Herik (2009b) apresentaram uma revisão comparativa de técnicas de redução de dimensionalidade, analisando vantagens e limitações de métodos como PCA, MVU, Isomap, entre outras. Este trabalho oferece uma base sólida para entender o panorama geral e identificar lacunas que a técnica proposta visa abordar. Ele se difere por comparar os algoritmos com dados sintéticos de menos dimensões que os *embeddings* utilizados neste trabalho.

Tervonen (2023) conduziu uma análise comparativa de técnicas de redução de dimensionalidade em *embeddings* textuais, com ênfase na comparação de seus desempenhos para vetores gerados por diferentes LLMs de *embedding*. Foram comparados modelos como *bert-base-multilingual-cased*, *distilbert-base-multilingual-cased*, *xlm-roberta-base*,

xlm-roberta-large e *LaBSE*.

3 Metodologia

Este capítulo detalha as etapas realizadas para investigar a percepção dos usuários de diversos aplicativos por meio da análise dos comentários coletados da *Google Play Store* utilizando o método proposto e comparando com o PCA. Inicialmente, os comentários foram coletados e filtrados. Em seguida, os comentários foram vetorizados utilizando um modelo de *embedding*, os transformando em vetores. A partir desses vetores, foram aplicadas duas técnicas de redução de dimensionalidade: o método proposto, baseado em descida de gradiente, que visa preservar melhor as relações de similaridade entre os comentários; e o PCA, para uma análise comparativa entre as duas técnicas, a fim de avaliar a qualidade das reduções e a precisão na preservação da distância original dos dados.

3.1 Coleta de Comentários

Para analisar a percepção dos usuários dos aplicativos, foram coletados comentários publicamente disponíveis na *Google Play Store*. Em seguida, eles foram filtrados para refletir apenas a última versão do aplicativo, remover comentários muito curtos ou muito longos e remover comentários duplicados.

3.2 Vetorização dos Comentários

Os comentários coletados foram vetorizados utilizando o modelo *text-embedding-3-small*, da *OpenAI*, que gera vetores de 1536 dimensões (OpenAI, 2024). Os vetores resultantes representam os comentários em um espaço de alta dimensionalidade, onde a distância entre os vetores é uma métrica fundamental para representar a similaridade semântica entre os textos.

3.3 Redução de Dimensionalidade com o Método Proposto

3.3.1 Inicialização

Os vetores no espaço reduzido são inicializados com valores aleatórios. Naturalmente, os vetores aleatórios representam mal as distâncias entre os vetores originais. Para aplicar a descida de gradiente, é necessário medir o quanto essa representação aleatória se distancia da original, utilizando somente funções deriváveis, para que possamos ajustar os vetores no espaço reduzido de forma a minimizar essa diferença.

Nos experimentos realizados, também foram testadas outras estratégias de inicialização, como a inicialização utilizando o resultado do PCA. No entanto, essas abordagens alternativas não apresentaram ganhos significativos em termos da preservação das relações entre os vetores originais, nem reduziram o tempo para convergência dos vetores reduzidos. Assim, a escolha por valores aleatórios como inicialização mostrou-se uma solução simples e suficiente, dado que os ajustes subsequentes via descida de gradiente compensam a escolha inicial.

3.3.2 Cálculo da Perda

A perda é calculada utilizando o erro quadrático médio (MSE, *Mean Squared Error*), que corresponde à média dos quadrados das diferenças entre as distâncias de cada par de vetores no espaço original e no espaço reduzido. As diferenças são elevadas ao quadrado para garantir que qualquer desvio, seja ele positivo ou negativo, contribua igualmente para o cálculo do erro, com o objetivo de aproximá-lo de zero.

Considerando $d_o(i, j)$ como a distância no espaço original entre o i-ésimo vetor e o j-ésimo vetor, e $d_r(i, j)$ como a distância entre eles no espaço reduzido:

$$MSE = \frac{1}{N} \sum_{i,j} \left(d_o(i, j) - d_r(i, j)\right)^2 \tag{3.1}$$

Além disso, para acelerar o processo de convergência, foi utilizada a estratégia de *mini-batches* (ASSOS, 2015), que consiste em calcular a perda e atualizar os vetores no espaço reduzido com base em pequenos subconjuntos aleatórios dos pares de vetores, ao contrário de considerar o conjunto completo a cada iteração. Essa abordagem permitiu uma redução no tempo de convergência em até 10 vezes, tornando o método mais rápido, especialmente em cenários com conjuntos de dados maiores. Essa etapa é conhecida como *forward pass*, em que calcula-se a perda para os vetores atuais no espaço reduzido (BISHOP, 1995).

3.3.3 Descida de Gradiente

Com a perda calculada, aplicamos a descida de gradiente para ajustar os vetores no espaço reduzido de forma a minimizar a perda. O algoritmo Adam é utilizado para otimizar a perda, ajustando os vetores na direção que mais rapidamente reduz o valor da função de perda.

Durante o processo podemos acompanhar a redução da perda, que iniciou em um valor alto, representando a má representação dos vetores aleatórios, e foi reduzindo até convergir para um valor baixo, representando uma boa representação dos vetores no espaço reduzido. Essa etapa é conhecida como *backward pass*, em que se ajusta os vetores no espaço reduzido para minimizar a perda (BISHOP, 1995).

3.3.4 Algoritmo em Pseudocódigo

O algoritmo acima pode ser descrito em pseudocódigo conforme segue:

Algoritmo 1: Redução de Dimensionalidade com Descida de Gradiente

Dados: Vetores originais *originais*, dimensionalidade a reduzir $DIMENSOES$, número de iterações $ITERACOES$

Resultado: Vetores reduzidos *reduzidos*

$reduzidos \leftarrow$ vetores aleatórios de $DIMENSOES$ dimensões;

Executar $ITERACOES$ **vezes**

- $loss \leftarrow 0$;
- $qnt \leftarrow 0$;
- **Para cada par** i, j **de vetores**
  - $distancia\_original \leftarrow d_o(i, j)$;
  - $distancia\_reduzida \leftarrow d_r(i, j)$;
  - $erro \leftarrow (distancia\_original - distancia\_reduzida)^2$;
  - $loss \leftarrow loss + erro$;
  - $qnt \leftarrow qnt + 1$;
- $loss \leftarrow loss/qnt$;
- $loss.backward()$;
- $Adam.update(reduzidos)$;

- `loss.backward()`: Esta operação computa os gradientes dos vetores no espaço reduzido com relação à função de perda, indicando como ajustar os vetores para reduzir o erro.
- `Adam.update(reduzidos)`: Aqui o algoritmo Adam utiliza os gradientes calculados para atualizar os vetores reduzidos com base nos gradientes calculados.

3.3.5 Visualização dos Vetores

Com os vetores ajustados no espaço reduzido, podemos visualizá-los em um gráfico de dispersão, onde os comentários mais relacionados estarão mais próximos entre si. Isso nos permite entender os temas mais abordados nos comentários e como eles se relacionam.

Nesta etapa, percebe-se se o algoritmo conseguiu separar comentários positivos de negativos, se grupos de comentários puderam ser identificados, quais foram os comentários atípicos, entre outras análises que poderão ser feitas visualmente.

3.4 Comparação entre PCA e o Método Proposto

O PCA foi utilizado como base de comparação para a técnica proposta neste trabalho. Visualizando o gráfico de dispersão dos vetores no espaço reduzido pelo PCA, podemos comparar a distribuição dos comentários com a distribuição obtida pela técnica proposta.

Para avaliar a qualidade das reduções quanto a preservação das distâncias entre os vetores, podemos calcular o erro quadrático médio entre as distâncias dos vetores no espaço original e no espaço reduzido, tanto para o PCA quanto para o método proposto. A comparação entre os dois métodos nos permite avaliar a eficácia da técnica proposta em preservar as relações de proximidade entre os vetores no espaço reduzido.

4 Experimentos

4.1 Escolha de Aplicativos

Para comparar o desempenho do método proposto com algoritmos existentes, foram escolhidos alguns aplicativos para executar as análises sobre os comentários deixados publicamente por seus usuários na *Google Play Store*:

- **NuBank**: Banco digital, contém uma separação clara entre comentários sobre o banco e sobre o aplicativo;
- **Itaú**: Banco mais tradicional, com perfis de usuários diferentes da NuBank;
- **Vivo Easy**: Plano de telefonia digital, também com separação entre comentários sobre a rede da operadora e comentários sobre o aplicativo;
- **Meu Vivo**: Planos de telefonia mais tradicionais, com perfis de usuários diferentes do Vivo Easy.

As categorias "banco" e "telecomunicações" foram escolhidas por conterem comentários de usuários que abordam funcionalidades específicas do aplicativo distintos dos comentários sobre funcionalidades de banco e rede. Dentro da categoria de bancos, os aplicativos NuBank e Itaú foram escolhidos por representarem, respectivamente, os maiores faturamentos entre instituições financeiras brasileiras nas categorias de bancos "nativamente digitais" e "tradicionais". Na categoria de telecomunicações, os aplicativos da Vivo foram selecionados devido à distinção entre o Vivo Easy, voltado para um plano nativamente digital, e o Meu Vivo, utilizado para gerenciar planos tradicionais.

As seguintes seções irão focar nos resultados do aplicativo NuBank, pois ele obteve os resultados que melhor demonstraram os pontos positivos e também negativos do algoritmo proposto. Ainda assim, serão apresentados todos os resultados dos experimentos.

4.2 Coleta e Tratamento dos Comentários

Foi utilizado o pacote *google_play_scraper* para baixar os últimos 10.000 comentários de cada aplicativo. Para o caso do NuBank, houve 7.078 comentários únicos.

Os comentários baixados foram tratados utilizando a extensão *DataWrangler* da IDE *Visual Studio Code*. Foram removidas colunas não utilizadas e os comentários foram filtrados para remover aqueles feitos em versões antigas, reduzindo para 1669 comentários

únicos e remover comentários muito curtos ou muito longos, reduzindo para 430 comentários únicos. Um exemplo dos comentários mais recentes após o filtro está na Tabela 1.

Conteúdo	**Estrelas**
A nubak não da limite não i. Não presta não	1
porque pedem meu CPF e senha se uso a digital	5
Empresa de confiança eu super indico:	5
amo meu aplicativo do nubank é o melhor	5
Boletos em dia e bloquearam meus cartões PJ e PF pqp	5
o bom é que pode menor de idade	5
Tô gostando muito de usar esse maravilhoso roxinho	5
e um banco prático fácil de manusear sou apaixonado	5
ótimo aplicativo, bem fácil pra movimentar sem complicações	5
Pouco intuitivo e muito confuso. Saudades app easynvest	1
Muito prático de utilizar. Atendimento ótimo!	5
ótimo uso para as minhas transferências	5

Tabela 1 – Últimos comentários com as respectivas avaliações.

Em seguida, os comentários foram vetorizados utilizando o modelo *text-embedding-3-small* da *OpenAI*, gerando vetores de 1536 dimensões para cada um. A nota em estrelas dada pelos usuários não foi enviada para o modelo e será utilizada somente na visualização dos dados para validar se o modelo conseguiu segmentar os comentários positivos de negativos.

4.3 Redução de Dimensionalidade

Os vetores gerados foram reduzidos para duas dimensões, de tal forma que possam ser colocados em um gráfico, utilizando PCA, como base de comparação, e descida de gradiente utilizando o método proposto. Com a descida de gradiente, mesmo utilizando *mini-batches*, a convergência foi rápida e aconteceu próxima da iteração 70, como demonstrado no gráfico da Figura 2.

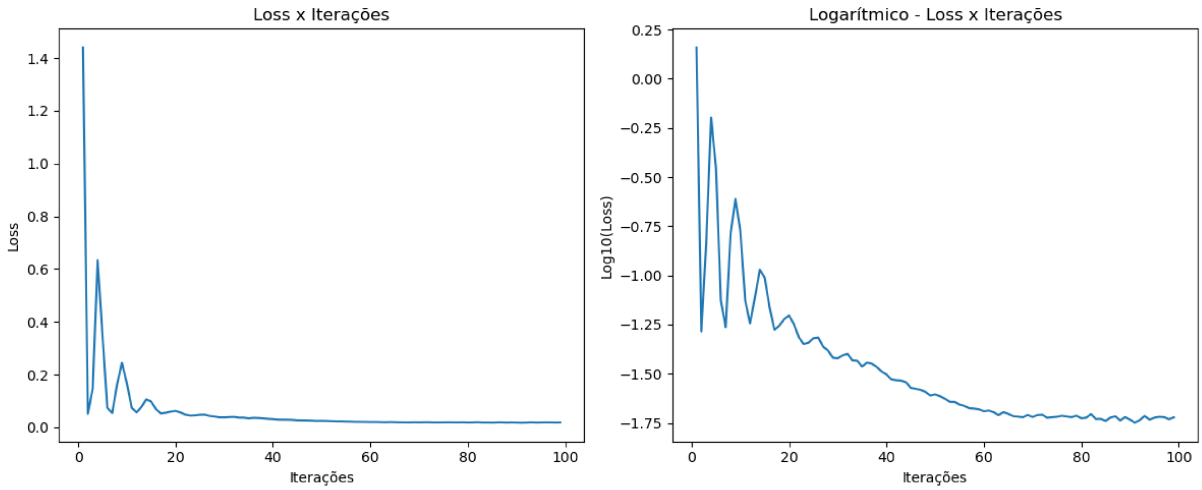


Figura 2 – Evolução da perda ao longo das iterações.

O resultado das duas reduções é o que segue na Figura 3. Cada ponto representa um comentário deixado pelos usuários publicamente na *Google Play Store*. Pontos mais próximos são comentários com alta similaridade semântica e vice-versa. Os pontos foram coloridos para refletir a quantidade de estrelas que o usuário deixou na avaliação. Comentários com cinco estrelas estão representados em verde, com uma estrela em vermelho e comentários com duas a quatro estrelas em diferentes tons de laranja. O gráfico da esquerda mostra os vetores reduzidos utilizando PCA como base de comparação, o gráfico da direita mostra os vetores reduzidos com o método proposto.

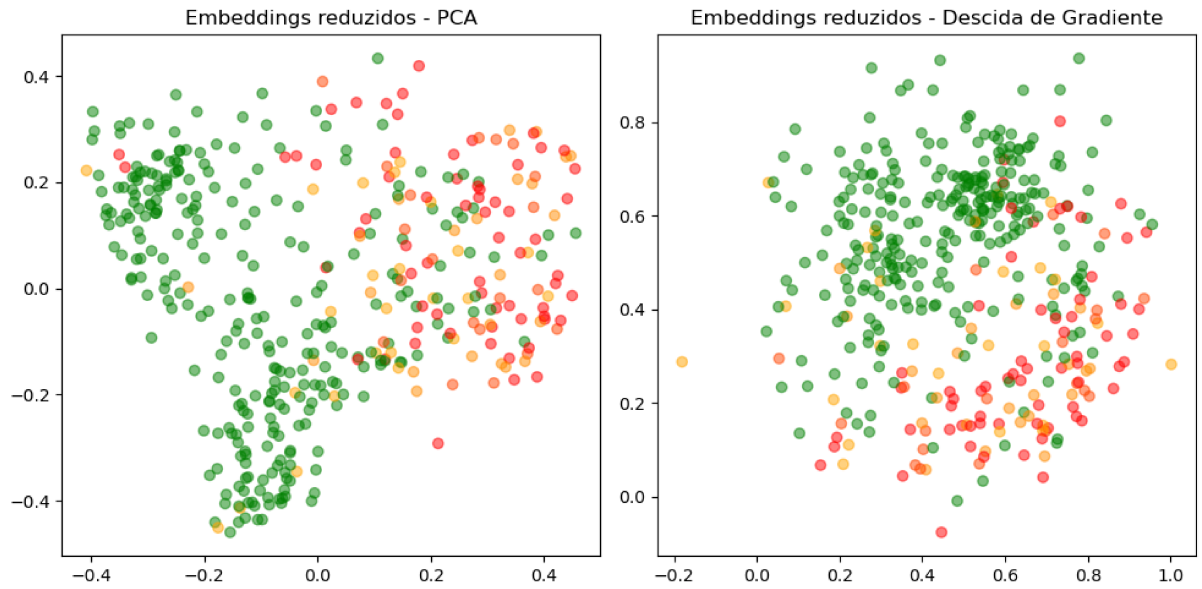


Figura 3 – Na esquerda, vetores reduzidos com PCA; na direita, vetores reduzidos com o método proposto.

Além da análise feita no aplicativo do NuBank, também foram avaliados os aplicativos do Itaú, MeuVivo e Vivo Easy. Seus resultados seguem nas Figuras 4, 5, 6 e 7. As

diferenças da Figura 3 com a Figura 4 são pelo fato de os vetores serem aleatoriamente inicializados. Portanto, são mínimos locais diferentes com MSEs próximos.

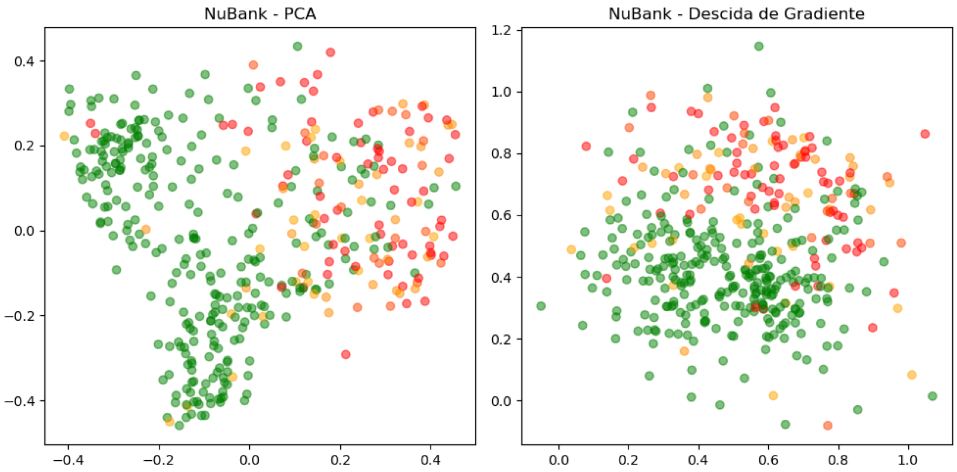


Figura 4 – Comentários vetorizados e reduzidos do NuBank.

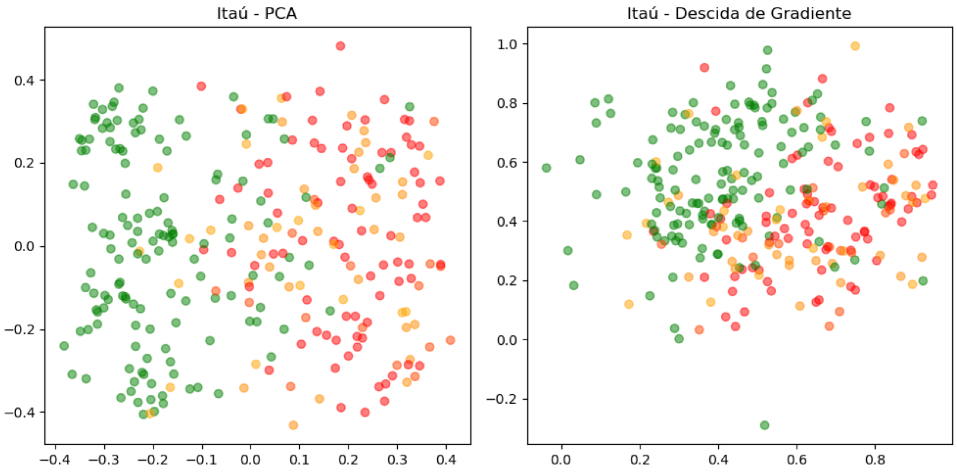


Figura 5 – Comentários vetorizados e reduzidos do Itaú.

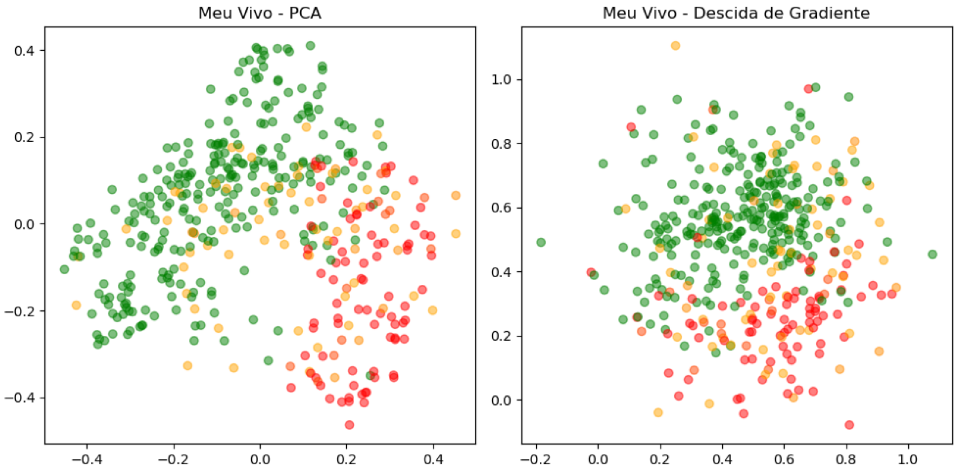


Figura 6 – Comentários vetorizados e reduzidos do MeuVivo.

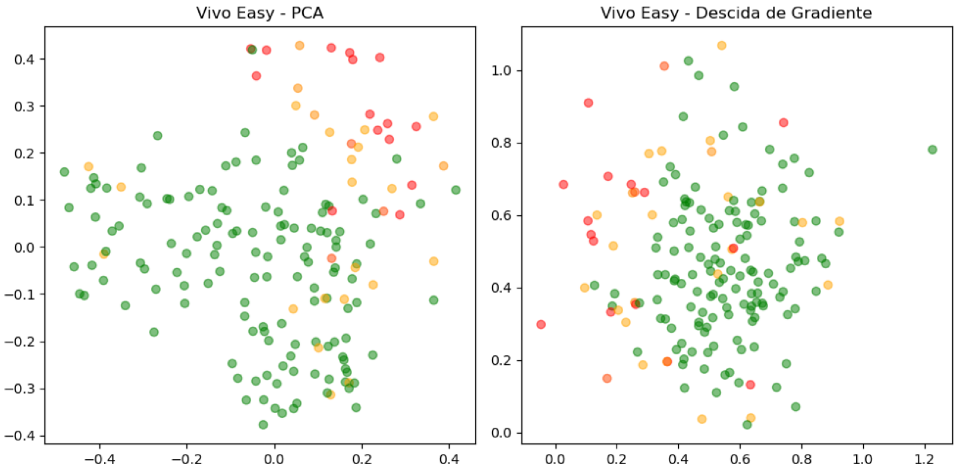


Figura 7 – Comentários vetorizados e reduzidos do Vivo Easy.

5 Resultados e discussões

5.1 Resultados

Em análise visual do resultado da Figura 3, algumas diferenças e semelhanças são notáveis entre os vetores reduzidos por cada algoritmo.

5.1.1 Segmentação de Notas

Reprodução da Figura 3.

Na Figura 3, é possível perceber que ambos os algoritmos conseguiram manter uma separação clara entre os comentários positivos, em verde, e os comentários negativos, em vermelho. A cor dos pontos foi designada de acordo com a nota que o usuário deu na loja entre 1 e 5, e não foi utilizada nem pelo modelo de *embedding* nem pelos algoritmos de redução. Assim, fica evidente que o sentimento dos comentários está compatível com a nota dada e que ambos os algoritmos de redução preservam bem essa relação.

Isso demonstra que ambos os métodos desempenharam igualmente bem para manter a diferenciação visual entre comentários positivos e negativos interpretados pelo modelo.

5.1.2 Isolamento de Grupos

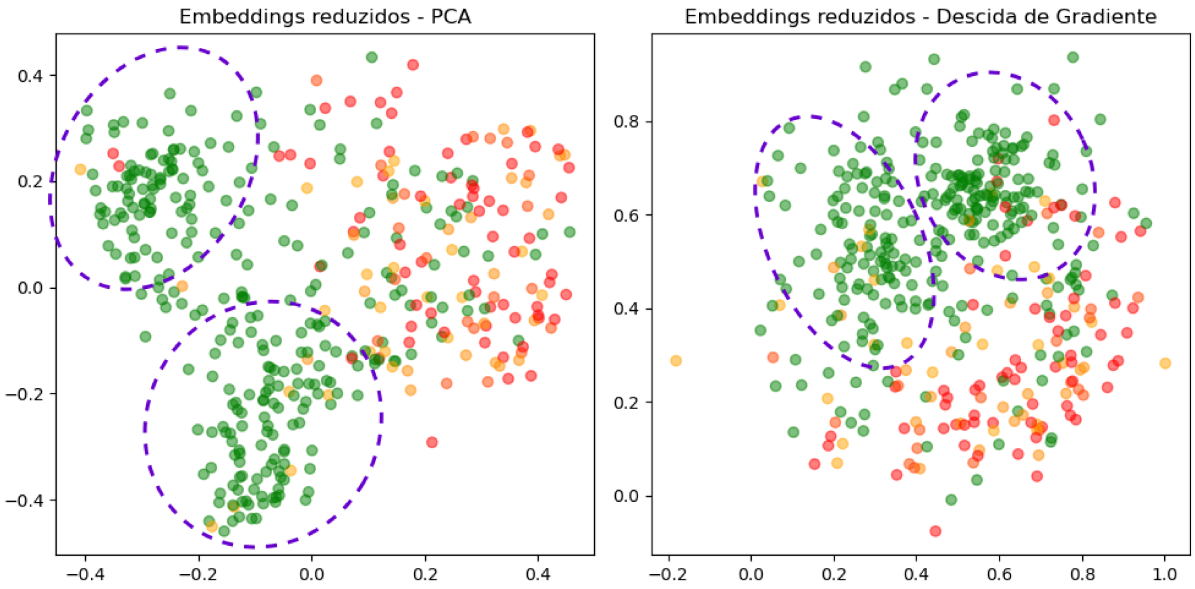


Figura 8 – Diferentes grupos de comentários positivos.

É possível perceber na Figura 8 que, entre os comentários positivos, houve separação entre usuários comentando sobre o aplicativo do NuBank e usuários comentando sobre os serviços de banco do NuBank. Essa separação existe em ambas as reduções, mas é muito mais clara no PCA, pois ele preservou melhor a topologia dos vetores originais, mesmo que isso signifique piorar as relações de distância entre os vetores, mantendo os dois grupos melhor separados.

Na Figura 9 seguem alguns exemplos de comentários transcritos na Tabela 2 dos dois grupos para cada um dos algoritmos, evidenciando que um é sobre o aplicativo e o outro sobre o banco.

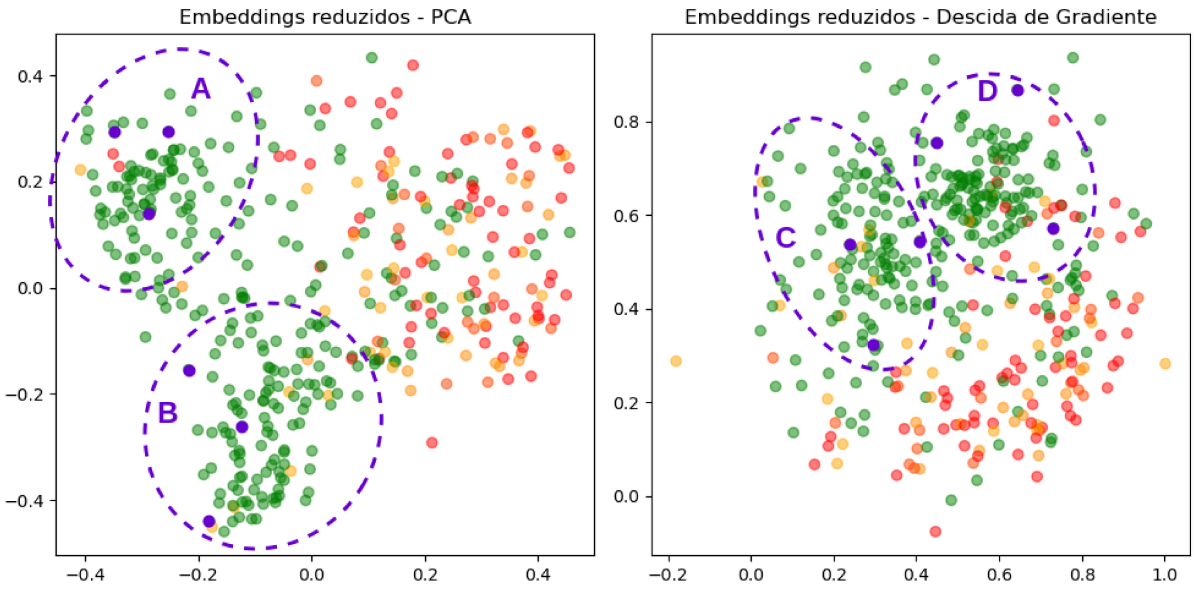


Figura 9 – Alguns vetores selecionados em cada grupo.

Grupo	Categoria	Comentário
A	Banco	o melhor banco digital de todos os tempos, parabéns Nubank
A	Banco	maravilha nubak um banco de confiança
A	Banco	é o melhor banco q já tive...meus parabéns
B	Aplicativo	Ótimo aplicativo e um cartão excelente
B	Aplicativo	muito bom aplicativo do nubake.
B	Aplicativo	Ótimo aplicativo, Intuitivo e muito fácil de usar
C	Aplicativo	Meu primeiro aplicativo o melhor que já tive sem problema.
C	Aplicativo	App ótimo. Fácil de usar. Controle total das minhas finanças
C	Aplicativo	o app é bom, uso todos os dias.
D	Banco	é um banco amigo, quando mais precisamos ele nos socorre
D	Banco	uns dos melhores bancos que já tivemos
D	Banco	maravilha nubak um banco de confiança

Tabela 2 – Comentários selecionados na Figura 9.

Isso demonstra que o PCA teve um desempenho melhor que o método proposto para analisar grupos ou *clusters* de comentários, por manter melhor a topologia dos dados.

5.1.3 Comentários Anômalos (*Outliers*)

Outra característica a se observar são os comentários anômalos, *outliers*, aqueles que mais se distanciam do restante dos comentários, estando nas bordas dos gráficos. A Figura 10 evidencia alguns desses comentários.

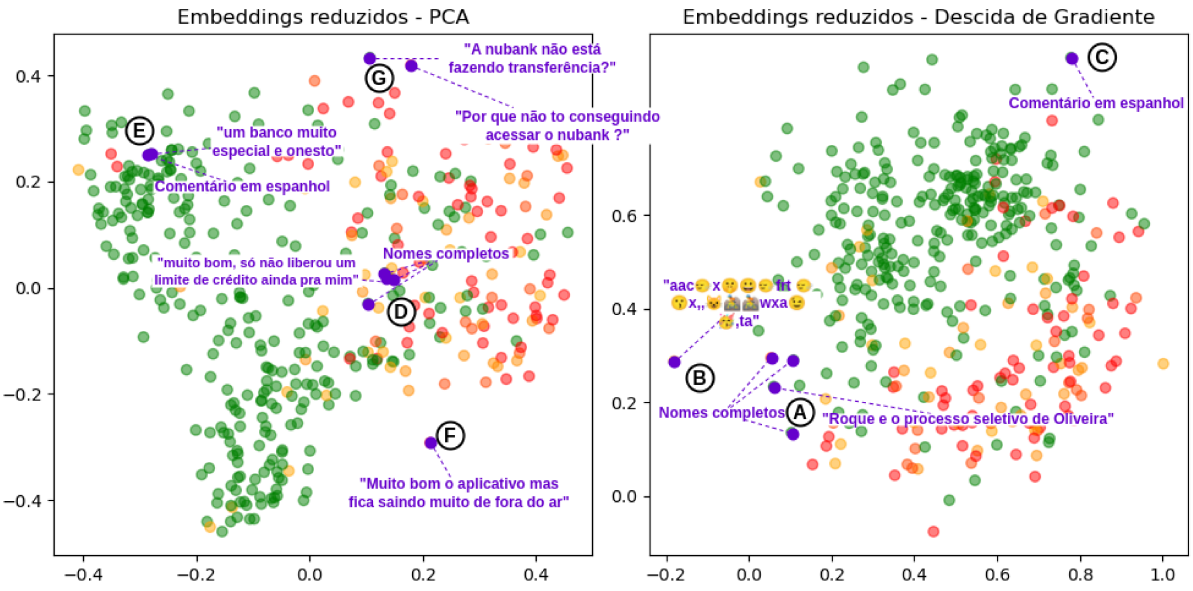


Figura 10 – Comentários *outliers*.

Nos *embeddings* reduzidos pelo método proposto, é possível perceber próximo do ponto "A" quatro comentários que estão distantes do restante, de usuários que comentaram o próprio nome. Junto deles está um comentário que, por mais que não seja somente um nome, contém nomes próprios como "Roque" e "Oliveira". O ponto em "B" é o que mais se distancia do restante dos comentários e é de um usuário que comentou emojis e letras aleatórias. Ainda, no ponto "C", o único comentário em outra língua que também se distancia do restante. Com isso, é possível perceber que o método proposto consegue destacar bem os comentários anômalos do restante.

Já nos *embeddings* reduzidos utilizando PCA, é possível perceber no ponto "D" que os mesmos comentários anômalos com nomes de pessoas se encontram misturados com outros comentários mais comuns, inclusive, destacado por um comentário que está extremamente próximo deles e não é anômalo. O mesmo acontece com o ponto "E" em que o comentário em espanhol ("Estoy muy feliz, me gusta mucho nubank") está junto com comentários padrões em português, inclusive com distância muito curta ao outro comentário destacado ("um banco muito especial e onesto") mesmo com o conteúdo muito diferente. No ponto "F" podemos notar o comentário que mais se distancia do restante dos outros pontos da redução, e é um comentário completamente normal. Já no ponto "G" temos dois comentários que, também, estão distantes do restante e realmente se diferenciam por ser perguntas. Mas, os outros cinco comentários com perguntas foram colocados junto com os comentários afirmativos, não se destacando como esses. Demonstrando, assim, que a redução com PCA desses vetores não é tão efetiva para analisar anomalias nos dados textuais.

Isso demonstra que o método proposto desempenhou melhor que o PCA para manter as anomalias separadas da massa de dados padrão, por garantir que a maior

distância que os vetores anômalos possuem dos outros vetores no espaço original será refletida no espaço reduzido.

5.1.4 Similaridade de Comentários Próximos

Outra característica a se observar é se os comentários que aparecem muito próximos no gráfico realmente têm semânticas tão similares quanto suas proximidades. Contudo, como demonstrado na Figura 11 e na Tabela 3, tanto a redução com PCA quanto o método proposto têm tanto grupos de comentários próximos com semântica parecida, quanto diferente.

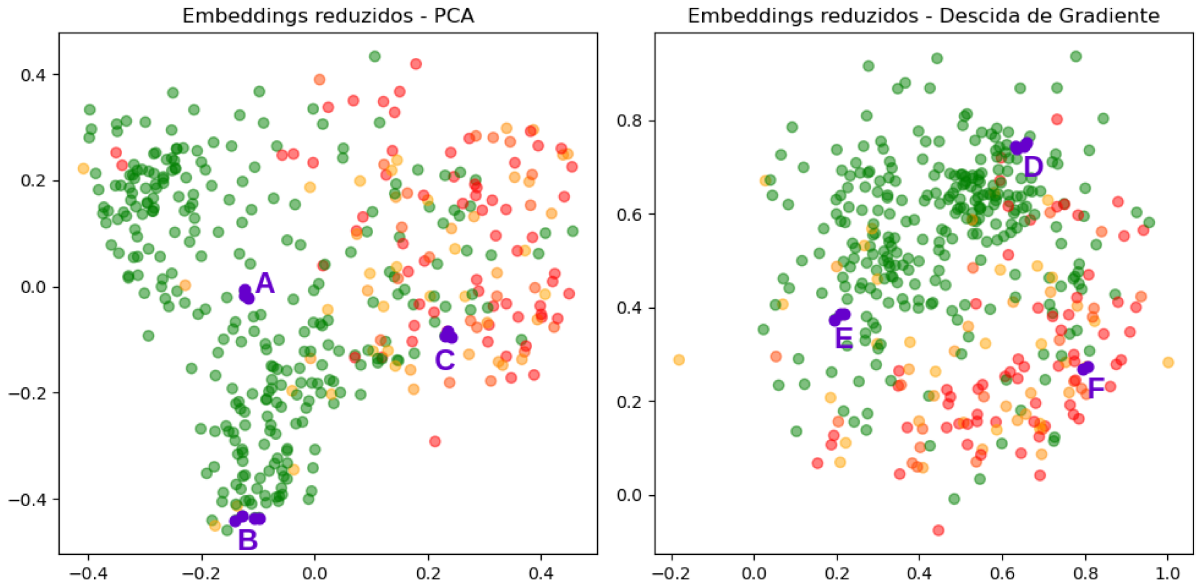


Figura 11 – Comentários próximos nos gráficos.

Grupo	**Comentário**
A	é muito bom graças a Deus eu tenho esse cartão
A	O app do banco é muito eficiente funciona com normalidade
A	Pode ter mais de um cartão ele e muito bom
B	um app ótimo, fácil de usar e muito prático
B	aplicativo ótimo, simples e fácil de usar
B	Esse aplicativo é muito bom eu gostei muito
B	gosto bastante do app, funciona muito bem.
C	se ficar pedindo avaliação, vai ganhar nota ruim
C	Mt bom, ate agr n ocorreu nenhum tipo de erro
C	bom de mais devia dar mais limites
D	top top um dos melhores banco digital
D	é o melhor cartão de crédito Brasil
D	Nubank é incrível, melhor banco de todos
D	A Nubank com certeza é a melhor
E	Tô gostando muito de usar esse maravilhoso roxinho
E	app é bom fácil de usar, explicações são claras
E	acho que deveria ter mais opções no app
F	Boa tarde paguei meu acordou e meu limite não foi liberado
F	liberou o limite, mas não autorizou uso o crédito.

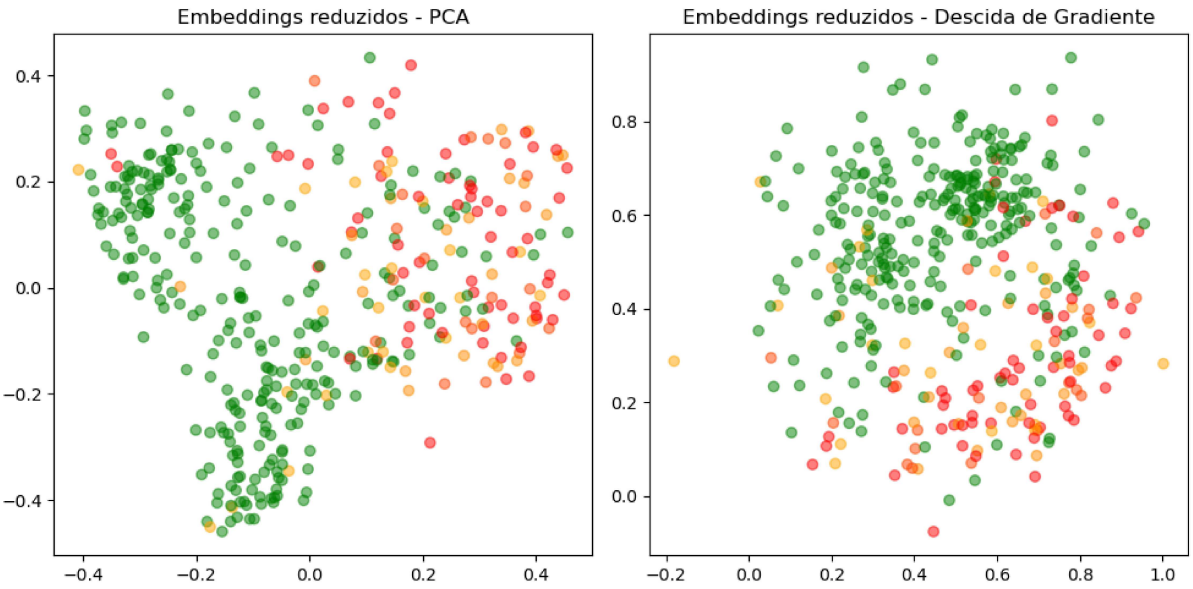
Tabela 3 – Comentários por grupos da Figura 11.

Isso pode se dar pelo fato de o modelo original gerar vetores extremamente próximos para alguns comentários moderadamente diferentes, quanto pelo fato de a redução de dimensionalidade ser um processo que perde informações, ainda mais quando tão extremo quanto aqui, reduzindo de 1536 dimensões para apenas duas, quanto por fatores diferentes em cada método.

Assim, é inconclusivo, sem uma análise mais profunda, qual dos dois métodos desempenham melhor para representar vetores reduzidos extremamente próximos, somente se os originais forem, também, extremamente próximos.

5.1.5 Ruído Aparente

Ao analisar a Figura 3, também é possível perceber que junto dos comentários positivos, em verde no gráfico, alguns comentários em vermelho, com avaliação negativa e vice-versa, tanto na redução do PCA quanto na do método proposto.



Reprodução da Figura 3.

Contudo, na maioria das vezes, eles se tratam de comentários em que a semântica do texto é realmente positiva mas o usuário deixou uma avaliação negativa, ou vice-versa, que não condiz com o conteúdo do texto. Alguns exemplos estão demonstrados na Tabela 4.

Comentário	**Avaliação**
Eu adorei o meu Nubank Super indico	1 Estrela
é muito maravilhoso esse banco eu amei ser cliente do nubank	1 Estrela
estou muito feliz com meu roxinho	3 Estrelas
Não to conseguindo receber um código	5 Estrelas
EU TAVA PRECISANDO DE DINHEIRO MAS EU NÃO TENHO DINHEIRO	5 Estrelas
Não consigo cancelar essa boata	5 Estrelas
só não sei porque não média um limite melhor	5 Estrelas
Péssimo n consigo abrir o aplicativo	5 Estrelas

Tabela 4 – Comentários fora do padrão e suas respectivas avaliações.

5.2 Discussões

Nesta seção, serão discutidas aplicações práticas em que o método proposto pode se mostrar útil, com exemplos e explorações que destacam sua relevância em diversos contextos.

Durante o trabalho, o uso concreto demonstrado foi de análise de dados com os vetores de *embedding*. O uso pode ser estendido para além de comentários públicos de aplicativos. Como, por exemplo, na avaliação privada que os usuários costumam fornecer, comentários de lojas em sistemas de *marketplace*, avaliações de produtos em lojas online,

publicações de usuários em redes sociais, comentários em publicações de redes sociais, entre outros.

Uma estratégia muito utilizada para contextualizar LLMs com dados relevantes, sem poluir seus contextos com dados não relacionados, é o RAG (*Retrieval-Augmented Generation*) (GUPTA; RANJAN; SINGH, 2024). No RAG, são gerados *chunks* de dados que são vetorizados por LLMs de embedding, e a distância desses chunks para o *prompt* é utilizada para escolher os mais relevantes. Em sistemas que utilizam RAG, utilizar o método proposto neste trabalho pode ser útil para analisar *chunks* anômalos que podem ser movidos para bases de conhecimento diferentes, *chunks* muito próximos que podem ser combinados, além de suas estruturas e como eles se relacionam.

O método proposto também apresenta potencial para ser utilizado na visualização de *embeddings* gerados para outros tipos de dados, além dos textuais. Atualmente, *embeddings* são amplamente utilizados em busca semântica para outras modalidades, como áudios (XIE; VIRTANEN, 2020), músicas (DOH et al., 2024) e imagens (LIU et al., 2020). A redução de dimensionalidade pode oferecer visualizações que auxiliam na identificação de semelhanças ou discrepâncias nos dados.

6 Conclusão

Os experimentos realizados com *embeddings* de comentários públicos da *Google Play Store* mostraram que tanto o PCA quanto o método proposto têm suas vantagens e limitações. O PCA destacou-se na análise de grupos ou *clusters* semânticos, por preservar de maneira eficaz a topologia dos vetores originais. Por outro lado, o método proposto foi mais eficaz na identificação de anomalias, destacando comentários que se distanciam significativamente dos demais no espaço vetorial, por preservar de maneira mais eficaz as distâncias relativas dos vetores originais.

A análise evidencia que a escolha entre o PCA e o método proposto deve ser guiada pelas necessidades específicas de cada aplicação. Enquanto o PCA é mais adequado para cenários onde a topologia dos dados é prioritária, o método proposto é mais vantajoso em casos que demandam maior precisão na preservação das distâncias entre os vetores originais, como na detecção de anomalias.

Em suma, a técnica apresentada não substitui métodos existentes, mas sim expande o conjunto de ferramentas disponíveis para a redução de dimensionalidade, oferecendo uma alternativa valiosa em aplicações que demandam maior fidelidade às relações originais entre vetores.

Estudos futuros podem explorar ajustes mais refinados ao algoritmo, focando em frentes não exploradas a fundo neste trabalho como tempo de convergência, diferentes algoritmos de descida de gradiente, outras inicializações dos vetores. Bem como sua aplicação em outros contextos e tipos de dados, como dados de música, fala e imagens, para ampliar ainda mais suas aplicabilidades. Também, explorar outras formas de comparar a performance do método proposto com os algoritmos existentes, incluindo os não explorados neste trabalho, como o t-SNE.

Referências

ASSOS, A. Convergence of mini-batch sgd. **Technical Report, Massachusetts Institute of Technology**, 2015. Disponível em: <https://web.mit.edu/people/assos/files/6_UAR.pdf>. Citado na página 16.

BISHOP, C. M. **Neural networks for pattern recognition**. [S.l.]: Oxford University Press, 1995. Citado na página 16.

DOH, S.; LEE, J.; JEONG, D.; NAM, J. Musical word embedding for music tagging and retrieval. **arXiv preprint arXiv:2404.13569**, 2024. Citado na página 31.

ETHAYARAJH, K. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. **arXiv preprint arXiv:1909.00512**, 2019. Citado na página 7.

Google AI. **Como treinar um classificador de texto usando embeddings**. 2023. Disponível em: <https://ai.google.dev/gemini-api/tutorials/text_classifier_embeddings?hl=pt-br>. Citado 2 vezes nas páginas 7 e 13.

_____. **Detecção de anomalias com embeddings**. 2023. Disponível em: <https://ai.google.dev/gemini-api/tutorials/anomaly_detection?hl=pt-br>. Citado 2 vezes nas páginas 7 e 13.

_____. **Embeddings na API Gemini**. 2023. Disponível em: <https://ai.google.dev/gemini-api/docs/embeddings?hl=pt-br>. Citado 2 vezes nas páginas 7 e 13.

GUPTA, S.; RANJAN, R.; SINGH, S. N. A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape and future directions. **arXiv preprint arXiv:2410.12837**, 2024. Citado na página 31.

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. **Science**, 2006. Citado na página 13.

Hugging Face. **Busca semântica com o FAISS**. 2023. Disponível em: <https://huggingface.co/learn/nlp-course/pt/chapter5/6>. Citado 2 vezes nas páginas 7 e 13.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2014. Citado na página 11.

_____. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2015. Citado 2 vezes nas páginas 11 e 12.

LIU, W.; SHAO, J.; MIN, H.; LIU, W.; LI, S. Deep learning for image retrieval: A survey. **Neurocomputing**, Elsevier, v. 406, p. 339–351, 2020. Citado na página 31.

MAATEN, L. Van der; HINTON, G. Visualizing data using t-sne. **Journal of Machine Learning Research**, 2008. Citado na página 13.

MAATEN, L. Van der; POSTMA, E.; HERIK, J. Van den. Dimensionality reduction: A comparative review. **Journal of Machine Learning Research**, v. 10, n. 66-71, p. 13, 2009. Citado 3 vezes nas páginas 7, 8 e 12.

_____. Dimensionality reduction: A comparative review. **Journal of Machine Learning Research**, 2009. Citado na página 13.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. In: **Proceedings of the International Conference on Learning Representations (ICLR)**. [s.n.], 2013. Disponível em: <https://arxiv.org/abs/1301.3781>. Citado na página 12.

OpenAI. **Introducing text and code embeddings**. 2022. Disponível em: <https://openai.com/index/introducing-text-and-code-embeddings/>. Citado na página 12.

_____. Gpt-4 technical report. **arXiv preprint arXiv:2303.08774**, 2023. Disponível em: <https://arxiv.org/abs/2303.08774>. Citado 2 vezes nas páginas 7 e 13.

_____. **New Embedding Models and API Updates**. 2024. Disponível em: <https://openai.com/index/new-embedding-models-and-api-updates/>. Citado 3 vezes nas páginas 7, 11 e 15.

REIMERS, N.; GUREVYCH, I. Sentence-bert: Sentence embeddings using siamese bert-networks. **arXiv preprint arXiv:1908.10084**, 2019. Citado 3 vezes nas páginas 7, 9 e 10.

SHLENS, J. A tutorial on principal component analysis. **arXiv preprint arXiv:1404.1100**, 2014. Disponível em: <https://arxiv.org/abs/1404.1100>. Citado na página 12.

TERVEN, J.; OUTROS. Loss functions and metrics in deep learning. **arXiv preprint arXiv:2307.02694**, 2023. Disponível em: <https://arxiv.org/abs/2307.02694>. Citado na página 11.

TERVONEN, J. Dimensionality reduction techniques for semantic embeddings: A comparative study. **Journal of Data Science**, 2023. Citado na página 13.

TOUVRON, H.; MARTIN, L.; STONE, K.; ALBERT, P.; ALMAHAIRI, A.; BABAEI, Y.; BASHLYKOV, N.; BATRA, S.; BHARGAVA, P.; BHOSALE, S. et al. Llama 2: Open foundation and fine-tuned chat models. **arXiv preprint arXiv:2307.09288**, 2023. Disponível em: <https://arxiv.org/abs/2307.09288>. Citado na página 13.

XIE, H.; VIRTANEN, T. Zero-shot audio classification via semantic embeddings. **arXiv preprint arXiv:2011.12133**, 2020. Citado na página 31.