UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA – CCET
DEPARTAMENTO DE COMPUTAÇÃO

# ANÁLISE E APLICAÇÕES DO ALGORÍTMO UMAP PARA CLASSIFICAÇÃO E REDUÇÃO DE DIMENSIONALIDADE DE CONJUNTOS DE DADOS COM MÚLTIPLAS VARIÁVEIS

São Carlos - SP
2023

Ariel Semensato Calixto

# Análise e aplicações do algoritmo UMAP para classificação e redução de dimensionalidade de conjuntos de dados com múltiplas variáveis

Trabalho de conclusão de curso para a graduação em Bacharelado em Engenharia de Computação;

Orientador:

Prof. Dr. Alexandre Luis Magalhães Levada

Banca avaliadora:

Prof. Dr. Alan Demétrius Baria Valejo

Prof. Dra. Heloísa de Arruda Camargo

São Carlos - SP

2023

# Resumo

Em muitas aplicações reais, o levantamento de informações por meio de instrumentos reais, sujeitos à ruídos e imperfeições tende a gerar conjuntos de dados onde o número de variáveis e componentes observados facilmente excede um número de dimensões onde a manipulação, o agrupamento e a classificação desses dados em conjuntos distintos baseados em suas similaridades se tornam não intuitivos ou extremamente custosos computacionalmente. Uma forma eficiente de preparar esses dados de maneira preemptiva para futuros processamentos e representações significativas é a redução de dimensionalidade, um processo que transforma um conjunto de dados de um espaço de muitas dimensões para um espaço de poucas dimensões de forma que o espaço de poucas dimensões ainda possui propriedades relevantes do conjunto.

Este trabalho propõe avaliar o atual estado da arte e estabelecer, utilizando critérios de desempenho, comparações entre os algoritmos de redução de dimensionalidade frequentemente utilizados atualmente e em um contexto histórico, com o foco principal no UMAP, um método que procura priorizar a classificação de conjuntos de dados localmente próximos por meio de suas características. Resultados foram obtidos utilizando diversos conjuntos de dados com propriedades diferentes, de forma a obter métricas relevantes sobre as influências que cada característica desses conjuntos possui nos resultados finais.

**Palavras-chave:** Redução de dimensionalidade. Classificação de conjuntos. Aprendizado de máquina. PCA. t-SNE. UMAP.

# Abstract

Through many real-life applications, gathering information through real instruments subject to noise and imperfections tends to generate datasets where the number of observed variables and components easily exceeds a number of dimensions where manipulation, clustering and classification of said data into distinct sets based on their similarities becomes either difficult or computationally expensive. An efficient way to preemptively prepare this data for further processing and meaningful representation is dimensionality reduction, a process that transforms a dataset from a high-dimensional space to a low-dimensional space such as the low-dimensional space still retains relevant properties from the original dataset.

This work proposes to evaluate the current state-of-the-art and establish, using performance criteria, comparisons between the frequently used dimensionality reduction used today and historically, with the main focus on UMAP, a method that seeks to prioritize the classification of data locally close by means of their characteristics. Results were obtained using different datasets with different properties, in order to obtain relevant metrics on the impact that each characteristic of these datasets has on the final results.

Keywords: Dimensionality reduction. Dataset classification. Machine learning. PCA. t-SNE. UMAP.

# Lista de Ilustrações

# Lista de Tabelas

# Lista de Abreviações

| | |
|---|---|
| **DR** | Dimensionality Reduction |
| **PCA** | Principal Component Analysis |
| **t-SNE** | t-Distributed Stochastic Neighbor Embedding |
| **UMAP** | Uniform Manifold Approximation and Projection |
| **CE** | Cross-Entropy |

# Sumário

# 1 Introdução

A interpretação de informações obtidas de por meio de instrumentos reais, sujeitos à ruídos e imperfeições traz inúmeros desafios para o campo de análise de dados. Estes conjuntos de dados, ou *datasets*, são complexos, podendo possuir múltiplas variáveis com relacionamentos entre si desconhecidos; desta forma, é difícil estabelecer um critério de classificação para esses conjuntos de dados em que se sabe muito pouco a respeito de sua composição.

A aplicação da redução de dimensionalidade nesses conjuntos possibilita a visualização e extração de informações relevantes atribuindo a cada elemento do conjunto a um grupo que compartilha características semelhantes, representando um conjunto de dados de alta dimensionalidade em um domínio de baixa dimensionalidade, de forma que as características do conjunto de dados são relativamente preservadas e a visualização de correlação entre variáveis e outros fenômenos que afetam os dados é mais simples e computacionalmente viável.

Com o objetivo de auxiliar o processo de aprendizado de máquina, foram projetados diversos algoritmos para a manipulação desses dados, notavelmente o algoritmo PCA (*Principal Component Analysis*, lit. Análise de Componente Principal), o algoritmo t-SNE (*T-distributed Stochastic Neighbor Embedding*, lit. incorporação de vizinhos estocástica distribuídos em T) e mais recentemente o algoritmo UMAP (*Uniform Manifold Approximation and Projection*, lit. Aproximação e Projeção Múltipla Uniforme).

O foco deste trabalho foi analisar e categorizar, quantitativamente e qualitativamente, o desempenho do algoritmo UMAP e suas vantagens e desvantagens quando comparado com outros algoritmos de redução de dimensionalidade, estabelecendo critérios de desempenho para os algoritmos como tempo de execução, dependência de condições iniciais, qualidade da resposta gerada, escalabilidade, entre outros, de forma que seja possível o julgamento da qualidade do algoritmo UMAP como uma ferramenta de redução de dimensionalidade, tal como possibilitar a comparação da performance do algoritmo UMAP com os algoritmos PCA e t-SNE, avaliando a evolução das ferramentas.

Para esta categorização, foram utilizados diversos conjuntos de dados que englobam diferentes desafios para os algoritmos em questão, modificando condições de

início dos dados, quantidade de dados coletados e número de variáveis do conjunto, de forma que seja possível estabelecer a influência que cada uma destas características possui sobre o resultado final.

# 2 Levantamento teórico e Estado da Arte

## 2.1 Redução de dimensionalidade

A redução de dimensionalidade (DR) é uma técnica que consiste em desenvolver uma representação em dimensões reduzidas de um conjunto de dados de altas dimensões (ou componentes), sendo que esta é uma etapa que muitas vezes precede qualquer tipo de técnica de reconhecimento de padrões, categorização ou processamento de dados sobre um conjunto.

Em um modelo ideal, a DR não acarretaria em perda de informações e seria capaz de representar fidedignamente qualquer conjunto de dados em menos dimensões; no entanto, perda de informações sobre relacionamentos espaciais dos dados do conjunto são inevitáveis em um contexto de aplicação real. Por conta disso, o objetivo final da DR é proporcionar uma transformação de forma que seja possível obter os mesmos resultados e atingir as mesmas conclusões tratando tanto o conjunto reduzido quanto o conjunto original. Desta forma, a DR procura estabelecer uma relação de equivalência entre os conjuntos, e não uma relação de igualdade, com a qualidade desta equivalência dependente das técnicas utilizadas para pré-processar o conjunto de dados em questão.

Atualmente, o algoritmo UMAP é considerado o estado da arte para DR, efetivamente substituindo os casos de uso onde se utilizaria o t-SNE (MCINNES *et al*, 2018). No entanto, o PCA ainda é muito utilizado por ter características extremamente valiosas como linearidade e a preservação de características globais do conjunto; por conta disso, a utilização de um algoritmo em detrimento do outro deve levar em conta objetivos finais ao manipular os dados, características desejadas da transformação final, escalabilidade temporal, e os estados iniciais tal como as heurísticas que serão aplicadas ao conjunto antes do seu processamento.

### 2.1.1 A “maldição da dimensionalidade”

Uma outra vantagem da aplicação da DR é a introdução de uma solução para a chamada “maldição da dimensionalidade” (*curse of dimensionsionality*), termo cunhado por Richard E. Bellman que descreve a presença de vários fenômenos decorrentes da

análise de espaços de altas dimensões que não ocorrem em espaços de baixas dimensões, como os de duas ou três dimensões que são habitualmente utilizadas para representação de dados. O problema descreve que, conforme dimensões são adicionadas no espaço de análise, o volume total do espaço cresce tão rapidamente que os dados relevantes neste espaço se tornam esparsos e dissimilares. Desta forma, é necessária uma quantidade exponencialmente maior de dados para que seja possível obter informações relevantes destes conjuntos (WANG *et al*, 2020).

Os métodos de DR procuram primeiramente estabelecer a dimensionalidade real do conjunto, descartando dimensões artificiais ou irrelevantes (ou seja, atributos que não possuem real influência sobre o conjunto de dados sendo analisado) e então transformar esses conjuntos de forma que haja a menor perda de informações relevantes possível.

### 2.1.2 Função de perda

A função de perda (ou *loss function*) é uma função ou método que possui como objetivo avaliar quão bem um algoritmo modela um conjunto de dados, comparando o resultado obtido com o resultado esperado final. Para algoritmos que utilizam o aprendizado de máquina para seu funcionamento, como o t-SNE e o UMAP, é necessário atribuir uma métrica que possibilite estabelecer uma relação entre os resultados obtidos e os esperados, de forma a medir a qualidade da resposta gerada.

O algoritmo t-SNE utiliza como sua função de perda a *Kullback-Leibler divergence* (*KL-divergence*), também chamada de entropia relativa, que quantifica as informações perdidas quando uma distribuição Q é utilizada para aproximar o de uma distribuição P. Esta divergência possui um valor entre 0 e 1, onde 0 indicaria uma transposição ideal entre as distribuições e 1 indicaria uma transposição completamente heterogênea entre as distribuições. Desta forma, deseja-se aproximar o valor da a função ao seu valor objetivo o máximo possível. Nota-se também que esta relação não é simétrica: ou seja, a divergência de Q aproximado a P não é necessariamente igual à divergência de P aproximado a Q (WANG *et al*, 2020).

Já o algoritmo UMAP não possui uma função de perda bem definida, e pode se adaptar a uma função de perda arbitrariamente escolhida. No entanto, por padrão, assume-se que o UMAP utiliza como sua função de perda a *Cross-Entropy* (CE), uma função que mede, similarmente à *KL-divergence*, a diferença entre duas distribuições

probabilísticas (MCINNES *et al*, 2018).. No contexto desta função, Entropia é definida como sendo a quantidade de dados necessários para transmitir informações sobre um evento aleatório de uma distribuição – uma distribuição homogênea tem alta entropia, enquanto uma distribuição com probabilidades heterogêneas possui uma baixa entropia. Esses dados podem tomar forma de um número inteiro, um número real, ou no caso do UMAP, a quantidade de bits utilizada para representar este número. A entropia cruzada, ou CE, é uma expansão deste conceito que passa a calcular a quantidade de bits necessária para transmitir um evento aleatório de uma distribuição P para uma distribuição Q. O resultado é um número positivo, medido em *bits*, que é igual à entropia da distribuição original, caso as duas distribuições sejam idênticas (DAMRICH *et al*, 2021).

## 2.2 Algoritmos utilizados neste trabalho

### 2.2.1 PCA (Principal Component Analysis)

O PCA é uma técnica estatística de DR, concebido em 1901, que procura manter a maior quantidade de informação de um conjunto ao reduzi-lo. O PCA é notável por duas características principais: a manutenção de distâncias globais entre os elementos do conjunto de dados; e a sua linearidade, o que permite que o algoritmo seja extremamente eficaz independentemente da quantidade de dados a serem processados.

O seu funcionamento consiste em fazer transformações lineares do conjunto original em um novo conjunto com menos coordenadas, onde as variações (ou distâncias) dos dados são mantidas em relação ao conjunto inicial; desta forma, o PCA pode ser interpretado como uma espécie de projeção de um espaço altamente dimensional em um espaço de baixa dimensões, mantendo da melhor forma possível as distâncias globais entre os elementos do conjunto. A implementação do PCA se dá da seguinte forma:

**Algoritmo 1: PCA**

**Início**

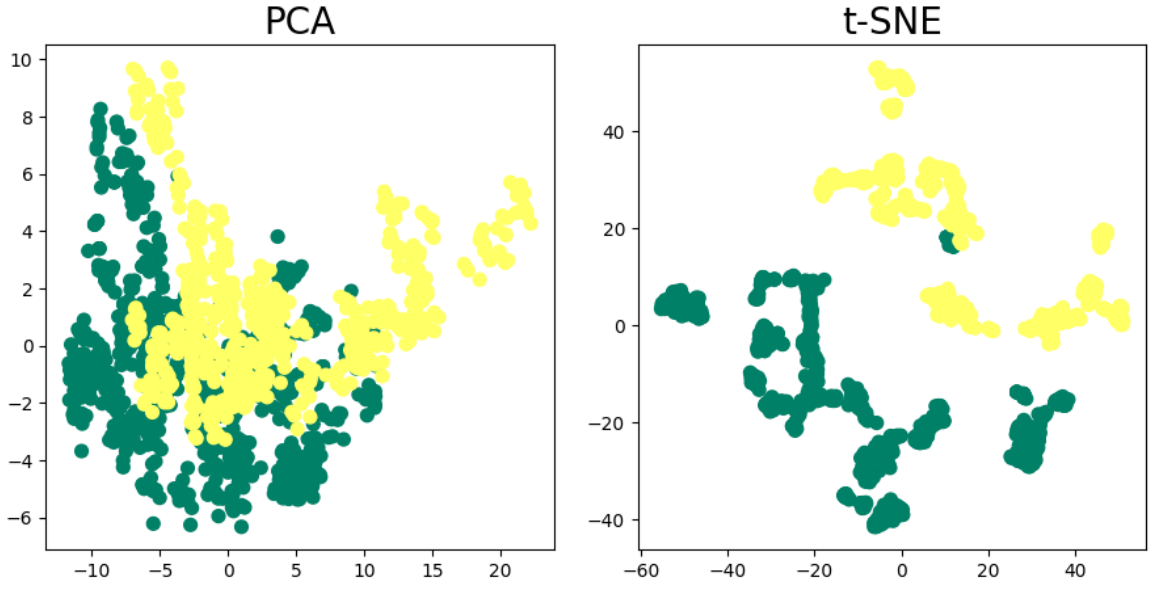
1. Padronização do conjunto de dados na matriz original $A$.
2. Cálculo da matriz de covariância para as componentes do conjunto.
3. Cálculo dos autovalores e autovetores da matriz de covariância.
4. Ordenação dos autovalores com seus respectivos autovetores.
5. Selecionar $k$ autovalores e criar uma nova matriz $B$ de autovetores.
6. Transformar a matriz original $A$ utilizando a matriz $B$, obtendo-se o resultado final.

**Fim**

### 2.2.2 t-SNE (t-distributed Stochastic Neighbor Embedding)

O algoritmo t-SNE foi desenvolvido em 2008 por Laurens van der Maaten e Geoffrey Hinton, e é uma outra técnica estatística de DR que difere do PCA no sentido em que o processo de redução de dimensões é não-linear; isso permite que o t-SNE possa separar e categorizar conjuntos de dados que normalmente não podem ser tratados por um algoritmo de redução linear como o PCA, permitindo que se extraia dados de relacionamento mais relevantes em detrimento da perda da distribuição desses dados no espaço global. O t-SNE consegue esses resultados por meio de agrupamento de vizinhos similarmente próximos baseado em suas componentes e as suas similaridades.

Figura 1 – Comparação da aplicação do PCA e t-SNE sobre um conjunto não-linear (banknote).



Como pode ser observado na figura 1, embora o PCA ainda consiga transpor o espaço altamente dimensional em um espaço de baixas dimensões, nota-se que sua representação não permite extrair uma quantidade de informações relevantes sobre todos os elementos do conjunto; em contrapartida, o efeito mais notável do t-SNE é o agrupamento (ou *clustering*) desses dados em conjuntos mais claramente bem definidos, porém sacrificando a estrutura global do conjunto do processo; ou seja, as distâncias entre pontos após a aplicação do t-SNE não representam suas distâncias reais, mas sim sua similaridade estatística com outros elementos vizinhos, onde pontos são alocados em espaços próximos devido às similaridades de seus componentes. Isso permite concluir que pontos dentro de um agrupamento são similares, porém não é possível tirar conclusões do relacionamento espacial entre dois ou mais agrupamentos diferentes. O funcionamento geral do t-SNE se dá da seguinte forma:

**Algoritmo 2: t-SNE**

**Início**

1 Etapa 1: construção da distribuição probabilística.
- 1.1 Para cada par de elementos, é atribuída uma probabilidade com base na similaridade de suas componentes; quanto maior a similaridade, maior a probabilidade.
- 1.2 Essas probabilidades são normalizadas entre todos os pontos, de forma que pontos próximos se agrupam em conjuntos distintos.

2 Etapa 2: construção da distribuição probabilística no mapa de baixas dimensões.
- 2.1 Para cada elemento, é atribuído uma nova posição no novo espaço de baixas dimensões, de forma a minimizar a divergência (ou seja, a função de perda *KL-divergence*) entre as duas distribuições obtidas, utilizando a distância euclidiana.

**Fim**

Notavelmente, o t-SNE possui uma outra limitação além da perda da estrutura global dos dados: o algoritmo não é escalável, e para conjuntos particularmente grandes com milhões de elementos e componentes, sua natureza exponencial o torna extremamente ineficiente.

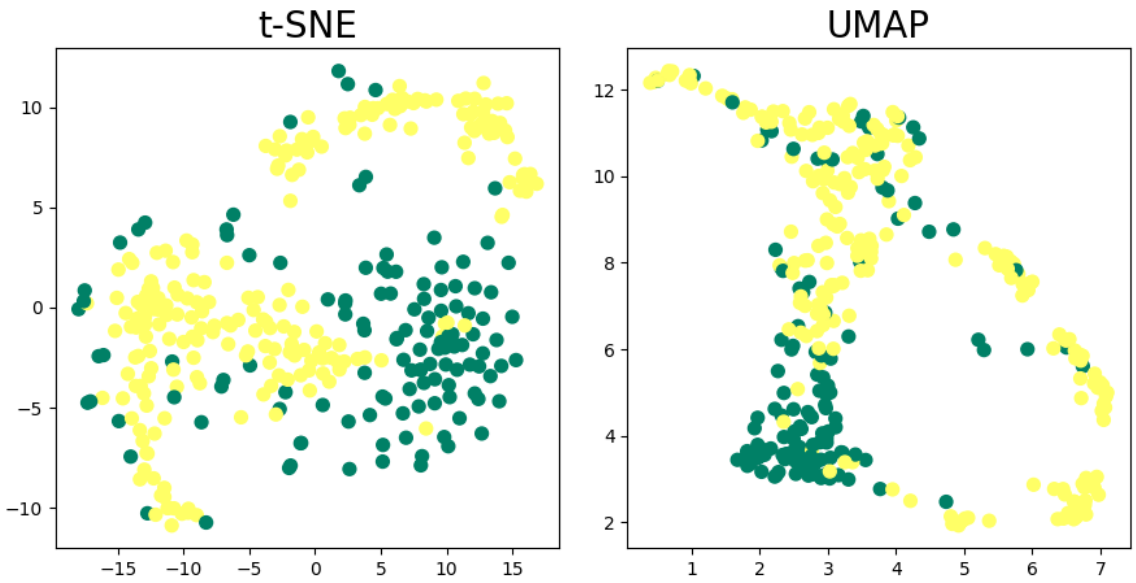
### 2.2.3 UMAP (Uniform Manifold Approximation and Projection)

O UMAP é um algoritmo de DR que utiliza técnicas de aprendizado de máquina, similar ao t-SNE. Comparado ao seu predecessor t-SNE, suas diferenças não são imediatamente óbvias: ambos algoritmos são utilizados em transformações não-lineares e ambos algoritmos possuem como seu foco principal o agrupamento (*clustering*) dos dados a partir das semelhanças de suas componentes. O UMAP, no entanto, apresenta duas principais vantagens sobre o t-SNE:

- Uma maior preservação de características estruturais globais do conjunto de dados, onde os resultados obtidos por meio do UMAP permitem com maior confiabilidade estabelecer relacionamentos entre agrupamentos de dados distintos baseado em sua posição espacial.
- Uma escalabilidade muito maior quando comparado ao t-SNE, permitindo que o algoritmo seja aplicado efetivamente sobre uma variedade muito maior de conjuntos de dados.

A figura 2 a seguir ilustra a diferença da aplicação dos algoritmos t-SNE e o algoritmo UMAP sobre o conjunto ionosphere (descrito a seguir no capítulo 2.3.1). Enquanto o t-SNE preocupa-se em criar regiões de dados estatisticamente semelhantes, o UMAP aplica seu método que mantém as distâncias globais relativamente conservadas, o que neste caso gera uma representação muito mais imediatamente útil do conjunto em questão.

Figura 2 – Comparação da aplicação do t-SNE e UMAP sobre um conjunto não-linear, composto de dados atmosféricos (ionosphere).



A sua implementação é feita da seguinte forma:

**Algoritmo 3: UMAP**

**Início**

1 Etapa 1: construção da distribuição probabilística.

1.1 Para a distribuição inicial, é construída uma representação topológica *fuzzy.* Partindo de uma categorização inicial, os conjuntos são alterados, onde distâncias entre os pontos passa de ser o fator que determina se um elemento pertence ao conjunto ou não, com um valor entre 0 e 1

2 Etapa 2: construção da distribuição probabilística no mapa de baixas dimensões.

2.1 Para cada elemento, é atribuído uma nova posição no novo espaço de baixas dimensões, de forma que o resultado final da entropia cruzada do conjunto seja a mais próxima possível do conjunto inicial.

**Fim**

Sendo baseado em aprendizado de máquina, o UMAP produz um resultado diferente a cada execução e, portanto, pode levar a interpretações diferentes de resultados sobre um mesmo conjunto. Notavelmente isso pode se tornar um empecilho quanto a reprodutibilidade de resultados experimentais.

## 2.3 Redução de dimensionalidade aplicada a conjuntos de dados

Um conjunto de dados é uma coleção de elementos com seus respectivos atributos (ou componentes). Para cada elemento se representa por meio de variáveis as características do dado em questão, que pode ser um número inteiro, um número real, um elemento binário, uma etiqueta (ou *label*) binária ou pertencente a um conjunto de possíveis valores.

Um conjunto altamente dimensional em geral não pode ser representado visualmente com facilidade, e estas representações não possibilitam extrair informações relevantes dos dados do conjunto, como se relacionam as suas características e o relacionamento dos dados entre si.

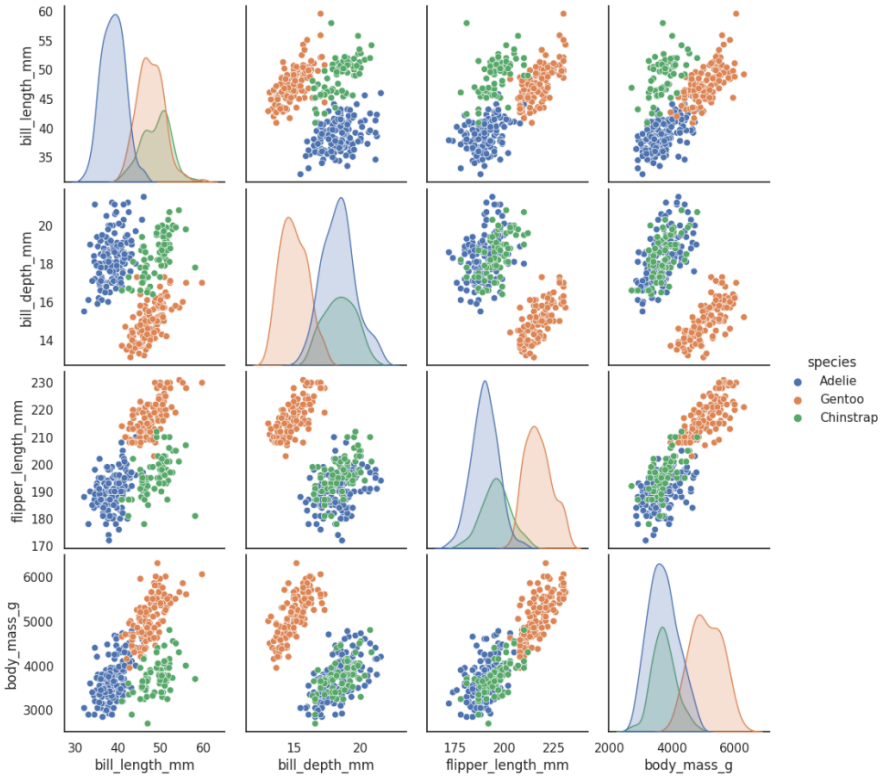
Conforme conjuntos agregam características, o volume do espaço aumenta exponencialmente, e as dificuldades de representação e processamento do conjunto acompanham esse aumento exponencial. Comumente, considera-se um espaço altamente dimensional um espaço que possui dez ou mais dimensões (ou características), momento em que qualquer tentativa de representação e processamento sobre o conjunto original se torna difícil, se não inviável.

O objetivo final da DR é proporcionar uma equivalência de forma que os resultados obtidos ao processar os dados em ambos espaços sejam equivalentes, ou seja, os mesmos objetivos e conclusões possam ser obtidas ao tratar ambos conjuntos; dadas estas suposições, uma DR bem-sucedida proporciona um imenso ganho em poder de processamento, escalabilidade e praticidade computacional (WANG *et al*, 2020).

Em termos técnicos, a DR possibilita comprimir o conjunto original, ocupando menos espaço em memória e armazenamento, aumentar a velocidade de processamento e cálculos sobre o conjunto e remover componentes redundantes ou irrelevantes para a análise dos dados.

Em contrapartida, a DR pode necessitar de uma alta quantidade de poder de processamento e memória, além de criar uma nova interpretação do conjunto de dados original, onde variáveis independentes podem levar a resultados de difícil compreensão. No entanto, o aspecto negativo mais relevante é que informações são perdidas no processo de DR; notavelmente, o espaço gerado não mais representa o conjunto de dados original, mas sim a própria representação em altas dimensões; desta forma, o espaço em baixas dimensões não possui uma relação direta com o conjunto de dados original, e em geral, não é uma operação reversível. Por conta disto, a escalabilidade e o tempo de execução dos métodos de DR são aspectos extremamente importantes de forma que o conjunto original deve ser re-processado em, por exemplo, aplicações que implementam aprendizado de máquina.

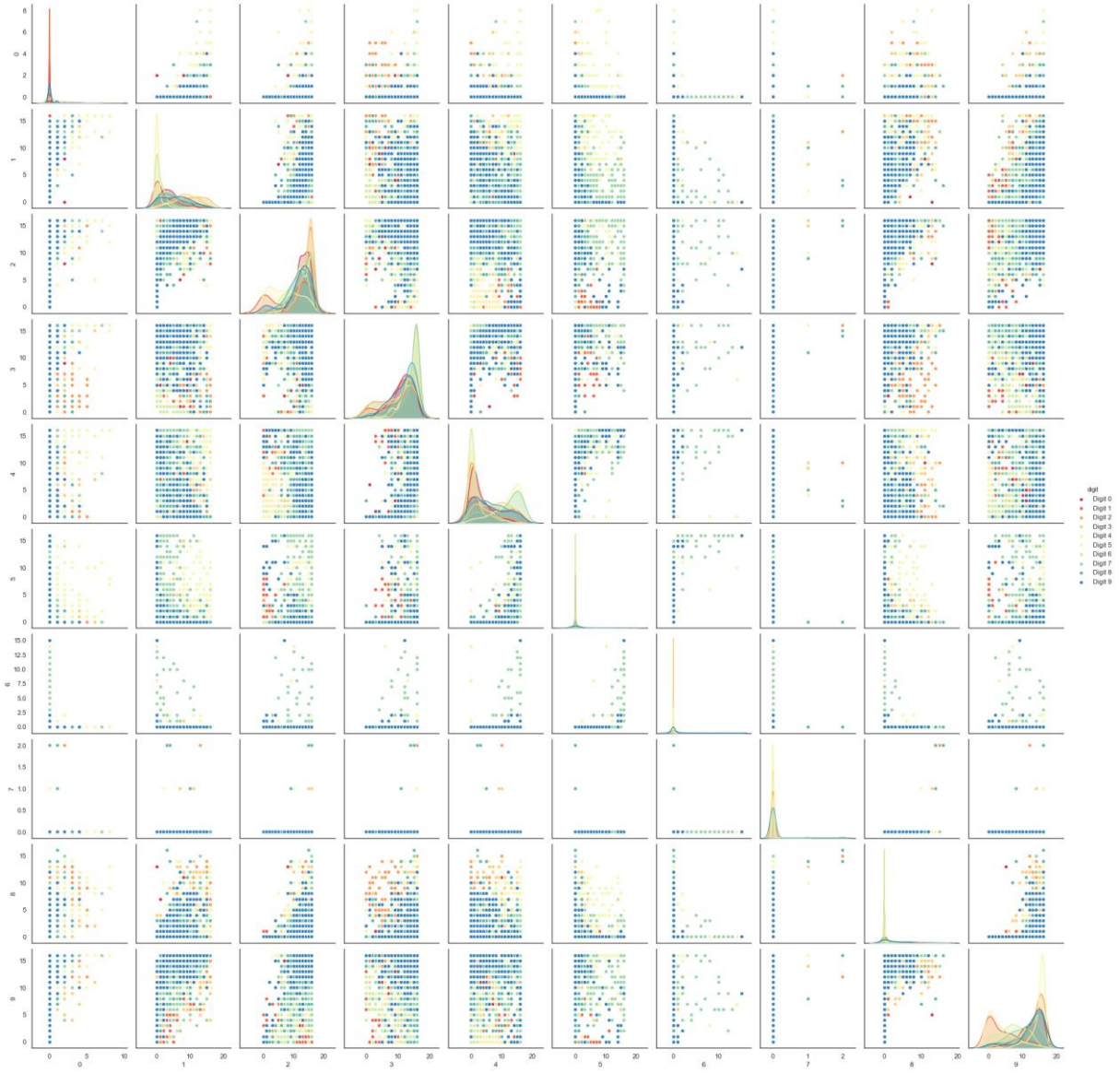
Figura 3 – Representação de um conjunto de dados com 150 elementos, 4 dimensões e 3 classes com suas componentes pareadas.




Fonte: https://umap-learn.readthedocs.io/en/latest/basic_usage.html. Acesso em 02/02/2023

Como pode-se observar na Figura 3, uma representação de um conjunto relativamente pequeno, mas ainda não-representável em baixas dimensões pode ainda ser viável para fins de extração de informações e estabelecimento de relacionamento entre as variáveis; no entanto, como ilustra a Figura 4 a seguir, uma quantidade relativamente ainda baixa de componentes (neste caso, 10) aumenta a complexidade de análise e extração de informações de forma exponencial, onde não é possível

Figura 4 – Representação de um conjunto de dados com 1700 elementos, 10 dimensões e 2 classes com componentes pareadas.



Fonte: https://umap-learn.readthedocs.io/en/latest/basic_usage.html. Acesso em 13/02/2023

Essas dificuldades (abordadas anteriormente no capítulo 2.1.1) criam a necessidade de ferramentas que façam o pré-processamento desses conjuntos, de forma a possibilitar um ambiente mais propício para o processamento e manipulação das informações que os dados originais representam.

### 2.3.1 Conjuntos de dados utilizados neste trabalho

Para fins de avaliação da performance de cada algoritmo, foram selecionados conjuntos de dados de variados tipos e tamanhos, explorando quantidades diferentes de

características e elementos de forma a avaliar o impacto destes sobre o resultado final. A seleção foi feita abordando um número de classes, elementos e componentes variados para determinar qual impacto cada aspecto de um conjunto de dados tem sobre o resultado final e sobre a performance do algoritmo.

Tabela 1 – Conjuntos utilizados para a avaliação dos algoritmos usados neste trabalho

| Conjuntos de dados | n° de elementos | n° de componentes | n° de classes |
|---|---|---|---|
| iris | 150 | 4 | 3 |
| ionosphere | 350 | 34 | 2 |
| bank-note | 1372 | 4 | 2 |
| pendigits | 7493 | 16 | 10 |
| htru2 | 17897 | 8 | 2 |
| swarm-aligned | 24016 | 2400 | 2 |
| mnist | 60000 | 784 | 10 |
| fashion-mnist | 60000 | 784 | 10 |
| postures | 78095 | 37 | 5 |
| accelerometer | 153000 | 4 | 5 |
| poker-hand | 1000000 | 10 | 10 |

# 3 Algoritmo UMAP

A implementação do UMAP, tal como dos algoritmos PCA e t-SNE também utilizados neste trabalho, foi feita em linguagem Python, devido ao seu natural suporte ao manuseamento de conjuntos de dados grandes. Esta seção irá se aprofundar na implementação de cada etapa do UMAP.

Inicialmente, precisamos de um conjunto de dados para a aplicação do algoritmo. Para fins demonstrativos, foi utilizado o conjunto de dados "íris". Por definição do UMAP, vamos mapear funções essenciais para o seu funcionamento, que são a função de probabilidade em altas dimensões, a função de probabilidade em baixas dimensões, e a função de perda CE (DAMRICH *et al*, 2021)..

O UMAP, em sua implementação, permite que sejam utilizadas quaisquer metodologias de aprendizado de máquina de forma a aproximar o valor da função de perda ao valor objetivo. Para a implementação desta iteração do algoritmo, foi utilizado o método do gradiente descendente.

A função de probabilidade em altas dimensões é dada pela seguinte formula:

$$p_{i|j} = e^{-\frac{d(x_i,x_j)-\rho_i}{\sigma_i}} \quad (1)$$

Esta formula estabelece a probabilidade P de um elemento i para um elemento j do conjunto original, onde i e j são elementos distintos arbitrários, por meio da distância euclidiana d.

Em uma situação ideal, o valor de $\sigma_i$ é fornecido, onde $\sigma_i$ é o valor de normalização sobre a distribuição. Um valor alto de $\sigma_i$ gerará uma resposta com maior dispersão dos dados, mantendo as distâncias globais mais fidedignas em detrimento do agrupamento dos dados semelhantes (MCINNES *et al*, 2018). Como para este conjunto de dados não possuímos o valor de $\sigma_i$, devemos obter uma aproximação apropriada. Utilizando a definição do número de vizinhos próximos de um elemento, e fixando este valor, podemos chegar a uma aproximação do possível valor de $\sigma_i$ para cada elemento do conjunto de dados. Esta aproximação foi implementada por meio de uma busca binária, de forma a aumentar a eficiência do algoritmo. Com o valor de $\sigma_i$, é possível calcular a probabilidade entre os

elementos no espaço altamente dimensional e construir a matriz de probabilidades final. Esta matriz deve satisfazer a condição de simetria imposta pelo UMAP, da seguinte forma:

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i} \tag{2}$$

Onde $p_{ij}$ é o elemento final da matriz P, construído por meio dos compoenetes encontrados pela formula anterior. Com estes dados obtidos, criamos o novo espaço de baixas dimensões onde cada elemento q do conjunto é definido da seguinte maneira:

$$q_{ij} = \left(1 + a\left(y_i - y_j\right)^{2b}\right)^{-1} \tag{3}$$

Utilizando ambas matrizes P e Q, é possível agora aplicar a transformação do espaço P para o espaço Q e utilizar a função de perda CE para avaliar a qualidade dos resultados obtidos em cada iteração. A função de perda é definida pela formula (MCINNES *et al*, 2018, p. 20):

$$C\big((A,\mu),(A,v)\big) = \sum_{a\in A}((\mu(a)\log\big(\mu(a)\big) + (1-\mu(a)\log\big(1-\mu(a)\big)) - \sum_{a\in A}((v(a)\log\big(v(a)\big) + (1-\mu(a)\log\big(1-v(a)\big)) \tag{4}$$

A primeira parte da equação que define a CE depende somente do um elemento μ que possui um valor fixo durante o processo de otimização da função de perda, portanto, o nosso interesse está apenas em minimizar a segunda parte da equação, que compreende o segundo somatório. Conforme esta segunda parte é reduzida por meio de iterações, a entropia do conjunto se aproxima de valores ideais e a transformação se aperfeiçoa.

A estrutura final do algoritmo portanto é a seguinte:

**Algoritmo 4: Implementação do UMAP**

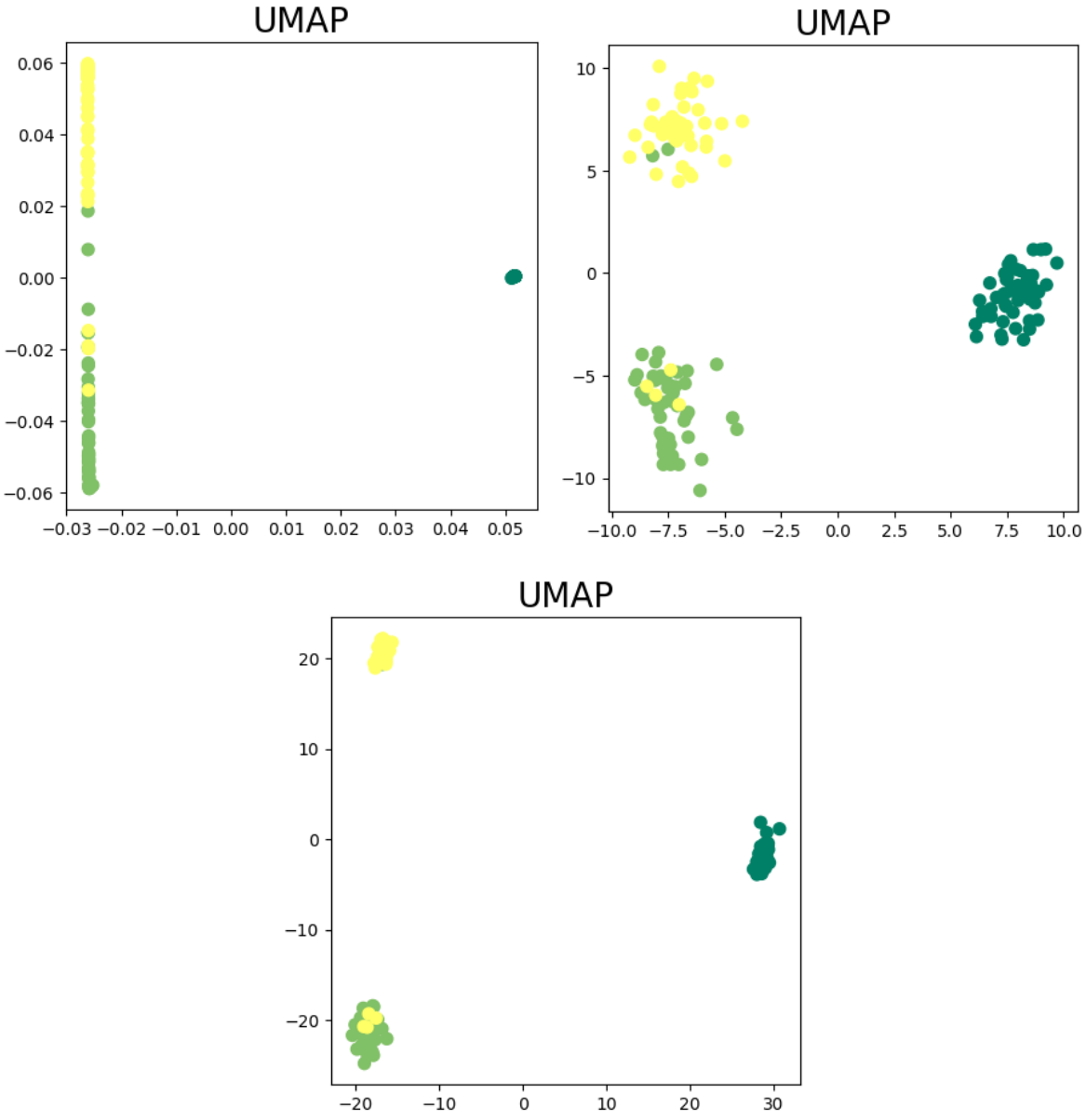
**Início**

1. Construir uma matriz de probabilidades M de tamanho n,n, computando as distâncias euclidianas de cada elemento $a_{i,j}$ para cada elemento $a_{k,l}$ da matriz original, com i, j, k, l variando de 1 à n.
2. Para cada elemento **i** inferir, utilizando um valor estático adequado de vizinhos próximos k, um valor aproximado de $\sigma_i$. Com este valor de $\sigma_i$, calcular a probabilidade **p** de cada elemento $a_{i,j}$ para cada elemento $b_{k,l}$.
3. Construir a matriz de probabilidades de altas dimensões P, utilizando a condição de simetria.
4. Construir a matriz de probabilidades de baixas dimensões Q.
5. Utilizando o método do gradiente descendente, com a função de perda CE como objetivo, executamos uma série de transformações sucessivas do conjunto P para o espaço do conjunto Q, por uma quantidade definida de iterações.
6. Apresentar os resultados finais da transformação do espaço de altas dimensões.

**Fim**

Particularmente, um dos pontos mais importantes do UMAP, tal como a maior influência no seu tempo de execução, é a quantidade de iterações, ou gerações, que o algoritmo irá executar sobre um conjunto de dados. É possível estabelecer métricas de forma que o algoritmo cesse a execução após um número de iterações sem mudanças significativas, tal como também é possível fazer com que o algoritmo seja executado uma quantidade pré-definida de vezes. O número de iterações possui influência direta na usabilidade e qualidade da resposta final, como pode ser observado na figura 4.

Figura 5 – Resultados da aplicação do algoritmo UMAP sobre o conjunto de dados íris por 1, 100 e 1000 iterações



# 4 Experimentos e resultados

Para a avaliação do desempenho e obtenção das métricas propostas neste trabalho, foram utilizados os conjuntos de dados descritos no capítulo 2.3.1. Os algoritmos foram implementados na linguagem Python, devido à facilidade natural que a linguagem possui para manipulação destes conjuntos, tal como a disponibilidade de bibliotecas voltadas a aprendizado de máquina e operações com matrizes e espaços de altas dimensões.

As bibliotecas utilizadas nestes experimentos foram a NumPy, a MatPlotLib, a SciPy e a Sciki-Learn. Para fins dos testes a seguir, foram utilizadas pré-implementações destas bibliotecas para os algoritmos PCA e t-SNE, e a implementação manual do algoritmo UMAP, descrita no capítulo 3. Para todos conjuntos, as transformações foram feitas do espaço original para um espaço bidimensional.

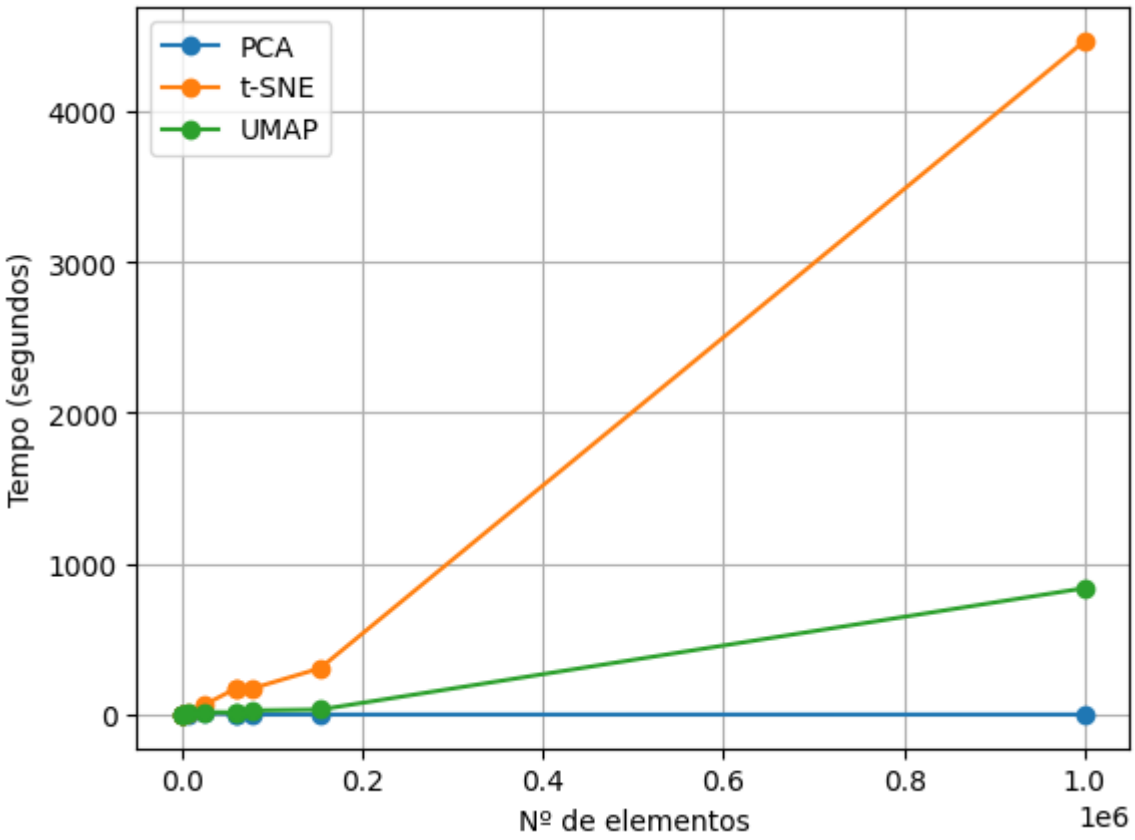
## 4.1 Avaliação temporal e escalabilidade

Para fins de avaliar a escalabilidade dos algoritmos, foram coletados os tempos de execução de cada transformação e suas representações resultantes no espaço bidimensional. O resultado do tempo de execução de cada algoritmo aplicado sobre cada conjunto de dados pode ser observado na tabela abaixo.

Tabela 2 – Resultados dos tempos de execução dos algoritmos sobre os conjuntos de dados experimentais.

| Conjunto de dados | PCA (s) | t--SNE (s) | UMAP (s) | Performance relativa (%) (t-SNE/UMAP) |
|---|---|---|---|---|
| iris | 0,001 | **0,296** | 1,494 | 19,813 |
| ionosphere | 0,002 | **0,768** | 1,468 | 52,316 |
| bank-note | 0,0004 | **2,161** | 2,456 | 87,989 |
| pendigits | 0,002 | 13,992 | **11,846** | 118,116 |
| htru2 | 0,004 | 40,197 | **3,967** | 1013,285 |
| swarm-aligned | 0,004 | 61,902 | **16,210** | 381,875 |
| mnist | 11,142 | 173,712 | **13,759** | 1262,534 |
| fashion-mnist | 3,911 | 173,719 | **14,011** | 1239,876 |
| postures | 3,683 | 172,822 | **27,126** | 637,108 |
| accelerometer | 0,077 | 306,431 | **32,782** | 934,754 |
| poker-hand | 0,291 | 4465,758 | **837,878** | 532,984 |

Figura 6 – Distribuição temporal do tempo de execução de cada um dos algoritmos testados.




Como esperado, o algoritmo PCA, sendo uma transformação linear, manteve seu tempo de execução constante por todos os testes. Observando os outros dois algoritmos experimentalmente, enquanto o algoritmo t-SNE possui um melhor tempo de execução para conjuntos muito pequenos, a sua natureza de tempo exponencial faz com que a escalabilidade do algoritmo sofra imensamente quando conjuntos começam a incorporar uma quantidade significativa de elementos.

Por meio dos resultados obtidos pelo processamento dos conjuntos swarm-aligned, mnist (ou fashion-mnist) e postures, conclui-se que a dependência temporal do algoritmo t-SNE se dá primariamente devido ao número de elementos; considerando a quantidade de dados brutos de cada um dos conjuntos (ou seja, o número de componentes vezes o número de elementos, 57.638.400, 47.040.000 e 2.889.515, respectivamente), o volume de dados decresce, apesar do gradativo aumento de número de elementos. Observa-se, no entanto, que o tempo de processamento respeita um aumento exponencial baseado apenas na quantidade de elementos de cada conjunto. Por conta deste fenômeno, pode-se inferir que

o algoritmo t-SNE é melhor aplicado em conjuntos com uma quantidade baixa de elementos, mas uma quantidade alta de componentes.

O algoritmo UMAP, em contrapartida, possui um tempo de execução base mais alto, mas sua vantagem temporal rapidamente se sobressai sobre o t-SNE conforme a quantidade de elementos aumenta, onde o tempo de execução observado foi em média 5,71 vezes menor quando comparado ao t-SNE. Enquanto a dependência temporal do UMAP também se dá primariamente pelo número total de elementos, é possível observar uma mais direta influência do número de componentes sobre os resultados finais, observando os conjuntos pendigits e htru2, e os conjuntos swarm-aligned e mnist

.

## 4.2 Manipulação de dados não-lineares

Para esta parte do experimento, foram utilizados 4 conjuntos de dados: íris, pendigits, bank-note e mnist. O conjunto de dados íris é notavelmente linear e facilmente representado com poucas transformações, enquanto o conjunto de dados mnist é completamente não-linear. Os conjuntos de dados bank-note e pendigits são conjuntos intermediários, onde a manipulação espacial permite a eliminação de variáveis que possam ser redundantes e características irrelevantes para a análise da transformação final.

Figura 7 – Conjunto de dados íris processado pelos algoritmos PCA, t-SNE e UMAP.

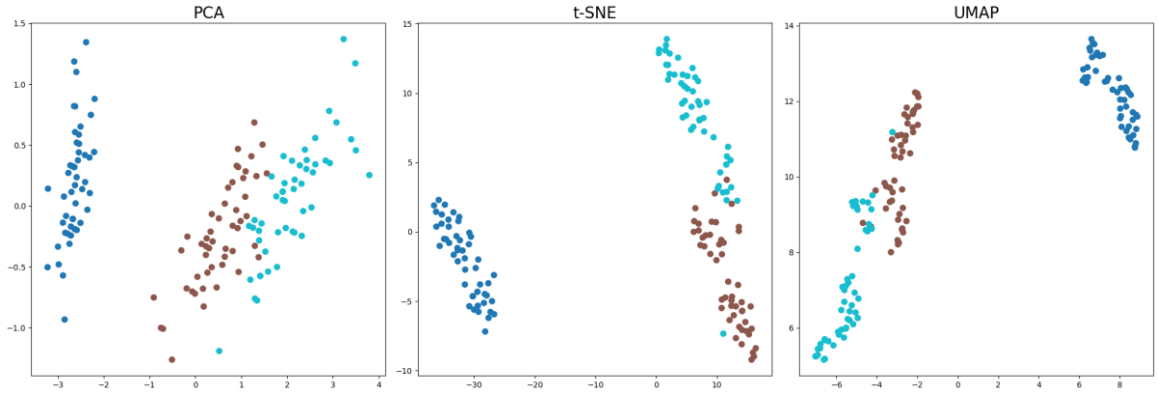


Figura 8 – Conjunto de dados bank-note processado pelos algoritmos PCA, t-SNE e UMAP.

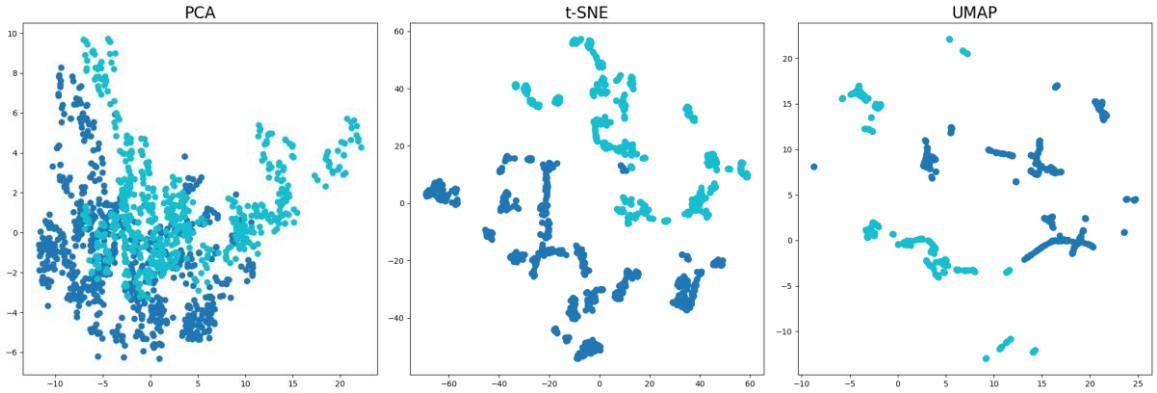
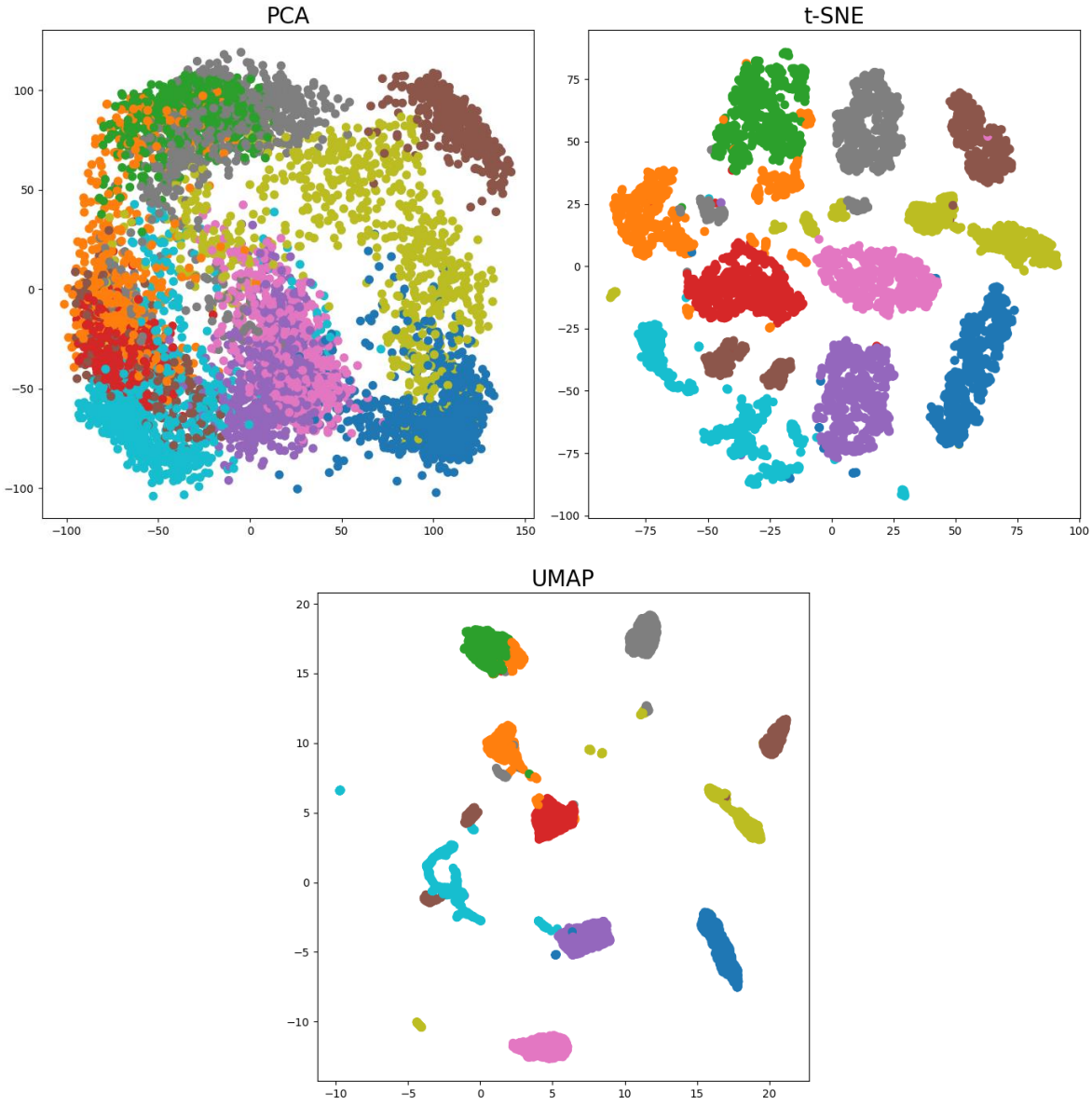


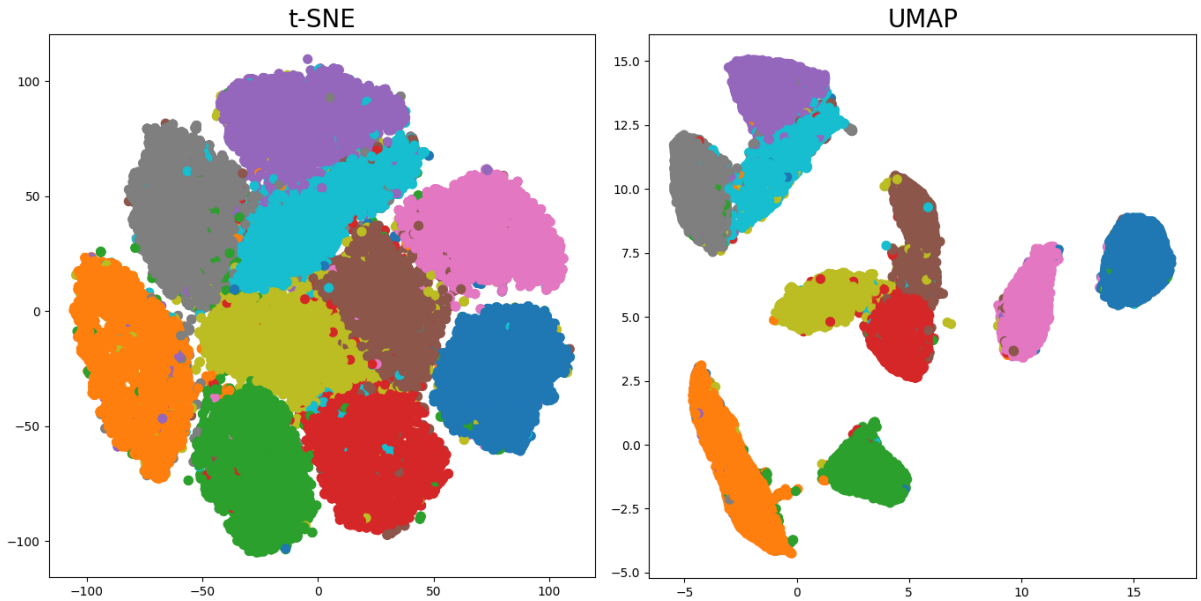

Figura 9 – Conjunto de dados pendigits processado pelos algoritmos PCA, t-SNE e UMAP.



As representações obtidas a partir das transformações dos conjuntos demonstram com clareza a fraqueza do método PCA quanto a transformação de espaços não-lineares: enquanto o algoritmo consegue aplicar a DR efetivamente sobre o conjunto íris e com razoável sucesso sobre os conjuntos pendigits e bank-note, o algoritmo não é capaz de criar uma representação do conjunto mnist de forma eficaz em duas dimensões.

Os algoritmos t-SNE e UMAP partilham em grande parte a sua taxa de sucesso; no entanto, nota-se que o algoritmo t-SNE falha em criar uma boa representação do espaço linear, de forma que, como as distâncias globais são perdidas no processo de agrupamento dos dados, o conjunto resultante apresenta um modelo do conjunto onde vizinhos estão estatisticamente próximos. Como o UMAP proporciona a possibilidade da manutenção de distâncias globais, a sua representação do conjunto íris é muito similar à do algoritmo PCA. Esta característica do UMAP é mais notável quando utilizado sobre o conjunto mnist, onde o estabelecimento das distâncias globais proporciona um resultado extremamente diferente quando comparado ao t-SNE, onde podemos ver os conjuntos semelhantes que representam os números 0, 6 e 9 estão agrupados na região noroeste do gráfico devido às suas semelhanças (todas representações possuem um único círculo em sua composição); em contrapartida, os conjuntos que representam os número 1 e 4 estão relativamente isolados na região leste do gráfico, com sua proximidade relativa e, ao mesmo tempo, sua distância dos outros conjuntos indicando a ausência de curvas nestes conjuntos em específico.

Figura 10 – Conjunto de dados mnist processado pelos algoritmos t-SNE e UMAP.



## 4.3 Inicialização

Para a realização do experimento sobre a performance dos algoritmos dadas condições iniciais, forma utilizados dois conjuntos de dados: mnist e fashion-mnist. A escolha foi feita devido aos conjuntos de dados possuírem a mesma quantidade de elementos e características, tal como compartilharem a mesma estrutura de dados, apesar das informações presentes em cada um destes conjuntos serem essencialmente completamente diferentes.

Neste experimento, os elementos e características dos conjuntos foram aleatorizados sequencialmente cinquenta vezes para cada conjunto de dados e para cada algoritmo. Em seguida, foram coletados os tempos de execução de cada iteração tal como obtidos dados estatísticos de cada parte do experimento.

Figura 11 – Distribuição temporal do tempo de execução de cada conjunto de iterações sobre os conjuntos de dados mnist e fashion-mnist.

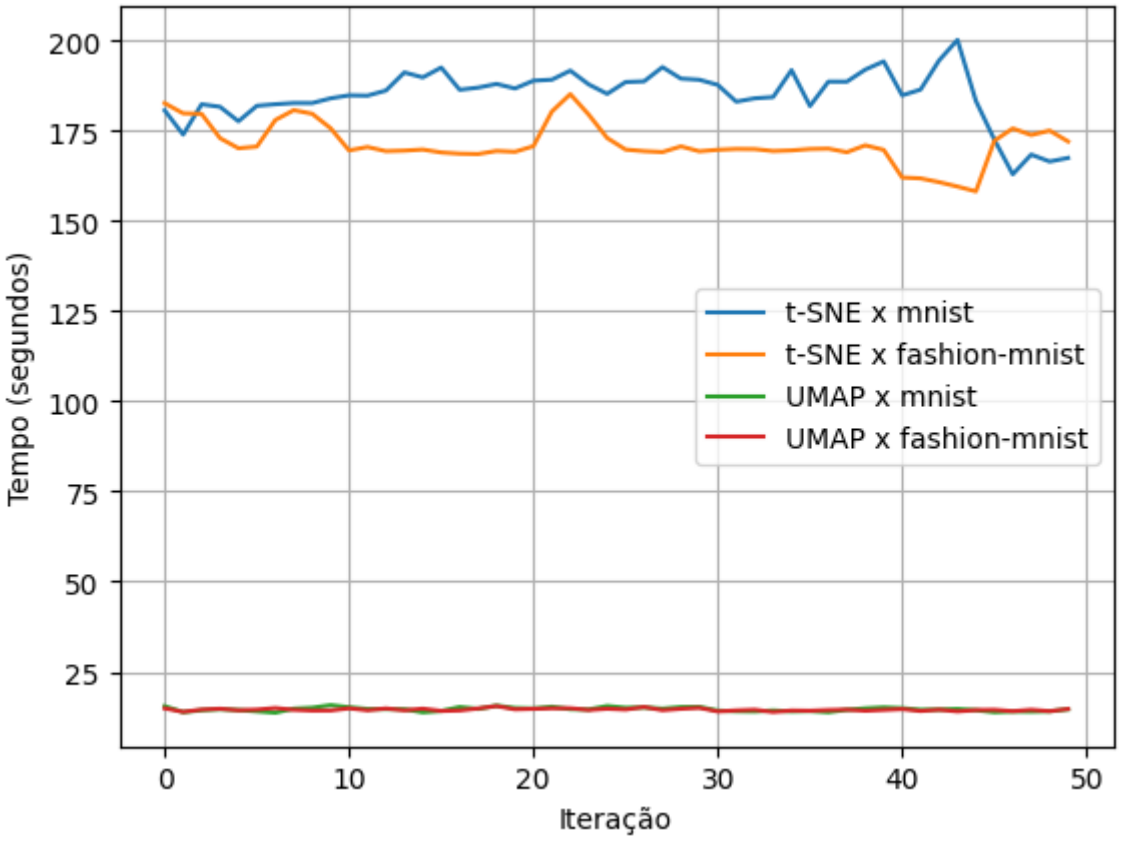


Tabela 3 – Resultados dos tempos de execução dos algoritmos sobre os conjuntos de dados com condições iniciais modificadas.

| Conjuntos de dados | Método | Tempo médio (s) | Desvio padrão (s) | Desvio padrão (%) | Variância (s) |
|---|---|---|---|---|---|
| mnist | t-SNE | 184,790 | 7,354 | 3.979 | 54,081 |
| fashion-mnist | t-SNE | 171,357 | 5,585 | 3,269 | 31.197 |
| mnist | UMAP | 14.618 | 3,612 | 0,528 | 0,278 |
| fashion-mnist | UMAP | 14,519 | 2,324 | 0,337 | 0,114 |

A partir dos resultados obtidos, nota-se que o algoritmo t-SNE possui uma variância alta no tempo de execução de acordo com o conjunto de dados inicial providenciado; embora em média o algoritmo tenda a possuir um tempo de execução relativamente estável quando submetido a um grande número de iterações, a variação de tempo de execução quando comparando duas execuções arbitrárias pode apresentar uma alta disparidade. O algoritmo UMAP, em contrapartida, possui uma sensibilidade menor quanto às condições iniciais do conjunto, com uma variação de tempo de execução tão pequena que esta pode ser atribuída ao fato do algoritmo utilizar métodos de aprendizado de máquina para o processamento dos conjuntos de dados que, devido à intrínseca aleatoriedade da técnica, pode gerar variações no tempo de execução.

# 5 Conclusões

Este trabalho explorou os conceitos, metodologia, teoria e aplicação de redução de dimensionalidade para fins de extração de informações de conjuntos em que, utilizando métodos convencionais, seriam de difícil manipulação.

Os resultados obtidos por meio dos experimentos realizados nos levam a conclusões que condizem com as métricas esperadas pela literatura, e efetivamente mostram o UMAP como uma poderosa ferramenta de redução de dimensionalidade, capaz de suplementar os campos de atuação anteriormente não cobertos por algoritmos predecessores, como a inabilidade de transpor conjuntos não-lineares do PCA tal como a baixa escalabilidade e a perda de distâncias globais do t-SNE.

Embora o algoritmo UMAP possa ser colocado como o Estado da Arte no campo de redução de dimensionalidade, fica claro que ainda existem casos de uso onde outros algoritmos são tão viáveis quanto ou mesmo se sobressaem sobre o UMAP; o algoritmo PCA possui vantagens únicas sobre os outros métodos como a linearidade de seu tempo de execução e a manutenção de distâncias globais entre os elementos, enquanto que o UMAP possui como pontos fortes uma robusta manipulação de espaços não-lineares e o agrupamento de dados em conjuntos com características comuns bem definidas. Desta forma, o algoritmo ideal para a manipulação de um conjunto irá depender em maior parte das características do conjunto em si. Embora o algoritmo t-SNE tenha sido de extrema importância para o avanço do campo de redução de dimensionalidade e aprendizado de máquina, é evidente que perante os métodos PCA e UMAP, seu nicho de atuação foi perdido, tornando-o um material referencial para futuras pesquisas e desenvolvimento de soluções na área.

# 6 Referências

OSKOLKOV, Nikolay. **How Exactly UMAP Works**: And why exactly it is better than tSNE. [*S. l.*], 3 out. 2019. Disponível em: https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668. Acesso em: 10 jan. 2023.

MCINNES, Leland; HEALY, John; MELVILLE, James. **UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction**. [*S. l.*], 9 fev. 2018.

DAMRICH, Sebastian; HAMPRECHT, Fred A.. **On UMAP's True Loss Function**. [*S. l.*], 26 mar. 2021. Disponível em: https://hci.iwr.uni-heidelberg.de/sites/default/files/publications/files/1675958088/damrich_21_on.pdf. Acesso em: 12 fev. 2023.

GHOJOGH, Benyamin; KARRAY, Fakhri; GHODSI, Ali; CROWLEY, Mark. **Uniform Manifold Approximation and Projection (UMAP) and its Variants: Tutorial and Survey**. ResearchGate, [*s. l.*], p. 1-11, 2021. Disponível em: https://www.researchgate.net/publication/354400863_Uniform_Manifold_Approximation_and_Projection_UMAP_and_its_Variants_Tutorial_and_Survey. Acesso em: 1 jan. 2023.

BEYER, Kevin; GOLDSTEIN, Jonathan, RAMAKRISHNAN, Raghu, SHAFT, Uri. **When Is "Nearest Neighbor" Meaningful?.** [*S. l.*], 1999. Disponível em: https://citeseerx.ist.psu.edu/doc/10.1.1.31.1422. Acesso em: 12 fev. 2023.

WANG, Yingfan; HUANG, Haiyang; RUDIN, Cynthia; SHAPOSHNIK, Yaron. **Understanding How Dimension Reduction Tools Work: An Empirical Approach to Deciphering t-SNE, UMAP, TriMAP, and PaCMAP for Data Visualization**. Journal of Machine Learning Research, [s. l.], v. 22, p. 1-73, 6 dez. 2020. Disponível em: https://www.jmlr.org/papers/volume22/20-1061/20-1061.pdf. Acesso em: 1 jan. 2023.

TANG, Jian; LIU, Jingzhou; ZHANG, Ming; MEI, Qiaozhu. **Visualizing Large-scale and High-dimensional Data**. Proceedings of the 25th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, p. 6-

10, 1 fev. 2016. Disponível em: https://arxiv.org/pdf/1602.00370.pdf. Acesso em: 5 jan. 2023.

WATTENBERG, Martin; VIÉGAS, Fernanda; JOHNSON, Ian. **How to Use t-SNE Effectively**: Although extremely useful for visualizing high-dimensional data, t-SNE plots can sometimes be mysterious or misleading. By exploring how it behaves in simple cases, we can learn to use it more effectively... [*S. l.*], 13 set. 2016. Disponível em: https://distill.pub/2016/misread-tsne/. Acesso em: 12 jan. 2023.