Cinthya Oestreich Silva

Mitigando o desbalanceamento de dados em problemas de classificação hierárquica com estrutura DAG por meio da combinação de uma abordagem local e resampling

Curitiba, Brasil

2025

Cinthya Oestreich Silva

Mitigando o desbalanceamento de dados em problemas de classificação hierárquica com estrutura DAG por meio da combinação de uma abordagem local e resampling

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Pontifícia Universidade Católica do Paraná como requisito para obtenção do título de mestre em Informática

Pontifícia Universidade Católica do Paraná- PUCPR

Programa de Pós-Graduação em Informática- PPGIa

Orientador: Prof. Dr. Carlos Nascimento Silla Junior

Curitiba, Brasil

2025

Dados da Catalogação na Publicação
Pontifícia Universidade Católica do Paraná
Sistema Integrado de Bibliotecas – SIBI/PUCPR
Biblioteca Central
Gisele Alves – CRB 9/1578

S586m
2025

Silva, Cinthya Oestreich
Mitigando o desbalanceamento de dados em problemas de classificação hierárquica com estrutura DAG por meio da combinação de uma abordagem local e resampling / Cinthya Oestreich Silva ; orientador : Carlos Nascimento Silla Junior. – 2025.
101 f. : il. ; 30 cm

Dissertação (mestrado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2025
Bibliografia: f. 80-83

1. Reamostragem (estatística). 2. Jackknife (Statistics). 3. Grafos acíclicos dirigidos. 4. Ontologia genética. I. Silla Junior, Carlos Nascimento. II. Pontifícia Universidade Católica do Paraná. Programa de Pós-Graduação em Informática. III. Título.

CDD. 20. ed. – 004

Pontifícia Universidade Católica do Paraná
Escola Politécnica
Programa de Pós-Graduação em Informática

Curitiba, 28 de fevereiro de 2025.

34-2025

DECLARAÇÃO

Declaro para os devidos fins, que **CINTHYA OESTREICH SILVA** defendeu a dissertação de Mestrado intitulada "**Mitigando o desbalanceamento de dados em problemas de classificação hierárquica com estrutura DAG por meio da combinação de uma abordagem local e resampling"**, na área de concentração Ciência da Computação no dia 02 de dezembro de 2024, no qual foi aprovada.

Declaro ainda, que foram feitas todas as alterações solicitadas pela Banca Examinadora, cumprindo todas as normas de formatação definidas pelo Programa.

Por ser verdade firmo a presente declaração.

gov.br Documento assinado digitalmente
EMERSON CABRERA PARAISO
Data: 06/03/2025 10:36:23-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Emerson Cabrera Paraiso
Coordenador do Programa de Pós-Graduação em Informática

Agradecimentos

Quero agradecer à minha família e amigos por me apoiarem nesta jornada. Minha mãe e meu pai, Cristina e Edward, sempre me incentivaram a buscar um grau mais elevado e a não desistir. Ao meu noivo, Gabriel, por me apoiar nos momentos bons e ruins, especialmente quando eu estava triste e me perguntava se deveria continuar com este trabalho. Ele esteve comigo quando passei pela qualificação e ficou muito feliz. No final, Gabriel me ajudou a lembrar que, não importa o que aconteça, ele sempre estará ao meu lado. Ao meu cachorro, Sushi, que esteve presente em todas as aulas online e ao meu lado enquanto escrevia este trabalho. Agradeço à minha família e amigos pela compreensão, especialmente quando precisei trabalhar na minha dissertação de mestrado e não tinha tempo para sair.

Quero estender meus sinceros agradecimentos ao meu orientador, Carlos Silla Jr., por acreditar em mim e pelo constante incentivo a buscar a excelência. Também quero reconhecer os professores do PPGIA pela paciência e apoio.

Por fim, agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro. A contribuição deles foi fundamental para tornar este trabalho uma realidade.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (ĆAPES) - Código de Financiamento 001.

If you failed, it means you tried. If you tried, it means you took a risk. Celebrate the fact that you put yourself out there and dared to go for it. That's damn brave, woman! Take time to honor that. Celebrate the fact that you got a result, even if it wasn't the result you'd hoped for, because it means you saw something through to its conclusion and can now pivot to your next move.

Reshma Saujani, Brave, Not Perfect: Fear Less, Fail More, and Live Bolder

Resumo

Problemas de classificação no mundo real frequentemente envolvem conjuntos de dados desbalanceados. Esse desafio se torna ainda mais complexo em cenários de classificação hierárquica, especialmente em taxonomias estruturadas em DAG (Directed Acyclic Graph), onde o desbalanceamento pode ocorrer em diferentes níveis da hierarquia. A abordagem de classificadores locais é a mais utilizada na literatura para a resolução desses problemas de classificação hierárquica, sendo observada uma melhora de performance quanto mais inclusiva se observa a política. Além disso, o método de *resampling* SMOTE (Synthetic Minority Oversampling Technique) é um dos métodos mais utilizados em problemas desbalanceados. Neste contexto, este trabalho investigou se a aplicação da política *less inclusive*, em conjunto com o SMOTE, poderia melhorar os resultados da métrica *F1-measur* hierárquica. As hipóteses formuladas previam que essa abordagem superaria o desempenho do uso isolado da política *less inclusive* em bases sintéticas e na GO (Gene Ontology). A metodologia adotada consistiu em traduzir a hierarquia em classificadores binários de acordo com a política *less inclusive* e treiná-los utilizando o classificador *Naive Bayes*, estabelecendo um *baseline*. Em seguida, o SMOTE foi aplicado a cada classificador binário e um novo treinamento foi realizado. Os experimentos foram conduzidos em bases sintéticas e reais (GO), e os resultados foram avaliados. Os resultados mostraram que, ao comparar o *baseline* com o método SMOTE nas bases sintéticas e GO, não houve diferença estatisticamente significativa, levando à rejeição das hipóteses iniciais. Da mesma forma, a comparação com o estado da arte em bases sintéticas não indicou diferença estatisticamente significativa da abordagem proposta sobre o método *flat*. Este estudo contribui para a investigação do desbalanceamento em problemas de classificação hierárquica em taxonomias do tipo DAG, abrindo caminho para futuras pesquisas sobre o impacto de técnicas de *resampling*, como o SMOTE, nesse contexto.

Keywords: Hierarchical Classification, Resampling, Directed Acyclic Graph, Local Classifier per node, Gene Ontology, Synthetic Datasets, SMOTE

Abstract

Real-world classification problems often involve imbalanced datasets. This challenge becomes even more complex in hierarchical classification scenarios, especially in taxonomies structured as Directed Acyclic Graphs (DAGs), where imbalance can occur at different levels of the hierarchy. The local classifier approach is the most widely used in the literature for addressing hierarchical classification problems, with improved performance observed as the policy becomes more inclusive. Additionally, the SMOTE (Synthetic Minority Oversampling Technique) resampling method is one of the most commonly used techniques for handling imbalanced problems. In this context, this study investigated whether applying the "less inclusive" policy, in combination with SMOTE, could improve the hierarchical F1-measure results. The formulated hypotheses predicted that this approach would outperform the isolated use of the less inclusive policy in synthetic datasets and GO (Gene Ontology). The adopted methodology involved translating the hierarchy into binary classifiers according to the "less inclusive" policy and training them using the Naïve Bayes classifier, establishing a baseline. Then, SMOTE was applied to each binary classifier, and a new training process was conducted. The experiments were performed on both synthetic and real datasets (GO), and the results were evaluated statistically. The results showed that, when comparing the baseline with the SMOTE method in synthetic and Gene Ontology datasets, no statistically significant difference was observed, leading to the rejection of the initial hypotheses. Similarly, a comparison with the state of the art in synthetic datasets did not indicate a statistically significant difference between the proposed approach and the flat method. This study contributes to the investigation of imbalance in hierarchical classification problems within DAG taxonomies, paving the way for future research on the impact of resampling techniques, such as SMOTE, in this context.

Keywords:Hierarchical Classification, Resampling, Directed Acyclic Graph, Local Classifier per node, Gene Ontology, Synthetic Datasets, SMOTE

Lista de Figuras

Lista de Tabelas

Lista de abreviaturas e siglas

ck-SVM	Composite Kernel Based SVM
DAG	Directed Acyclic Graph
D	DAG
FD	Full Depth Labeling
FT	Friedman Test
FMA	Free Music Archive
FunCat	Functional Catalog
GG	Global Classifier
GO	Gene Ontology
hF	Hierarchical F-mesasure
HMC	Hierarchical Multi-Label Classification
HMC-LMLP	Hierarchical Multi-label Classification using Local Multi-Layer Perceptron
hP	Hierarchical Precision
hR	Hierarchical Recall
KDD	Knowledge Discovery in Database
LCL	Local Classifier per Level
LCN	Local Classifier per Node
LCPN	Local Classifier per Parent Node
MDC	Multi-Dimensional Classification
MDHC	Multi-Dimensional Hierarchical Classification
MLNP	Mandatory Leaf-node Prediction
MPL	Multiple Path of Labels
NMLNP	Non-Mandatory Leaf-node Prediction
NA	Not Applicable

NN	Nearest Neighbor
NT	Nemenyi Test
PD	Partial Depth Labeling
SMOTE	Synthetic Minority Oversampling Technique
SPL	Single Path of Labels
SVM	Support Vector Machine
T	Tree
TPR	True Path Rule

Sumário

1 Introdução

Muitos estudos no campo da classificação em *machine learning* foram realizados focados na estrutura *flat* de acordo com Silla and Freitas (2011). Uma estrutura *flat* é representada por problema de classificação multi-class e multi-label. Por outro lado, muitos problemas de classificação do mundo real são naturalmente hierárquicos, por exemplo: categorização de texto (Stein; Jaques; Valiati, 2019), previsão de funções de proteína (Tang et al., 2019), classificação musical (Ariyaratne; Zhang, 2012) e classificação de imagens de Covid (Pereira et al., 2020). Na Figura 1 é possivel observar a diferença entre classificação *multi-class*, *multi-label* e hierárquica. Onde a classificação hierárquica é definida por uma hierarquia, ou seja, na Figura 1 o cachorro é definido por uma hierarquia e não apenas uma classe ou múltiplas classes.

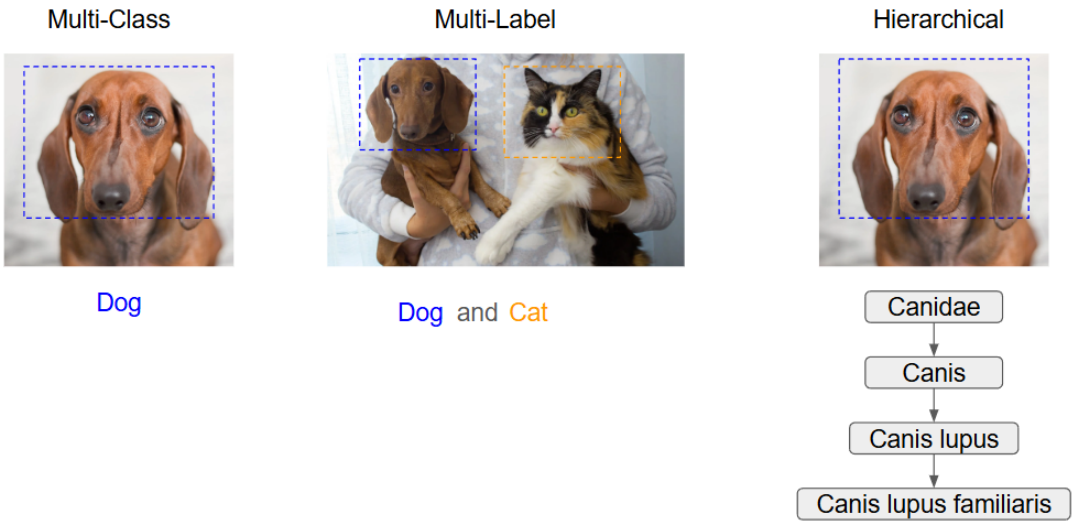
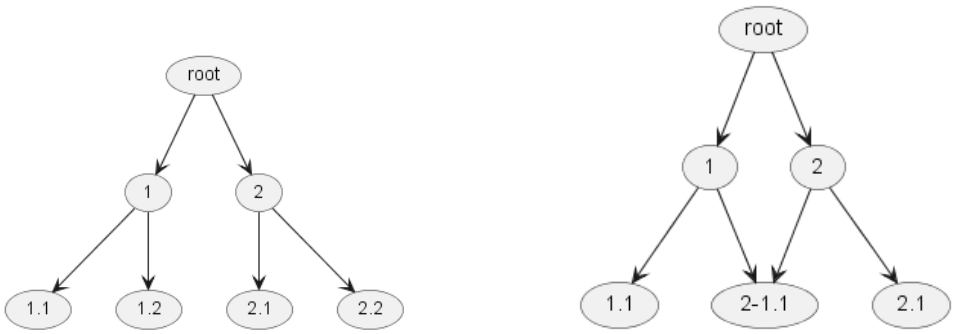


Figura 1 – Tipos de problemas de classificação

Existem dois tipos de estruturas hierárquicas: a árvore (*tree*) e o grafo acíclico direcionado (DAG). Elas se diferenciam em relação aos nós pais. Na taxonomia em árvore, um nó possui apenas um pai (Figura 2a); na taxonomia em DAG, um nó pode possuir um ou mais pais (Figura 2b). De acordo com Silla and Freitas (2011) problemas de classificação podem ser resolvidos utilizando alguns métodos como classificadores locais e classificadores globais. Sendo os classificadores locais a abordagem mais utilizada na literatura.

O desbalanceamento de classes em problemas reais acontece em todos os cenários, seja estrutura *flat* (Mrozek; Panneerselvam, 2020) ou hierárquica (Pereira; Costa; Silla Jr, 2021a). Os problemas de desbalanceados de dados são caracterizados por uma diferença de exemplos das classes nas bases de dados. Nos cenários de classificação, a classe majoritária é aquela que contém o maior número de exemplos em um conjunto de dados, enquanto a classe minoritária é



(a) *Tree* (b) Dag

Figura 2 – Taxonomias de classificação hierárquica.

a que contém o menor número de exemplos.

Para lidar com o desbalanceamento de dados dados em problemas de classificação tradicionais com estrutura multi-classe, muitos métodos de *resampling*(reamostragem) foram criados assim como demonstrado por Haixiang et al. (2017) e Spelmen and Porkodi (2018). Onde as abordagens mais comuns são o *oversampling* (sobreamostragem), *undersampling* (subamostragem) e *hybrid* (híbrido). O método de *undersampling* reduz o número de exemplos na classe majoritária. Os trabalhos de Batista, Prati and Monard (2004), Tomek (1976) e Hart (1968) apresentam diferentes métodos de *undersampling*. Os métodos de *oversampling* aumentam o número de exemplos da classe minoritária. Os trabalhos de Batista, Prati and Monard (2004) e Chawla et al. (2002) apresentam diferentes métodos de *oversampling*. Por fim, o método *hybrid* é uma combinação entre os métodos de *undersampling* e *oversampling*; o trabalho de Batista, Prati and Monard (2004) apresenta essa combinação.

Na classificação hierárquica o tema de desbalanceamento foi abordado por alguns autores. Os autores Chen and Hu (2012), Chen, Duan and Hu (2012), Pereira, Costa and Silla Jr (2018), Pereira et al. (2020), Pereira, Costa and Silla Jr (2021b), De Barros et al. (2021) e Stiller et al. (2024) realizaram estudos com taxonomia árvore e utilizaram métodos de *resampling*, porém não utilizaram nenhuma politica de classificadores locais por nó (LCN). Os autores Klungpornkun and Vateekul (2019) utilizaram taxonomia árvore e DAG e métodos de *resampling* porém não utilizaram nenhuma política de LCN. Os unicos autores a utilizarem política s de LCN é Pereira, Costa and Silla Jr (2021a) onde exploram problemas de classificação hierárquica em taxonomia árvore e utilizam métodos de *resampling*.

Pode-se perceber que o problema de desbalanceamento na classificação hierárquica utilizando taxonomia DAG e as política s de LCN ainda foram pouco explorados. Além disso, os autores que estudaram sobre o desbalanceamento em estruturas árvore sugerem em seu trabalhos futuros que problemas em estruturas DAG sejam estudados, uma vez que os métodos apresentados por estes não seriam eficazes nessa estrutura. Finalmente este trabalho realizou estudos do desbalanceamento em problemas de classificação hierárquica em taxonomia DAG utilizando política s de LCN.

1.1 Objetivos

O principal objetivo deste trabalho é avaliar se o uso de política de classificação local por nó (LCN) *less inclusive* combinado a um método de reamostragem SMOTE pode melhorar os resultados de *F1-measure* hierárquicos em problemas de classificação hierárquica estruturados em DAG.

Para alcançar o propósito principal deste trabalho, temos os seguintes objetivos específicos:

- Comparar os resultados das métricas hierárquicas entre experimentos utilizando apenas o classificadores local por nó (LCN) *less inclusive* e a combinação do *less inclusive* com o método de *resampling* SMOTE em bases sintéticas desbalanceadas.
- Comparar os resultados de f-measure dos estados da arte das bases sintéticas desbalanceadas e os experimentos que combinam a política *less inclusive* com o método de *resampling* SMOTE em bases sintéticas desbalanceadas.
- Comparar os resultados das métricas hierárquicas entre experimentos utilizando apenas o classificadores local por nó (LCN) *less inclusive* e a combinação do *less inclusive* com o método de *resampling* SMOTE em bases reais genéticas *Gene Ontology* (GO).

1.2 Declarações de Hipóteses

As hipóteses deste trabalho são:

- O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados da aplicação do classificador local por nó *less inclusive*.
- O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados do método *flat* do estado da arte.
- O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados nas bases GO, pode superar os resultados da aplicação do classificador local por nó *less inclusive*.

1.3 Organização do Texto

O restante deste documento está organizado em quatro capítulos: O Capítulo II apresenta o referencial teórico sobre os métodos de *resampling* e classificação hierárquica. O Capítulo

III apresenta os trabalhos relacionados. O método proposto é discutido no Capítulo IV. Os resultados deste trabalho são apresentados no Capítulo V. Por fim, o Capítulo VI conclui e sugere trabalhos futuros.

2 Referencial Teórico

2.1 Métodos de *Resampling* (Reamostragem)

Problemas de classificação no mundo real possuem bases de dados desbalanceadas. Bases de dados desbalanceadas são caracterizadas por classes que não possuem o número de instâncias iguais entre si. Existem exemplos onde essa diferença é expressiva como nos exemplos do sistema financeiro (Mrozek; Panneerselvam, 2020) ou da medicina (Pereira et al., 2020). Nesses exemplos as classes de interesse, que no caso de fraude são as instância que indicam fraude e no caso da medicina são as instâncias que indicam uma doença como o Covid, podem chegar a possuir um exemplo para milhões de exemplos das classes não de interesse.

A principal dificuldade de bases de dados desbalanceadas é o *overfitting*. Este problema se caracteriza pela alta acurácia do modelo devido a um alto aprendizado da classe majoritária, ou seja, a classe que possui mais instâncias. E muitas vezes em problemas desbalanceados a classe majoritária não é a classe de interesse.

Para resolver esse problema, muitos pesquisadores propuseram diversos métodos de *resampling*. Os métodos são divididos em *undersampling* (subamostragem), *oversampling* (superamostragem) e *hybrid* (híbridos). A diferença entre cada método está na forma como eles lidam com o desbalanceamento. Como o nome sugere, o *undersampling* reduz o número de amostras na classe majoritária, enquanto a *oversampling* aumenta o número de amostras na classe minoritária. Onde a classe minoritária é a classe que possui menos instâncias nas bases de dados. O método híbrido aplica ambos os métodos e geralmente apresenta o *oversampling* antes do *undersampling*.

2.1.1 *Undersampling*

Os métodos de *Undersampling* são usados para criar um novo conjunto de dados com um número reduzido de exemplos na classe majoritária ou, em alguns casos, para limpar os dados.

2.1.1.1 *Random Undersampling*

A *Random Undersampling* é um método para balancear os dados, escolhendo exemplos aleatórios da classe majoritária para serem eliminados. Esse método pode descartar dados essenciais para a classificação, uma vez que a eliminação ocorre de forma aleatória. No trabalho de Batista, Prati and Monard (2004), esse método é utilizado como referência básica (*baseline*).

2.1.1.2 Tomek-Link

Este método é um exemplo de *Undersampling* para balancear dados e realizar limpeza. Ele utiliza o conceito de Tomek-Link, proposto por Tomek (1976). O termo *link*(ligação) é usado porque conecta dois exemplos de classes diferentes. Para balancear os dados, descarta-se o exemplo da classe majoritária, e, para limpeza, ambos os exemplos são descartados.

Vamos definir o que é um Tomek-Link. Começamos com um par $d(E_i, E_j)$, em que esse par consiste em um exemplo da classe majoritária e outro da classe minoritária. Esse par pode ser chamado de Tomek-Link se não houver outro exemplo E_l tal que $d(E_i, E_l) < d(E_i, E_j)$ ou $d(E_j, E_l) < d(E_i, E_j)$.

A regra do Tomek-Link pode ser difícil de imaginar, então vamos apresentar um exemplo numérico na Figura 3. Imagine que temos E_i como o ponto (2,4), E_j como o ponto (3,4) e E_l como o ponto (5,5). Se utilizarmos a fórmula $d(E_i, E_l) < d(E_i, E_j)$ ou $d(E_j, E_l) < d(E_i, E_j)$, e $d(A, B) = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}$, podemos calcular $d(E_i, E_j) = 1$, $d(E_i, E_l) = 3,16$ e $d(E_j, E_l) = 2,23$. Neste exemplo, a distância entre o Tomek-Link é 1, e a distância entre E_l não é menor do que a distância entre E_i e E_j. Para confirmar se o par E_i e E_j é um Tomek-Link, todos os outros exemplos precisam ser testados. Neste caso, todas as distâncias são maiores do que a distância entre o par, portanto, temos aqui um Tomek-Link.

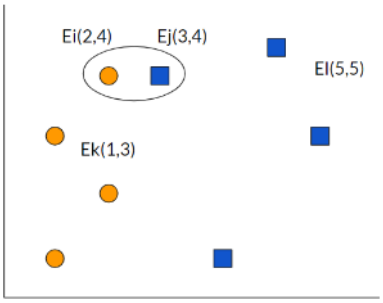


Figura 3 – Exemplo Numérico de Tomek-Link

2.1.1.3 *Condensed Near Neighbou*

Este método tem como objetivo encontrar um subconjunto menor da classe majoritária que seja mais representativo do que o conjunto de dados completo proposto, conforme sugerido por Hart (1968). O método *Condensed Nearest Neighbor* utiliza a regra do Vizinho Mais Próximo (*Nearest Neighbor* - NN) para realizar o *undersampling*. O método inicia com dois locais de armazenamento chamados *store* e *grabbag*. Em seguida uma amostra é adicionada ao local *store*, outra amostra é escolhida e classificada pela regra NN utilizando a amostra no local *store* como referência. Se a amostra foi classificada corretamente, ela é enviada para o local *grabbag*, caso contrário, ela será enviada para o local *store*. Esses passos se repetem até que todos os exemplos sejam transferidos para um dos locais. Depois o processo é refeito utilizando os exemplos que estão dentro do *grabbag*. O processo termina quando o *grabbag* está vazio, significando que o

menor subconjunto é todo o *dataset*, ou quando nenhum exemplo é enviado para o *store*. Assim os exemplos que estão no local *store* são o novo subconjunto de dados e os exemplos do local *store* são descartados.

2.1.2 *Oversampling*

Os métodos de *oversampling* são usados para criar subconjuntos adicionando exemplos à classe minoritária.

2.1.2.1 *Random Oversamplinng*

Random oversampling é um método que seleciona exemplos aleatórios da classe minoritária para serem duplicados e adicionados à base de dados. No trabalho de Batista, Prati and Monard (2004), esse método é utilizado como referência básica (*baseline*). Esse método pode introduzir *overfitting* ao problema de classificação, pois duplica aleatoriamente exemplos, podendo duplicar os mesmos exemplos várias vezes.

2.1.2.2 SMOTE

O método SMOTE, proposto por Chawla et al. (2002), baseia-se na criação de exemplos sintéticos da classe minoritária por meio da interpolação das instâncias mais próximas. A interpolação ocorre quando calculamos a diferença entre o vetor de características da amostra analisada e seu vizinho mais próximo. Multiplicamos essa diferença por um valor aleatório entre 0 e 1 e, em seguida, adicionamos esse resultado ao vetor de características em análise, resultando na seleção de novas amostras localizadas entre a reta que conecta a amostra inicial e seu vizinho.

2.1.3 Híbrido

Os métodos híbridos são boas opções para reduzir o *overfitting*, pois geralmente aplicam um método de *oversampling*, em seguida, eliminam alguns exemplos por meio de um método de*undersampling*. Eliminando assim exemplos que podem ser repetidos ou estarem em locais que dificultem o treinamento do modelo.

2.1.3.1 SMOTE + Tomek-Links

Este método foi proposto por Batista, Prati and Monard (2004) com o objetivo de criar *clusters* bem definidos. O conjunto de dados original é superamostrado (*oversampling*) utilizando o método SMOTE e, em seguida, os Tomek-links são identificados e removidos, produzindo um conjunto de dados balanceados. Este método foi desenvolvido para evitar a invasão da classe minoritária na classe majoritária após a aplicação de métodos de *oversampling*. O processo de

interpolação no método SMOTE pode adicionar exemplos dentro da classe majoritária, causando essa invasão. Essa proximidade entre a classe majoritária e a classe minoritária gera um *cluster* não bem definido, o que pode prejudicar o problema de classificação.

2.2 Classificação Hierárquica

A classificação hierárquica é um problema de classificação onde as classes são representadas por uma hierarquia de classes. Essa hierarquia possui uma taxonomia de classes. Essa taxonomia de classes possui algumas propriedades, definidas por Wu, Zhang and Honavar (2005), e apresenta uma relação "IS-A" em uma estrutura de árvore. A taxonomia de classes é definida como um conceito com (C, $\prec$), onde C é um conjunto finito de classes no domínio da aplicação, e $\prec$ é a relação "IS-A". No estudo de Wu, Zhang and Honavar (2005), a relação é definida como anti reflexiva e transitiva. No entanto, no estudo de Silla and Freitas (2011), a relação "IS-A" é definida como assimétrica, anti reflexiva e transitiva. Para que um problema seja classificado como um problema de classificação hierárquica, a estrutura de classes deve seguir quatro regras:

- O único elemento soberano é a raiz da árvore ('R').
- $\forall c_i, c_j \in C$, se $c_i \prec c_j$ então $c_j \not\prec c_i$.
- $\forall c_i \in C, c_i \not\prec c_i$.
- $\forall c_i, c_j, c_k \in C, c_i \prec c_j$ e $c_j \prec c_k$ então c_j implicando em $c_i \prec c_k$.

A regra na taxonomia de classes estruturada em árvore também pode ser aplicada à taxonomia de classes estruturada em um Grafo Acíclico Direcionado (DAG). A diferença entre uma árvore e um DAG está no número de pais que cada nó pode ter. Na estrutura de taxonomia em árvore, um nó pode ter apenas um único pai, como mostrado na Figura 4. Já na estrutura de taxonomia de classes em DAG, um nó pode ter múltiplos pais, como ilustrado na Figura 5.

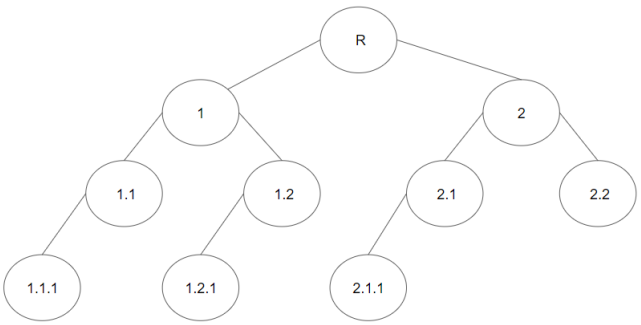


Figura 4 – Exemplo de classificação hierárquica com taxonomia árvore.

Existem diferentes maneiras de lidar com problemas de classificação hierárquica, utilizando a abordagem de classificação plana (*flat*), a abordagem de Classificador Local (*Local*

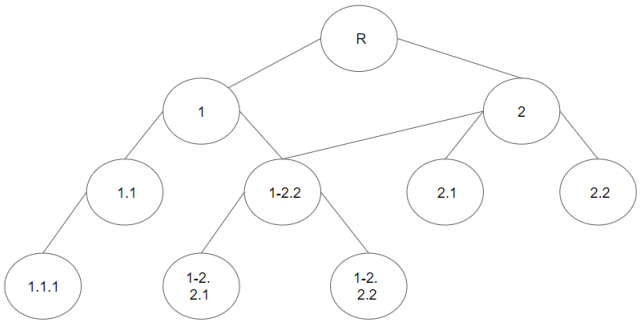


Figura 5 – Exemplo de classificação hierárquica com taxonomia DAG.

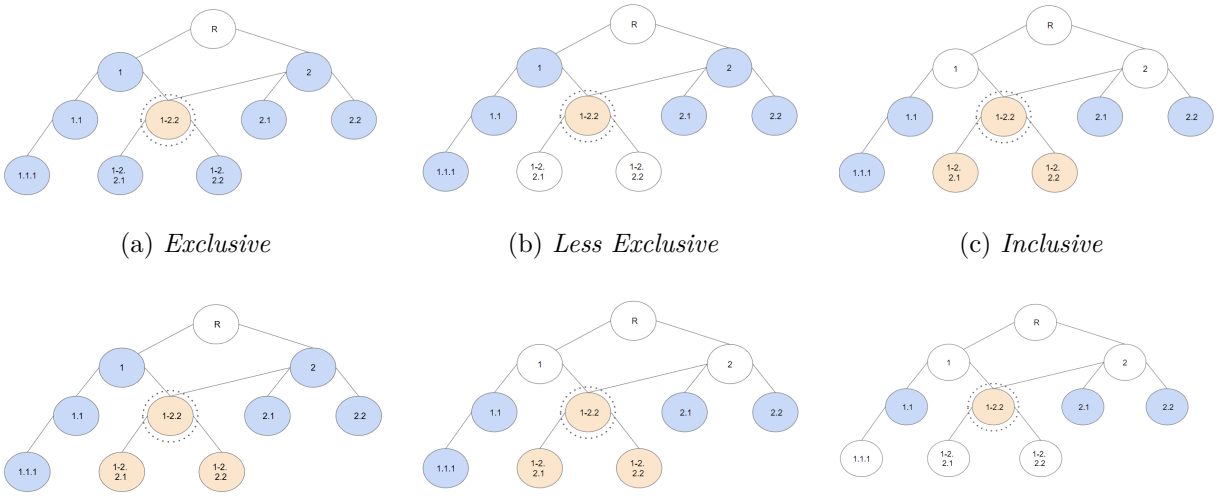
Classifier) e a abordagem de Classificador Global (*Global Classifier*), conforme descrito por Silla and Freitas (2011). A abordagem *flat* funciona como um algoritmo de classificação tradicional, onde a predição ocorre apenas nos nós folha. A desvantagem dessa abordagem é que ela ignora a hierarquia de classes do problema.

Na abordagem de *Local Classifier*, existem três métodos que podem ser aplicados: classificador local por nó (*Local Classifier per Node*), classificador local por nó pai (*Local Classifier per Parent Node*) e um classificador local por nível (*Local Classifier per Level*). O *Local Classifier per Node* (LCN) treina um classificador binário para cada nó. Embora pareça simples, essa abordagem apresenta diferentes política s para determinar os exemplos positivos e negativos no treinamento, os quais chamamos de políticas. Existem seis políticas a serem consideradas: exclusiva (*exclusive*), menos exclusiva (*less exclusive*), menos inclusiva (*less inclusive*), inclusiva (*inclusive*), irmãos (*siblings*) e irmãos exclusivos (*exclusive siblings*).

- *Exclusive*: Apenas a classe verdadeira é considerada como exemplo positivo, enquanto os outros exemplos são considerados negativos. Um exemplo desta política eh apresentado na Figura 6a.
- *Less exclusive*: A classe verdadeira é considerada como exemplo positivo, e os outros exemplos, exceto os descendentes, são considerados negativos (os descendentes são ignorados). Um exemplo desta política é apresentado na Figura 6b.
- *Inclusive*: Considera como exemplos positivos a classe verdadeira e seus descendentes, enquanto os outros exemplos são considerados negativos, exceto seus ancestrais, que não são considerados. Um exemplo desta política é apresentado na Figura 6c
- *Less inclusive*: Considera como exemplos positivos a classe verdadeira e seus descendentes, enquanto os outros exemplos são considerados negativos. Um exemplo desta política é apresentado na Figura 6d.
- *Siblings*: Considera como exemplos positivos a classe verdadeira e seus descendentes, enquanto seus irmãos são considerados exemplos negativos (os outros exemplos não são considerados). Um exemplo desta política é apresentado na Figura 6e.

- *Exclusive siblings*: Apenas a classe verdadeira é considerada como exemplo positivo, e seus irmãos são considerados exemplos negativos (os outros exemplos não são considerados). Um exemplo desta política é apresentado na Figura 6f

.



(a) *Exclusive* (b) *Less Exclusive* (c) *Inclusive*

(d) *Less Inclusive* (e) *Siblings* (f) *Exclusive Siblings*

Figura 6 – Esta imagem apresenta as seis políticas do classificador local por nó. Um círculo pontilhado circunda a classe verdadeira. Os exemplos positivos são laranja, os exemplos negativos são azuis e os brancos não são considerados no classificador.

A abordagem de Classificador Local por Nó Pai utiliza um classificador multi-classe em cada nó pai para distinguir os filhos. Já a abordagem de Classificador Local por Nível constrói um classificador multi-classe para cada nível da hierarquia.

O último classificador é o global, onde todas as classes são treinadas em um único classificador.

O trabalho de Silla and Freitas (2011) propôs uma estrutura unificada para classificação hierárquica. A primeira estrutura é uma categorização dos problemas de classificação hierárquica com uma 3-tupla γ, ψ, ϕ:

- γ: Indica como a estrutura do grafo é especificada e suas inter-relações. Os valores podem ser:
  - Árvore (*Tree*): Indica que a estrutura da classificação hierárquica é uma árvore.
  - DAG: Indica que a estrutura da classificação hierárquica é um Grafo Acíclico Direcionado.
- ψ: Indica uma característica fundamental: os caminhos dos rótulos das classes. Os rótulos em um problema de classificação hierárquica podem ter um único caminho de rótulos (SPL) ou múltiplos caminhos de rótulos (MPL). Como exemplo de único caminho de rótulos, voltando à Figura 5, a classe(rótulo) 1.1.1 é a classe esperada, onde há apenas um

caminho a se seguir. Para MPL, podemos dizer que, na Figura 5, os exemplos possuem duas classes, 1.1.1 e 1.2.2.1. Essa situação equivale ao conceito de multi-rótulo hierárquico. Para este atributo, temos duas opções de valores:

  - SPL (Caminho Único de Rótulos).
  - MPL (Múltiplos Caminhos de Rótulos).

- ϕ: Outra característica essencial é a profundidade dos rótulos, que pode ser de rótulos com profundidade total (FD- *Full Depth*) ou com profundidade parcial (PD - *Partial Depth*).

  - FD: É caracterizada por uma hierárquica onde os rótulos não possuem descendentes, os rótulos são as folhas desta hierarquia. Como na Figura 5, a classe 1.1.1 não possui descendentes e é a folha deste caminho. Assim, todas as instâncias devem ter rótulos que representam o final da hierarquia.
  - PD: No rótulo com profundidade parcial (*Partial Depth Labeling*), como o nome sugere, nem todas as classes são rotuladas até as folhas. Imagine que, no exemplo da Figura 5, o rótulo 1.1.1 fosse substituído por 1.1. O rótulo então possui um descendente e não é o nó folha do caminho daquel rótulo.

Uma vez categorizados os problemas de classificação hierárquica, outro aspecto importante é a categorização dos algoritmos. Aqui, temos uma 4-tupla Δ, Ξ, Ω, Θ, onde:

- Δ: Indica se os algoritmos podem prever um único caminho de rótulos ou múltiplos caminhos de rótulos. Os dois valores possíveis são:

  - SPL (*Single Path of Labels*): O algoritmo pode prever um único caminho de rótulos.
  - MPL (*Multiple Path of Labels*): O algoritmo pode prever múltiplos caminhos de rótulos.

- Ξ: Indica a profundidade da predição e pode assumir dois valores:

  - MLNP (*Mandatory Leaf-Node Prediction*): O algoritmo sempre atribuirá um rótulo a um nó folha.
  - NMLNP (*Non-Mandatory Leaf-Node Prediction*): O algoritmo pode atribuir rótulos em qualquer nível.

- Ω: O tipo de taxonomia que o algoritmo pode lidar, com dois valores possíveis:

  - T (*Tree*): A taxonomia esta em formato de árvore.
  - D (DAG): A taxonomia está em formato de DAG.

- Θ: Indica como o algoritmo será categorizado dentro da taxonomia proposta. Essas taxonomias foram apresentadas anteriormente, e os valores possíveis são:

- LCN (*Local Classifier per Node*).
- LCL (*Local Classifier per Level*).
- LCPN (*Local Classifier per Parent Node*).
- GC (*Global Classifier*).

É fundamental utilizar o método apropriado para avaliar a classificação hierárquica. A precisão hierárquica (hP), o *recall* hierárquico (hR) e o *f-measure* hierárquico (hF), propostas por Kiritchenko, Canada and Famili (2014), são amplamente utilizados. Nas equações (2.1) e (2.2), $\widehat{P_i}$ representa todas as classes previstas para o exemplo de teste i e todas suas classes ancestrais correspondentes. Já $\widehat{T_i}$ representa todas as classes específicas do exemplo de teste i e todas as suas classes ancestrais.

$$hP = \frac{\sum_i |\widehat{P_i} \bigcap \widehat{T_i}|}{\sum_i |\widehat{P_i}|} \tag{2.1}$$

$$hR = \frac{\sum_i |\widehat{P_i} \bigcap \widehat{T_i}|}{\sum_i |\widehat{T_i}|} \tag{2.2}$$

$$hF = \frac{2 * hP * hR}{hP + hR} \tag{2.3}$$

3 Trabalhos Relacionados

Neste capítulo, apresentamos os trabalhos relacionados e os procedimentos utilizados para encontrar os trabalhos relevantes.

3.1 Procedimento de Revisão de Literatura

O procedimento de revisão de literatura foi dividido em três etapas. A primeira etapa foi determinar os objetivos dos trabalhos relacionados. A segunda etapa consistiu em definir as palavras-chave para formular as consultas nos sites de busca. A terceira etapa foi estabelecer os critérios de inclusão e exclusão.

3.1.1 Objetivos dos Trabalhos Relacionados

Os objetivos dos trabalhos relacionados são:

- Identificar trabalhos relevantes dentro da classificação hierárquica multirrótulo com taxonomia em DAG utilizando a abordagem de classificador local por nó e aplicando técnicas de *resampling*.
- Identificar trabalhos relevantes na classificação hierárquica multirrótulo utilizando a abordagem de classificador local por nó e aplicando técnicas de *resampling*.
- Identificar trabalhos relevantes na classificação hierárquica multirrótulo com taxonomia em DAG e aplicando técnicas de *resampling*.
- Identificar trabalhos relevantes na classificação hierárquica que aplicaram técnicas de *resampling*.

3.1.2 Palavras-chave e Consultas

As palavras-chave podem ser encontradas na Tabela 1. Como a revisão da literatura foi realizada em artigos e jornais no idioma Inglês, as palavras chave também foram definidas neste idioma.

A revisão da literatura foi conduzida no ambiente do Google Acadêmico, abrangendo o período de 2012 a 2024. A revisão foi aplicada em duas etapas. Na primeira etapa, as consultas foram aplicadas a todos os artigos disponíveis no ambiente do Google Acadêmico. Uma busca mais restrita foi realizada na segunda etapa, utilizando a técnica de *forward snowball* (Wohlin, 2014) em três artigos. Os três artigos escolhidos foram dois relacionados às bases de dados

Palavras-chave
Hierarchical
Hierarchical Classification
Multi-label
resampling
Directed Acyclic Graph

Tabela 1 – Palavras-chave

utilizadas neste trabalho (Vens et al., 2008; Serrano-Pérez; Sucar, 2021) e um relacionado à técnica de *resampling* utilizada neste trabalho (Chawla et al., 2002).

As consultas foram definidas utilizando as palavras-chave e realizadas no título ou no corpo do artigo. O prefixo "intitle:" indicava que a palavra foi consultada no título, enquanto, sem o prefixo, a palavra seria consultada no corpo do artigo.

3.1.2.1 Consultas Aplicadas em Todos os Artigos

1. Todos Artigos
   a) (intitle:"hierarchical"OR intitle:"Hierarchical classification").
   b) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and (intitle:"multi-label").
   c) intitle:"hierarchical"OR intitle:"Hierarchical classification") and (intitle:"multi-label") and "resampling".
   d) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling".
   e) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling"and "multi-label".
   f) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling"and "multi-label"and "directed acyclic graph".
   g) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling"and "hierarchical classification".
   h) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling"and "hierarchical classification"and "directed acyclic graph".

3.1.2.2 *Forward Snowball*

1. Vens et al. (2008)
   a) (intitle:"hierarchical"OR intitle:"Hierarchical classification").
   b) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and (intitle:"multi-label").
   c) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and (intitle:"multi-label") and "resampling".

d) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling".

2. Serrano-Pérez and Sucar (2021)

   a) (intitle:"hierarchical"OR intitle:"Hierarchical classification").

3. Chawla et al. (2002)

   a) (intitle:"hierarchical"OR intitle:"Hierarchical classification").

   b) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and (intitle:"multi-label").

   c) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and (intitle:"multi-label") and "resampling".

   d) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling".

   e) (intitle:"hierarchical"OR intitle:"Hierarchical classification") and "resampling"and "multi-label".

3.1.3 Critérios de Inclusão e Exclusão

Os critérios de inclusão e exclusão utilizados foram.

Critérios de Exclusão

- Excluir artigos que não estão em inglês.
- Excluir artigos com títulos não relacionados à classificação hierárquica.
- Excluir artigos com resumos não relacionados à classificação hierárquica e *resampling*.
- Excluir livros, artigos de *workshops*, dissertações e teses.
- Excluir artigos duplicados.

Critérios de Inclusão

- Incluir artigos com problemas de classificação hierárquica utilizando taxonomia DAG e aplicação de métodos de *resampling*.
- Incluir artigos com problemas de classificação hierárquica e aplicação de métodos de *resampling*.
- Incluir artigos com problemas de classificação hierárquica utilizando o política s de Classificador Local por Nó e aplicação de métodos de *resampling*.

3.2 Resultado da Revisão de Literatura

Os resultados da revisão de literatura para os trabalhos relacionados são representados em diagramas de Venn. O primeiro diagrama, apresentado na Figura 7, representa a pesquisa realizada em todos os artigos utilizando as consultas no ambiente do Google Acadêmico, conforme descrito na Seção 3.1.2.1. A primeira consulta (intitle:"hierarchical"OR intitle:"Hierarchical classification") resultou em 250.000 artigos, dos quais apenas 73 permaneceram após a consulta das *queries*.

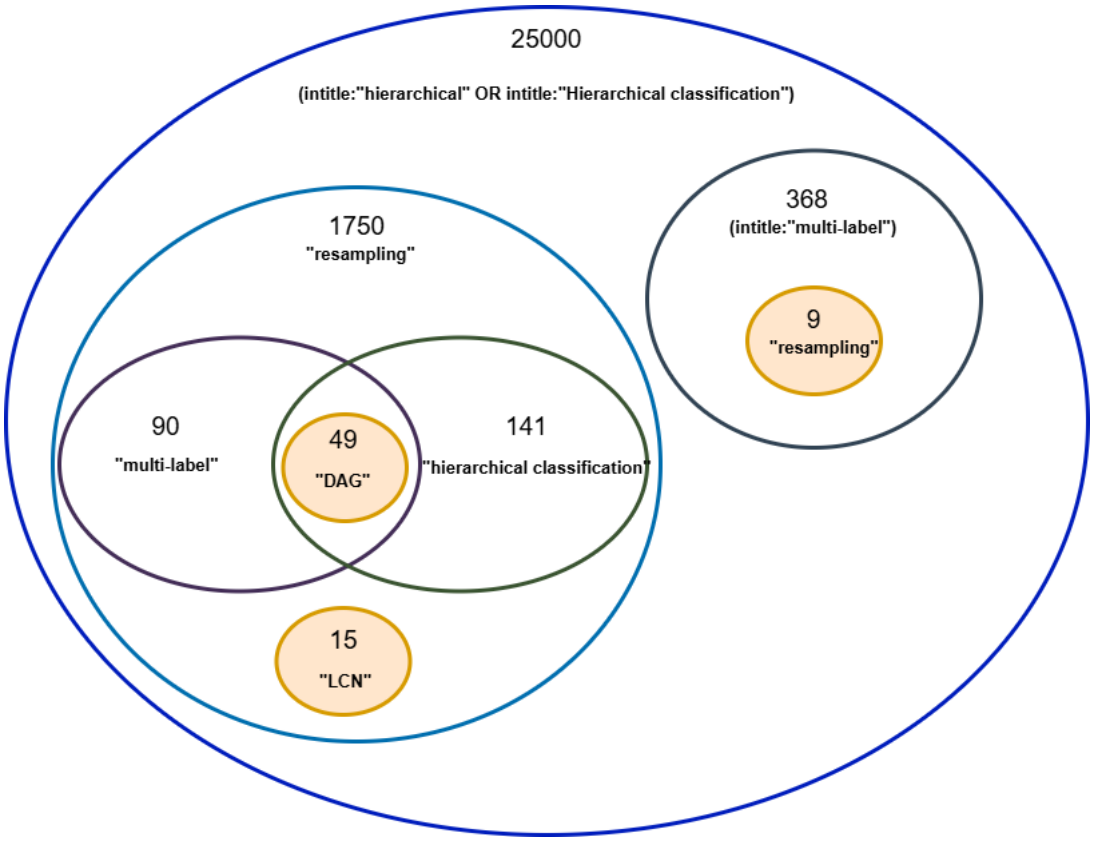


Figura 7 – Diagrama de Venn de consultas aplicadas a todos os artigos do *framework* Google Acadêmico apresentado na seção 3.1.2.1.

O segundo diagrama é apresentado na Figura 8 e representa as citações de Vens et al. (2008) e suas respectivas consultas, conforme descrito na Seção 3.1.2.2. Todas as citações de Vens et al. (2008) totalizam 859 artigos, dos quais apenas 15 permaneceram utilizando as consultas das *queries*.

O terceiro diagrama, apresentado na Figura 9, representa as citações de Serrano-Pérez and Sucar (2021) e suas respectivas consultas, conforme descrito na Seção 3.1.2.2. Todas as citações totalizam cinco artigos, dos quais apenas quatro permaneceram após as consultas das das *queries*.

O quarto diagrama, apresentado na Figura 10, representa as citações de Chawla et al. (2002) e suas respectivas consultas, conforme descrito na Seção 3.1.2.2. Todas as citações totalizam 32.227 artigos, dos quais apenas 36 permaneceram após a consultas das das *queries*.

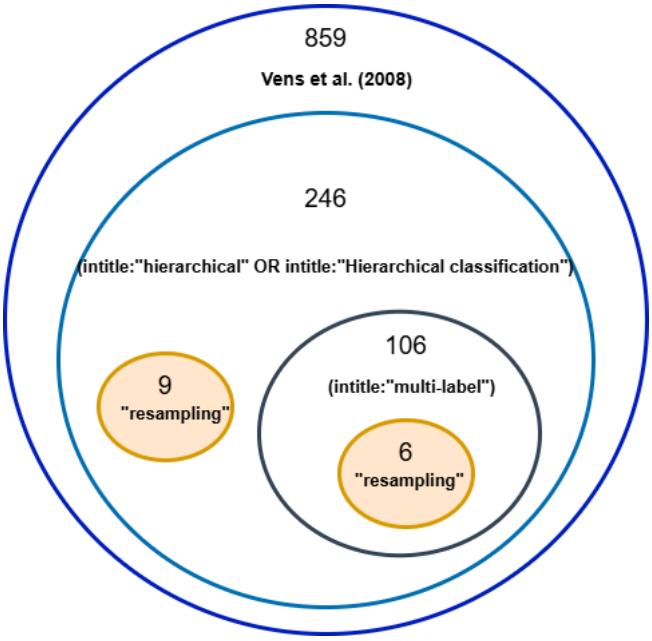


Figura 8 – Diagrama de Venn de consultas aplicadas nas citações de Vens et al. (2008) apresentado na seção 3.1.2.2.

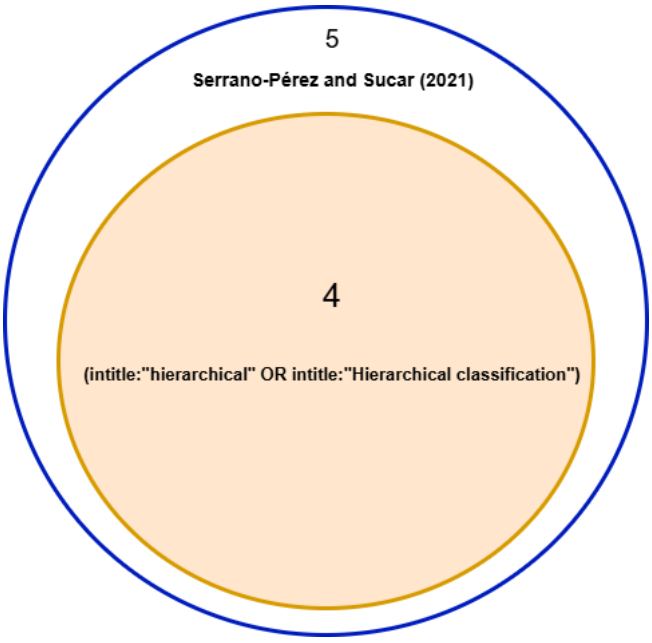


Figura 9 – Diagrama de Venn de consultas aplicadas nas citações de Serrano-Pérez and Sucar (2021) apresentado na seção 3.1.2.2.

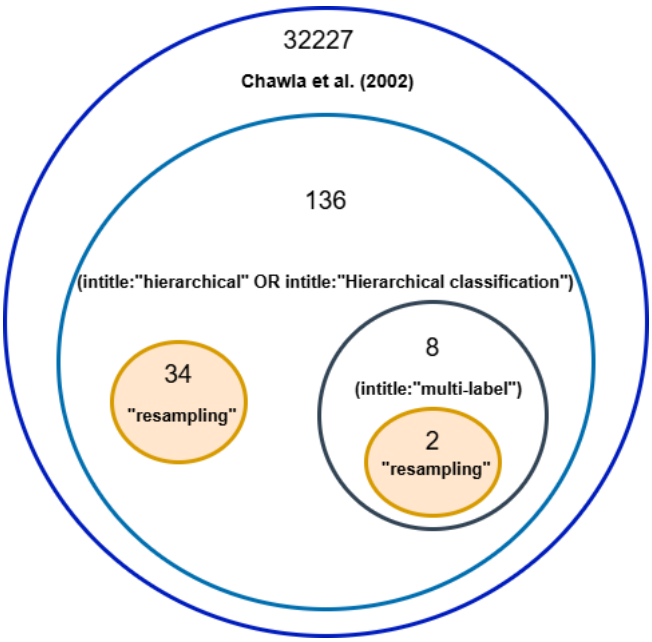


Figura 10 – Diagrama de Venn de consultas aplicadas nas citações de Chawla et al. (2002) apresentado na seção 3.1.2.2.

Esta pesquisa resultou em 128 artigos. Após essa etapa, os critérios de exclusão e inclusão foram aplicados, resultando em apenas dez artigos. A Tabela 2 apresenta os artigos encontrados e suas principais características para comparação: o domínio da base de dados (Domínio), a taxonomia do problema hierárquico (Taxonomia), se o trabalho utiliza o Classificador Local por Nó (LCN), tipo de rótulo utilizado (Tipo) e o tipo de método de *resampling* aplicado (*Resampling*).

Artigo	**Domínio**	**Taxonomia**	**LCN**	**Tipo**	**"Resampling"**
Vens et al. (2008)	Biologia	Árvore e DAG	Não	Multi-Label	NA (Não Aplicável)
Chen and Hu (2012)	Biologia	Árvore	Não	Multi-Label	Oversampling
Chen, Duan and Hu (2012)	Biologia	Árvore	Não	Multi-Label	Oversampling
Pereira, Costa and Silla Jr (2018)	Musica	Árvore	Não	Multi-Class	Oversampling e Undersampling
Klungpornkun and Vateekul (2019)	Texto	Árvore e DAG	Não	Multi-Class	Oversampling e Hibrido
Pereira et al. (2020)	Imagem	Árvore	Não	Multi-Class	Oversampling, Undersampling e Hybrid

Pereira, Costa and Silla Jr (2021b)	Texto, Musica, Biologia, Áudio, Vidro e Imagem	Árvore	Não	Binaria, Multi-Class and Multi-label	Oversampling e Undersampling
Pereira, Costa and Silla Jr (2021a)	Texto, Musica, Biologia e Imagem	Árvore	Sim	Multi-Class e Multi-label	Oversampling, Undersampling e Hibrido
De Barros et al. (2021)	Imagem	Árvore	Não	Multi-Class	Oversampling e Hibrido
Serrano-Pérez and Sucar (2021)	Artificial	Árvore e Dag	Não	Multi-Class e Multi-label	NA
Montenegro, Santana and Lozano (2023)	Artificial	Árvore (Multidimensional)	Não	Multi-Class	NA
Stiller et al. (2024)	Imagem	Árvore	Não	Multi-Class	Ovesampling

Tabela 2 – Trabalhos Relacionados e suas Características. Domínio: Domínio das bases de dados utilizadas. Taxonomia: Taxonomia da classificação hierárquica utilizada. LCN: Indica se o trabalho utilizou a abordagem de Classificador Local por Nó (*Local Classifier per Node*). Tipo: Tipo de rótulos utilizados no trabalho. *Resampling*: Tipo de técnica de *resampling* aplicada no trabalho.

3.3 Discussão dos Artigos Relacionados

Na Tabela 2, é possível observar diversos domínios das bases de dados utilizadas nos trabalhos relacionados, como biologia, imagem, música, texto, áudio, vidro e bases de dados artificiais. As bases são problemas de classificação hierárquica que possuem taxonomias em árvore ou DAG. É possível observar nestes casos que problemas reais são naturalmente desbalanceados.

Quando observamos bases reais de classificação hierárquica com taxonomia em DAG temos o trabalho de Vens et al. (2008) que utiliza as bases de dados genéticas *Functional Catalog* (Funcat) e *Gene Ontology* (GO); essas bases são complexas e apresentam desbalanceamento. Onde a base Funcat está organizada em taxonomia de árvore e a GO está organizada em taxonomia de DAG.

O trabalho de Vens et al. (2008) compara e propõe um novo método para resolver o

problema de classificação hierárquica multi-rótulo (HMC). O estudo comparou os métodos CLUS-HMC, CLUS-SC e CLUS-HSC.

CLUS-HMC utiliza uma única árvore para prever a probabilidade de cada classe na hierarquia em cada nó folha. CLUS-SC treina uma árvore separada para cada classe da hierarquia, cada árvore é uma árvore de classificação binária de rótulo único. CLUS-HSC treina uma árvore separada para cada aresta da hierarquia.

Para a taxonomia em DAG, os métodos CLUS-HMC e CLUS-HSC precisaram ser modificados para se adaptarem aos múltiplos caminhos das classes. O CLUS-SC, por ignorar a hierarquia das classes, não necessitou de alterações.

Ao analisar como cada método lida com o desbalanceamento, observa-se que o CLUS-HMC, por aprender todas as classes simultaneamente, não avalia os efeitos das distribuições desbalanceadas das classes. O CLUS-SC, ao construir várias árvores, precisa lidar diretamente com distribuições desbalanceadas das classes. O CLUS-HSC, segundo os autores, parece ter uma distribuição de classes mais balanceada, mas possui uma pequena probabilidade de se tornar desbalanceada. Embora cada método apresente diferenças na forma como lida com o desbalanceamento, nenhuma técnica de reamostragem foi aplicada para tratar o problema.

Podemos perceber que os problemas de classificação hierárquicos demandam soluções complexas. As bases de dados GO e Funcat são reais, e segundo Vens et al. (2008), com profundidade média de 14 e 6 níveis, e quantidade total de classes de 22.960 e 1.362, respectivamente. A solução de problemas com este tipo de base demanda tempo e capacidade computacional, e muitas vezes dificulta o entendimento da solução. Por este motivo, pesquisas recentes, como a de Serrano-Pérez and Sucar (2021), têm focado na criação de conjuntos de bases de dados sintéticos que são mais simples do que os conjuntos de dados reais, com o objetivo de estudar taxonomias em árvores e em DAG. Esses conjuntos de dados sintéticos oferecem uma oportunidade de compreender os desafios e propor soluções para problemas hierárquicos em taxonomias mais complexas, sem a necessidade de lidar inicialmente com toda a complexidade dos dados reais.

Muitos trabalhos que abordam o desbalanceamento em problemas de classificação hierárquicas trabalham com bases de taxonomia de árvore. Uma vez que a taxonomia em árvore não possui múltiplos caminhos para uma determinada classe, e possui muitas vezes menor complexidade como visto na diferença de profundidade e classes entre Funcat e GO do trabalho de Vens et al. (2008).

Os trabalhos de Chen (2012) e Chen, Duan and Hu (2012) utilizaram as bases de dados FunCat para propor novos métodos para o problema de classificação hierárquico e lidar com o desbalanceamento nos conjuntos de dados. No trabalho de Chen (2012), às classes são divididas em subconjuntos, e neste subconjuntos é aplicado o método de *oversampling Hierarchical SMOTE*, proposto pelos autores. A ideia do *Hierarchical SMOTE* é realizar um *oversampling* nas classes folhas da hierarquia e propagar as classes sintéticas para os pais. Após essa etapa, a classificação é realizada utilizando Máquinas de Vetores de Suporte Probabilísticas (SVMs), com um classificador treinado por classe.

O trabalho de Chen, Duan and Hu (2012) propôs um método de classificação hierárquica multi-rótulo baseado em SVM com *kernel* composto (ck-SVM). As informações de distribuição dos dados de treinamento são usadas para estimar os parâmetros do modelo ck-SVM. Informações de clusterização geralmente são utilizadas para particionar os dados de treinamento em subconjuntos.

No estudo de Chen, Duan and Hu (2012), foi utilizada uma clusterização supervisionada com uma estratégia de *oversampling*. A técnica SMOTE foi aplicada a cada conjunto de dados e, em seguida, os dados foram particionados por meio de uma abordagem de clusterização supervisionada. As classes foram divididas em subconjuntos, e o modelo ck-SVM treinou classificadores binários, que posteriormente foram combinados utilizando o método TPR (*True Path Rule*).

O estudo de Pereira, Costa and Silla Jr (2018) propôs um método para lidar com o desbalanceamento em bases de dados musicais, onde os conjuntos de dados originais hierárquicos multi-rótulos foram convertidos em conjuntos de dados multi-rótulos. Em seguida, uma técnica de reamostragem foi aplicada. Nesse trabalho, foram utilizadas seis técnicas de *resampling*, incluindo o *Multi-label Synthetic Minority Over-sampling Technique* (Charte et al., 2015), que modifica a técnica SMOTE para problemas de classificação multi-rótulos. Após a aplicação das técnicas de *resampling*, os conjuntos de dados multi-rótulos foram reconvertidos para conjuntos de bases hierárquicas multi-rótulos, e o modelo CLUS-HMC foi aplicado.

O trabalho de Pereira et al. (2020) lidou com imagens de radiografias de tórax (CXR) com o objetivo de identificar a COVID-19 em imagens torácicas. A identificação foi realizada utilizando abordagens de classificação multi-rótulo e hierárquica. O método proposto começa com a extração de características das imagens. Após essa etapa, um método de técnica de *early fusion* pode ser aplicado. *Early fusion* é um método em que diferentes características são agrupadas em um único conjunto de características para alimentar o modelo de aprendizado. Depois disso, uma técnica de *resampling* foi aplicada. No método de classificação hierárquica utilizado por Pereira et al. (2020), cada caminho de rótulos associado aos nós folha foi tratado como um rótulo multi-classe individual para aplicar as técnicas de reamostragem. No total, foram utilizadas nove técnicas de *resampling*, incluindo o SMOTE. Para a classificação hierárquica, os classificadores CLUS-HMC foram treinados. Após a fase de treinamento, uma fase de *late fusion* pode ser realizada. *Late fusion*a é uma técnica que combina as saídas dos modelos aprendidos. Essa abordagem demonstrou a aplicação bem-sucedida de métodos de reamostragem e classificação hierárquica em um problema real e relevante, destacando o uso de técnicas avançadas para melhorar a identificação de COVID-19 em imagens de tórax.

O trabalho de Pereira, Costa and Silla Jr (2021b) propôs dois algoritmos de *oversampling* e dois de *undersampling* para lidar com conjuntos de dados desbalanceados. Diversos conjuntos de bases de dados foram utilizados neste estudo. A primeira etapa do método proposto foi identificar os caminhos majoritários e minoritários, que correspondem aos caminhos na árvore hierárquica do conjunto de dados das classes majoritárias e minoritárias. Os quatro algoritmos recebem como entrada o conjunto de bases de dados, o percentual de amostras a ser aumentado

ou reduzido, e como saída produzem o conjunto de base de dados reamostrado. Os algoritmos identificam os caminhos majoritários/minoritários e removem/criam aleatoriamente amostras desses caminhos até que as amostras atinjam a porcentagem determinada pela entrada. Os algoritmos foram divididos em duas categorias: profundidade total (*full depth*) e profundidade parcial (*partial depth*), onde cada tipo foi projetado para lidar com hierarquias com profundidade total ou parcial. Os algoritmos propostos foram testados e comparados com os conjuntos de dados originais, utilizando o CLUS-HMC como modelo para o problema. Tanto Pereira, Costa and Silla Jr (2018) quanto Pereira, Costa and Silla Jr (2021b) mencionaram que, para testar seus métodos com taxonomia em DAG, estes precisam ser revisados e sugeriram a realização desse estudo em trabalhos futuros.

Os trabalhos apresentados até agora demonstram a importância do estudo do desbalanceamento em problemas de classificação hierárquica em problemas reais. Muitos deles utilizaram o método de *resampling* SMOTE, ou o modificaram para o desenvolvimento de novos métodos. Apesar de a utilização de classificadores locais ser um dos métodos mais aplicado na literaturas de acordo com Silla and Freitas (2011), os autores citados até agora não o utilizavam.

Muitos dos trabalhos apresentados utilizam de um método(conhecido ou novo) em combinação com métodos de *resampling*. Na literatura observa-se que métodos de classificadores locais são aplicados com métodos de *resampling*. Os trabalhos de Klungpornkun and Vateekul (2019) e Stiller et al. (2024) utilizaram aprendizado profundo (deep learning), métodos de classificador local por nível e de *resampling* para resolver o problema de classificação hierárquica.

Klungpornkun and Vateekul (2019) propôs o Perceptron Multicamadas Local para Classificação Hierárquica Multi-Rótulo (HMC) e introduziu o conceito de compartilhamento de informações entre camadas. Esse método utiliza camadas compartilhadas, com tamanho controlável de entradas, que aproveitam camadas bem treinadas dos níveis superiores de predição. A primeira etapa do perceptron multicamadas local para HMC é gerar embeddings de palavras utilizando uma rede neural convolucional (CNN). O trabalho seguiu a ideia da abordagem Classificador Local por Nível e criou uma rede neural para cada nível da hierarquia. Treinar uma rede neural requer estratégias de treinamento para calcular o erro do modelo e ajustar lentamente muitas variáveis até que o modelo passe a convergir.

Klungpornkun and Vateekul (2019) utilizou quatro estratégias de treinamento, sendo uma delas a *resampling* balanceada com *mini-batch*. Essa estratégia realiza um *resampling* em cada batch substituindo as classes com baixa frequência para que o modelo seja treinado com *batches* mais balanceados. Ao final do processo, antes da avaliação, foi aplicada uma correção de rótulos para ajustar inconsistências nas predições. Duas abordagens de correção foram utilizadas.

O trabalho de Stiller et al. (2024) relatou que seus resultados foram, em geral, baixos devido a diversos fatores como: ao forte desbalanceamento dos dados, ao modelo hierárquico, ao pequeno conjunto de dados e a outras limitações. As imagens de fungos filamentosos foram coletadas e anotadas com uma hierarquia que segue as categorias: filo > classe > ordem > família > gênero > espécie. Stiller et al. (2024) criou três conjuntos de dados. O conjunto

de dados original, pré-processado de acordo com as categorias hierárquicas. Um conjunto de dados com *naive random oversampling*, no qual a representação das classes minoritárias foi aumentada artificialmente pela duplicação e alteração das imagens. Um conjunto de dados com *augmented random oversampling*, onde, além da duplicação e alteração das imagens, como na abordagem *naive random oversampling*, foram aplicadas transformações mais complexas, como redimensionamento e recorte. Para treinar os dados, foi aplicado um algoritmo de Rede Neural Convolucional (CNN) utilizando três diferentes arquiteturas. Classificadores Locais Separados por Nível: Cada nível possui seu classificador independente, tratado separadamente, sem compartilhamento de informações entre os níveis. Classificador Multi-Rótulo: Um classificador global que considera todos os níveis da hierarquia. Classificadores Locais por Nível Hierarquicamente Encadeados: Passam as características aprendidas de um nível ao próximo na hierarquia.

Podemos observar o uso de classificadores locais também sem a utilização de *deep learning*. O trabalho de De Barros et al. (2021) propôs um esquema que emprega técnicas de *resampling* para balancear a distribuição das classes em imagens de radiografias de tórax (CXR), seguido pelo uso de um Classificador Hierárquico Local. O esquema proposto segue as seguintes etapas: (1) Criação do conjunto de dados e extração de características: As características das imagens foram extraídas utilizando algoritmos de *Local Binary Pattern* (LBP); (2) Estratificação dos dados: Os dados foram estratificados em folds para validação cruzada; (3) Aplicação de técnicas de reamostragem: Seis técnicas de reamostragem foram aplicadas, incluindo o SMOTE; (4) Aplicação da abordagem LCPN: Foi utilizada a abordagem Classificador Local por Nó Pai (LCPN), que consiste em treinar um classificador multi-classe para cada nó que não seja folha. No total, doze classificadores foram utilizados. Essa abordagem combina a extração de características, balanceamento dos dados por reamostragem e a aplicação de classificadores locais para lidar com problemas hierárquicos, demonstrando uma estratégia eficaz para melhorar a performance de classificação em bases desbalanceadas.

Apesar de alguns dos trabalhos vistos utilizarem classificadores locais, eles não chegam a empregar LCN. O trabalho de Pereira, Costa and Silla Jr (2021a) é o mais similar ao presente estudo. Pereira, Costa and Silla Jr (2021a) propôs esquemas de reamostragem para problemas de classificação hierárquica utilizando a abordagem de classificadores locais. As abordagens de Classificadores Locais estudadas foram: Classificador Local por Nó (LCN), Classificador Local por Nível (LCL) e Classificador Local por Nó Pai (LCPN). Além disso, foram propostas três métricas de desbalanceamento, representadas pelas equações IR_{LCN}, IR_{LCPN} e IR_{LCL}, para medir o desbalanceamento em conjuntos de base de dados hierárquicos para classificadores locais. Embora todos os classificadores locais possuam sua relevância, este trabalho se concentra em explicar a abordagem LCN em detalhes. O IR_{LCN} é baseado na definição da proporção entre amostras positivas e negativas ao construir classificadores binários.

Pereira, Costa and Silla Jr (2021a) investigou o uso de três diferentes abordagens de *resampling* acopladas às três abordagens de classificadores locais estudadas em seu trabalho. Esse estudo fornece uma base sólida para avaliar como técnicas de *resampling* podem ser integradas a diferentes estratégias de classificação hierárquica, destacando a importância de abordar o

desbalanceamento em conjunto de bases de dados. Os experimentos de classificação plana com algoritmos de *resampling* multi-rótulo realizados no estudo de Pereira, Costa and Silla Jr (2021a) serviram para estabelecer resultados de referência (*baseline*). A ideia da abordagem de resampling para o LCN era reamostrar cada conjunto de dados binarizado antes de construir o modelo de classificação. A Figura 11 Pereira, Costa and Silla Jr (2021a) apresenta o esquema geral dessa abordagem de reamostragem para o LCN. O esquema possui três etapas principais: (1) Construção de um classificador binário por nó rotulado na hierarquia; (2) Classificação do conjunto de dados de teste; (3) Avaliação dos resultados da classificação com métricas hierárquicas. A primeira etapa é subdividida em três subetapas: (1.1) Aplicação de uma política escolhida, como visto na Seção 2.2, para determinar as amostras positivas e negativas; (1.2) Aplicação de um algoritmo plano de *resampling* binária no conjunto de dados binarizado (foram testados 13 algoritmos de reamostragem, incluindo o SMOTE); (1.3) Aplicação de um algoritmo plano de classificação de rótulo único para construir o modelo de classificação. Pereira, Costa and Silla Jr (2021a) também utilizou uma abordagem top-down na fase de teste para evitar inconsistências na predição de classes em diferentes níveis.

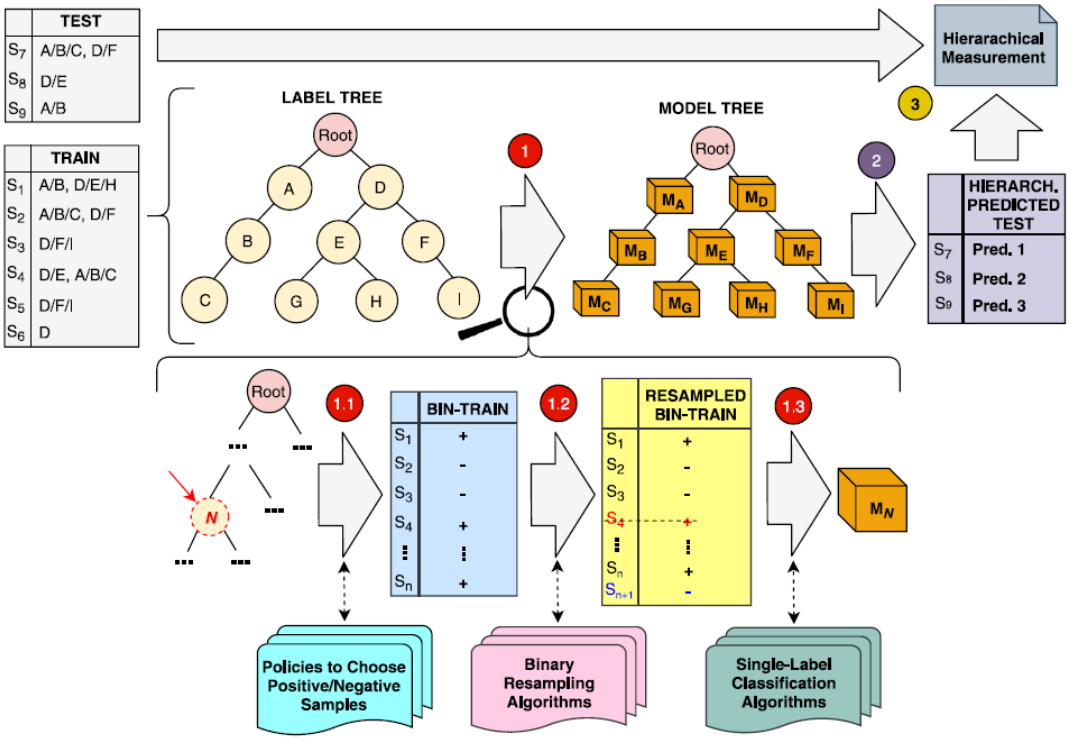


Figura 11 – O esboço geral de classificação para a abordagem LCN (Pereira; Costa; Silla Jr, 2021a).

Na seção de resultados, foi observado que o LCN apresentou o melhor desempenho, e o SMOTE e suas variações foram os mais eficazes. Além disso, foi comprovada a hipótese de que não havia diferença significativa entre a abordagem plana e todas as abordagens locais. Entretanto, ao comparar o esquema de classificação hierárquica de *resampling* com sua linha de base (*baseline*), foi observado que o esquema pode melhorar os resultados da classificação. Pereira, Costa and Silla Jr (2021a) destacou, porém, uma preocupação com conjuntos de dados hierárquicos de grande escala, já que a técnica proposta não conseguiu lidar adequadamente

com o desbalanceamento e melhorar significativamente os resultados. O estudo sugere como trabalhos futuros a análise do problema com estruturas de DAG, tema explorado neste trabalho com a abordagem LCN.

O método utilizado neste estudo é estreitamente relacionado ao esquema apresentado por Pereira, Costa and Silla Jr (2021a), mas com adaptações necessárias devido à estrutura da taxonomia em DAG, especialmente no que diz respeito às políticas do LCN. Foi possível perceber que o estudo de classificação hierárquica utilizando políticas LCN em bases desbalanceadas é um de problemática ainda não explorada, e que a academia obteve um foco nas bases com taxonomia de árvore por ter uma complexidade menor. Podemos perceber que o SMOTE foi um dos métodos de *resampling* mais utilizado junto a problemas de classificação hierárquica, fazendo com que este método seja um ótimo candidato para experimentos neste campo.

Pereira, Costa and Silla Jr (2021a) comentou sobre conjunto de dados de grande escala e sua dificuldade em lidar com o desbalanceamento. O trabalho de Serrano-Pérez and Sucar (2021) apresenta conjuntos de bases de dados artificiais que possuem profundidade e quantidade menores do que vistos até o momento em bases reais. O estudo comparou dez métodos que utilizam redes bayesianas e classificadores encadeados. Métodos planos e *top-down*, HBA (proposto por Barutcuoglu et al. (2008) e posteriormente adaptado por Serrano-Pérez and Sucar (2019)), CLUS-HMC (proposto por Vens et al. (2008)), CPE (proposto por Ramírez-Corona, Sucar and Morales (2016)), nLLCPN e LCPNB (propostos por Nakano et al. (2017)), Redes Bayesianas com Classificadores Encadeados (propostos por Serrano-Pérez and Sucar (2019)). Pereira, Costa and Silla Jr (2021a) exploram tanto taxonomias em árvores quanto em DAG, com diferentes níveis de dificuldade. Os conjuntos de dados foram divididos em balanceados e desbalanceados. Apesar de os conjuntos sintéticos incluírem exemplos desbalanceados, o autor não aplicou nenhum método de reamostragem. Os métodos de comparação foram aplicados sem o uso de técnicas de reamostragem nesses conjuntos de bases de dados. Essa lacuna destaca uma oportunidade para investigar como métodos de reamostragem podem melhorar os resultados em problemas de classificação hierárquica utilizando esses conjuntos de dados sintéticos, especialmente em contextos com taxonomias complexas como DAGs.

Os conjuntos de bases de dados sintéticos têm sido amplamente utilizados em diversas aplicações, incluindo problemas de classificação hierárquica multidimensional (MDHC). O trabalho de Montenegro, Santana and Lozano (2023) introduz problemas de MDHC, que envolvem múltiplas variáveis de classe que precisam ser previstas conjuntamente. O estudo propôs quatro estratégias combinando duas estratégias de classificação multidimensional (MDC) com duas estratégias de classificação hierárquica (HC). As estratégias de MDC são : (1) *Stacking*: Um procedimento de regularização em que as predições são obtidas aprendendo cada rótulo separadamente e corrigindo-as com as predições dos outros rótulos; (2) *Grouping*: Uma estratégia baseada em correlação, onde o coeficiente de correlação é medido para cada par de classes, dividindo-as em dois grupos. Classes independentes são treinadas utilizando um classificador único para cada classe, enquanto classes dependentes são treinadas com o mesmo classificador para o grupo. As estratégias de HC são: (1) Classificador Local por Nó Pai (LCPN): Onde um

classificador multi-classe é treinado para cada nó pai; (2) Classificador Global (GC): Treina um único classificador que considera toda a hierarquia simultaneamente.

O estudo de problema de classificação utilizando classificadores locais vem sendo utilizado e testados em diversos tipos de bases. Este campo de estudos ainda tem espaço para experimentos e avanços.

4 Metodologia

Neste capítulo é apresentado o processo e etapas utilizadas na realização deste estudo. A Figura 12 apresenta este processo e etapas. Este processo é realizado com dois conjuntos de bases, um sintético e um genético. A dificuldade em encontrar conjunto de bases reais com taxonomia em DAG fez com que este trabalho escolhesse um conjunto sintético para agregar ao estudo.

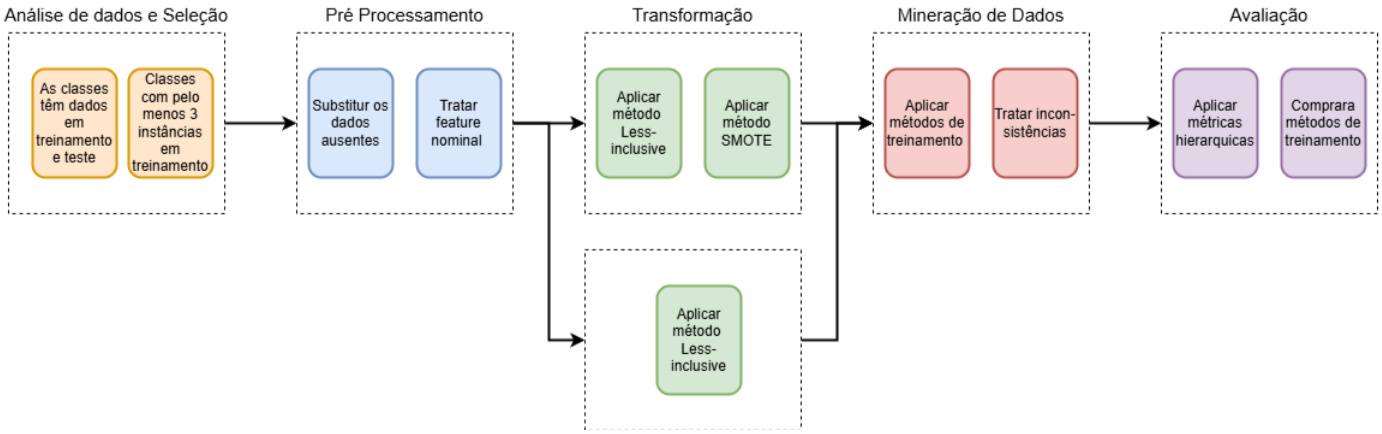


Figura 12 – Processo e etapas da metodologia.

As seções a seguir apresentam as etapas. A seção 4.1 apresenta as bases de dados utilizadas e a seleção dos dados. A seleção é focada em desenvolver uma base que seja possível utiliza-la tanto nos experimentos sem *resampling* quanto com *resampling*. Na seção 4.2 de pre-processamento é apresentado como dados faltantes e *features* nominais foram tratadas. Na seção 4.3 de transformação é apresentado a transformação necessária a ser realizada nas bases quando se aplica o método local por nó *less inclussive* e o método de *resampling* SMOTE. Na seção 4.4 de mineração de dados é apresentado o classificador utilizado para o treinamento e como tratar inconsistências. Por fim, a seção 4.5 apresenta como foi realizada a escolha das métricas e sua avaliação.

4.1 Análise Dados e Seleção

Este trabalho utilizou dois conjuntos de bases para realizar os experimentos. O primeiro conjunto de bases é do trabalho de Serrano-Pérez and Sucar (2021), os quais são sintéticos, criados especificamente para problemas de classificação hierárquica. Os conjuntos de dados são divididos em taxonomias de árvore e DAG.

Os conjuntos de dados com taxonomia de árvore são subdivididos em níveis de dificuldade: *easy*, *hard* e *very hard*. Os conjuntos de dados com taxonomia em DAG são divididos em dois níveis de dificuldade: *easy* e *hard*. Serrano-Pérez and Sucar (2021) utilizou o conceito *easy* e

hard de problemas de classificação onde a dificuldade de baseia na distribuição das classes. A Figura 13 apresenta uma representação do problema *easy* e *hard*. No exemplo um problema *easy* possui as classes A(class_A) e C(class_C), onde a distribuição de classes não possui interseção e um classificador consegue facilmente distingui-las. Um problema considerado *easy* é caracterizado, por exemplo, pela distribuição das classes A(class_A) e B(class_B) na Figura 13, onde está distribuição se sobrepõe e a tarefa de distinção de classes será mais desafiadora para um classificador.

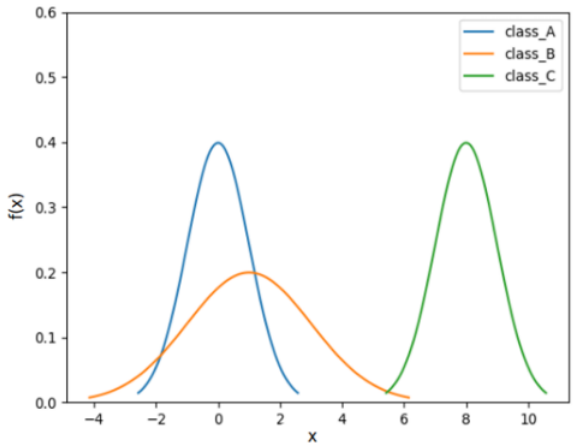
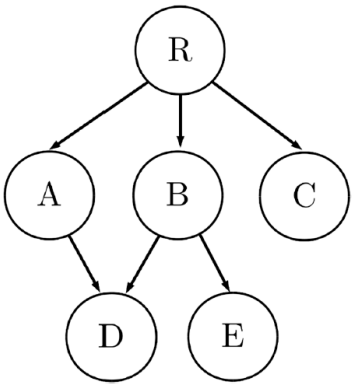


Figura 13 – Exemplo da distribuição de dados das classes A, B e C. Distinguir entre instâncias das classes A e C é *easy*, enquanto para as classes A e B é *hard*. (Melhor visualizado em cores.)(Serrano-Pérez; Sucar, 2021)

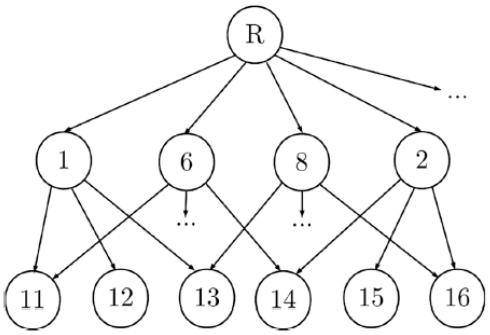
Serrano-Pérez and Sucar (2021) desenvolveu algoritmos que utilizam o conceito *easy* e *hard* para a criação de conjunto de dados hierárquicos. Nas bases de dados desenvolvidas a bases *easy* são caracterizadas pelas folhas e os ancestrais da hierarquia possuírem distribuição similares além de a interseção entre as classes ser pequena. Na Figura 14a é possível observar um exemplo de hierarquia *easy* para taxonomia em DAG e na Figura 15 é representado como a distribuição de cada classe(nó) que foi desenvolvida. Agora para as bases *hard* e *very hard* a interseção entre as classes é alta. Na Figura 14b é possível observar um exemplo de hierarquia *hard* para taxonomia DAG e na Figura 16 é representado como a distribuição de cada classe(nó) que foi desenvolvida.

Os conjuntos de bases com taxonomia de árvores *easy* produziram doze conjuntos de bases, enquanto os *hard* e*very hard* produziram vinte conjuntos cada. Os conjuntos de dados DAG *easy* quanto os *hard* produziram doze conjuntos de bases cada. Este trabalho vai utilizar as bases de taxonomia DAG e desbalanceadas, considerando os objetivos deste trabalho. Os conjuntos de bases de taxonomia DAG *easy* podem ser encontrados na Tabela 3, e os conjuntos de dados difíceis estão listados na Tabela 4. Para taxonomias em DAG, conjuntos de dados *very hard* não foram desenvolvidos por Serrano-Pérez and Sucar (2021).

Cada conjunto de bases DAG possui três estruturas de dificuldade. As estruturas têm os mesmos atributos (Attr.), nós e profundidade máxima (MD). O trabalho de Serrano-Pérez and Sucar (2021) gerou quatro conjuntos de dados para cada estrutura. Os conjuntos de dados de



(a) Estrutura hierárquica (DAG) usada para os conjuntos de dados fáceis cujo nome começa com D_EA_01. (Serrano-Pérez; Sucar, 2021)



(b) Estrutura hierárquica parcial (DAG) usada para conjuntos de dados cujo nome começa com D_HA_01. (Serrano-Pérez; Sucar, 2021)

Figura 14 – Exemplo de Estrutura Hierárquica

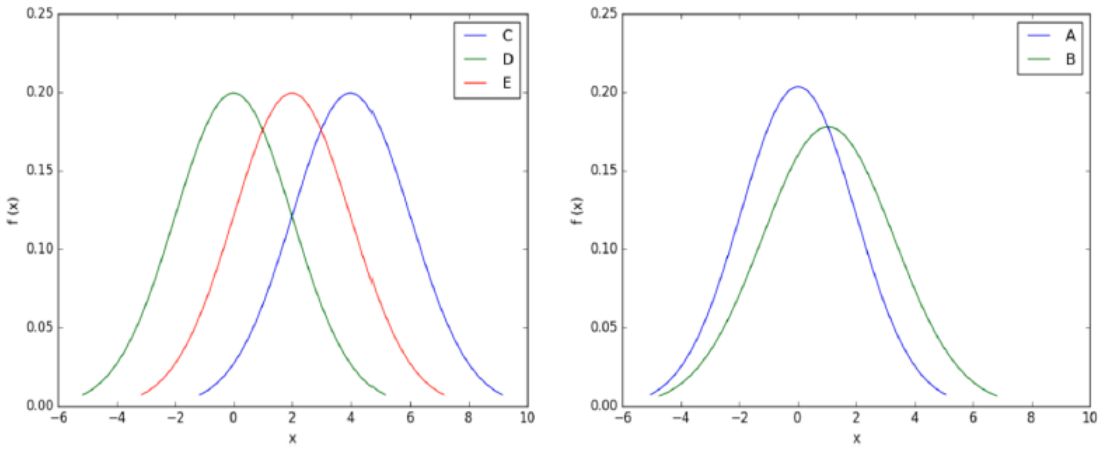


Figura 15 – Distribuições associadas à hierarquia de conjuntos de dados D_EA_01. Esquerda: Distribuições fornecidas para os nós folha. Direita: Distribuição estimada para os nós internos. (Melhor visualizado em cores.)(Serrano-Pérez; Sucar, 2021)

cada estrutura podem ter: profundidade total e balanceado, profundidade total e desbalanceado, profundidade parcial e balanceado, e profundidade parcial e desbalanceado. Os conjuntos de dados recebem nomes que refletem o tipo e a estrutura. Por exemplo, o primeiro conjunto de dados da Tabela 3, D_EA_01_FD_b, onde D significa DAG, EA significa Fácil (Easy), 01 representa o primeiro tipo de estrutura, FD indica Profundidade Total (*Full Depth*) e b significa Balanceado (*balanced*). Entre parênteses, este trabalho renomeia os conjuntos de dados da Tabela 3 como DSEx, onde x representa o número do conjunto de dados atribuído por este trabalho. Outro exemplo da Tabela 4 : D_HA_01_FD_b, onde D significa DAG, HA significa Difícil (*Hard*), 01 representa o primeiro tipo de estrutura, FD indica Profundidade Total (*Full Depth*) e b significa Balanceado (*balanced*). Entre parênteses, este trabalho renomeia os conjuntos de dados da Tabela 4 como DSHx, onde x representa o número do conjunto de dados atribuído por este trabalho.

O segundo conjunto de dados é do trabalho de Vens et al. (2008), que foi baseado no

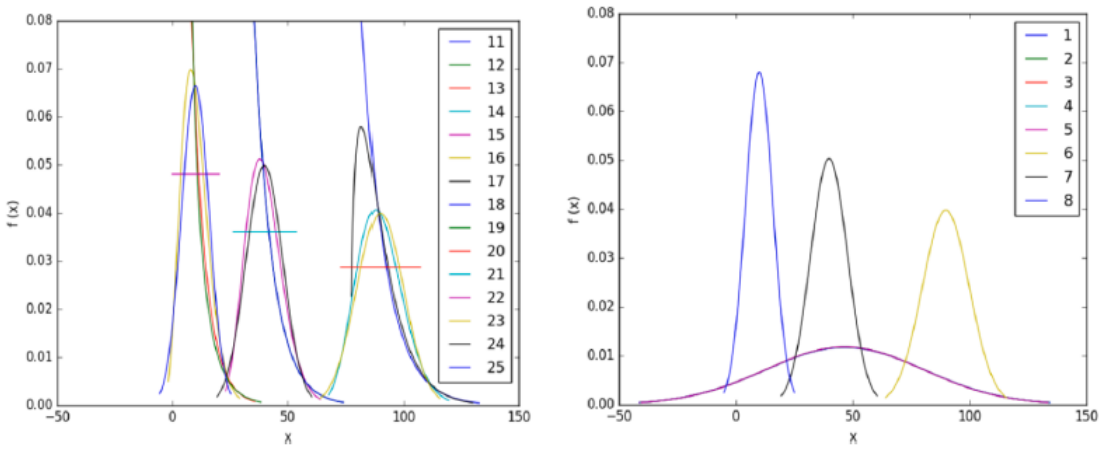


Figura 16 – Distribuições associadas à hierarquia de conjuntos de dados D_HA_01. Esquerda: Distribuições fornecidas para os nós folha. Direita: Distribuições estimadas para os nós internos. Observe que as distribuições estimadas para os nós 1, 2, 3, 4 e 5 são as mesmas, portanto, as linhas se sobrepõem; no entanto, essa situação foi gerada intencionalmente. (Melhor visualizado em cores.) (Serrano-Pérez; Sucar, 2021)

Datasets	Train	Test	Attr.	Nodes	MD
D_EA_01_FD_b (DSE1)	120	60	9	5	2
D_EA_01_FD_ub (DSE2)	240	120	9	5	2
D_EA_01_PD_b (DSE3)	148	74	9	5	2
D_EA_01_PD_ub (DSE4)	274	137	9	5	2
D_EA_02_FD_b (DSE5)	160	80	5	7	2
D_EA_02_FD_ub (DSE6)	360	180	5	7	2
D_EA_02_PD_b (DSE7)	202	101	5	7	2
D_EA_02_PD_ub (DSE8)	454	227	5	7	2
D_EA_03_FD_b (DSE9)	120	60	6	5	2
D_EA_03_FD_ub (DSE10)	240	120	6	5	2
D_EA_03_PD_b (DSE11)	148	74	6	5	2
D_EA_03_PD_ub (DSE12)	274	137	6	5	2

Tabela 3 – Descrição dos conjuntos de dados fáceis (DAG). MD: Profundidade Máxima. No nome do conjunto de dados, FD indica caminhos com Profundidade Total (*Full Depth*), PD indica Profundidade Parcial (*Partial Depth*); b significa folhas balanceadas e ub folhas desbalanceadas. Em cada conjunto de dados, cada nó possui pelo menos 40 instâncias associadas ao conjunto de treinamento e 20 instâncias associadas ao conjunto de teste. (Serrano-Pérez; Sucar, 2021)

trabalho Ashburner et al. (2000). A *Gene Ontology* é amplamente utilizado nos problemas de classificação hierárquica mais estudados, estando relacionado ao domínio da bioinformática.

Os problemas envolvendo genes geralmente utilizam anotações dos genes para produzir conjuntos de dados. Atualmente, existem dois projetos bem conhecidos que catalogaram o esquema de proteínas a partir de genomas:

- GO - O projeto *Gene Ontology* (GO) foi desenvolvido para construir um vocabulário estruturado e bem definido que descreve as funções de genes e produtos gênicos em

Datasets	Train	Test	Attr.	Nodes	MD
D_HA_01_FD_b (DSH1)	600	300	10	23	2
D_HA_01_FD_ub (DSH2)	1120	560	10	23	2
D_HA_01_PD_b (DSH3)	712	356	10	23	2
D_HA_01_PD_ub (DSH4)	1324	663	10	23	2
D_HA_02_FD_b (DSH5)	240	120	9	17	3
D_HA_02_FD_ub (DSH6)	480	240	9	17	3
D_HA_02_PD_b (DSH7)	394	197	9	17	3
D_HA_02_PD_ub (DSH8)	769	386	9	17	3
D_HA_03_FD_b (DSH9)	520	260	2	30	5
D_HA_03_FD_ub (DSH10)	1040	520	2	30	5
D_HA_03_PD_b (DSH11)	758	379	2	30	5
D_HA_03_PD_ub (DSH12)	1477	742	2	30	5

Tabela 4 – Descrição dos conjuntos de dados difíceis (DAG). MD: Profundidade Máxima. No nome do conjunto de dados, FD indica caminhos com Profundidade Total (*Full Depth*), PD indica Profundidade Parcial (*Partial Depth*); b significa folhas balanceadas e ub folhas desbalanceadas.Em cada conjunto de dados, cada nó possui pelo menos 40 instâncias associadas ao conjunto de treinamento e 20 instâncias associadas ao conjunto de teste. (Serrano-Pérez; Sucar, 2021)

organismos (Ashburner et al., 2000)

- FunCat - O *Functional Catalog* (FunCat) é um sistema de classificação hierarquicamente estruturado e controlado que descreve as funções de proteínas de um organismo (Ruepp et al., 2004)

O trabalho de Vens et al. (2008) produziu 24 conjuntos de dados a partir de anotações de projetos de genes. Vens et al. (2008) utilizou os conjuntos de dados do Funcat e do GO. Os conjuntos de dados do Funcat possuem uma hierarquia de classes estruturada em árvore, enquanto os conjuntos de dados do GO têm uma hierarquia de classes estruturada em DAG.Neste trabalho, estamos interessados em explorar o uso do algoritmo de reamostragem SMOTE em conjuntos de dados estruturados em DAG. Portanto, utilizaremos apenas os dados relacionados ao *Gene Ontology*. Considerando os existentes bancos de dados de Classificação Hierárquica do *Gene Ontology*, temos doze bases de dados.

As Tabelas 5 e 6 mostram alguns detalhes sobre as bases de dados GO.

	FunCat	**GO**
Scheme version	2,1 (2007/01/09)	1,2 (2007/04/11)
Yeast annotations	2007/03/16	2007/04/07
Total classes	1362	22960
Average number of classes per dataset	492 (6 levels)	3997 (14 levels)
Average number labels per example	8,8 (3,2 most specific)	35,0 (5,0 most specific)

Tabela 5 – Propriedades dos conjuntos de dados FunCat e GO, Tabela adaptada do trabalho de Vens et al. (2008).

Dataset	**|D|**	**|A|**
seq	3932	478
pheno	1592	69
struc	3851	19628
hom	3867	47034
cellcycle	3766	77
church	3764	27
derisi	3733	63
eisen	2425	79
gasch1	3773	173
gasch2	3788	52
spo	3711	80
expr	3788	551

Tabela 6 – Propriedades dos conjuntos de dados, onde |D| representa o número de instâncias e |A| representa o número de atributos. Tabela adaptada do trabalho de Vens et al. (2008).

A primeira etapa do processo metodológico possui duas sub-etapas, a primeira que foca na avaliação se instâncias de uma classe existem em teste existem também em treino, e vice versa. A segunda foca em analisar a quantidade de instância para verificar se cada base possui o mínimo para a aplicação do SMOTE. Esta primeira etapa representa uma adequação do conjunto de bases real GO, uma vez que o conjunto de bases sintéticas não possui dificuldade em questão de falta de instâncias por classe.

A primeira sub-etapa mostra a importância da verificação de bases reais. Quando utilizamos um classificador para realizar o treinamento de uma classe se espera que esta mesma classe possua instâncias nas bases de testes. Caso não existam instâncias em testes não é possível calcular as métricas hierárquicas para realizar a comparação entre os modelos. E quando encontramos classes que possuem instâncias na base de teste e não possuem na base de treino significa que essa classe não foi treinada no modelo, assim não será possível realizar a previsão desta.

Para o conjunto de dados GO, a análise de instâncias mostra que, para o conjunto de dados *church*, existem classes que possuem exemplos no conjunto de treinamento, mas não no conjunto de teste, e também classes que possuem exemplos no conjunto de teste, mas não no conjunto de treinamento. O conjunto de dados *church* possui 2.432 classes nos conjuntos de treinamento e teste. A distribuição das classes para o conjunto de dados *church* pode ser encontrada no Apêndice A.

A segunda etapa é verificar se há ou não pelo menos três instâncias por classe nos conjuntos de bases de treinamento. Contar as instâncias é essencial, pois os conjuntos de bases precisam possuir certas características específicas para que possamos aplicar técnicas de reamostragem. Considerando que estamos utilizando o algoritmo SMOTE, o número de instâncias por classe no conjunto de treinamento deve ser, no mínimo, três. Caso contrário, não seria possível executar o algoritmo SMOTE.

Nossa análise sobre o número mínimo de instâncias por classe no conjunto de treinamento mostrou que 479 classes da base de dados *church* podem ser utilizadas com o SMOTE. Dado os problemas encontrados no conjunto de dados *church*, para que possamos utilizar os conjuntos de dados apresentados na Tabela 6 em nossa pesquisa, utilizaremos subconjuntos dos conjuntos de dados originais que respeitem as seguintes restrições:

- A classe deve possuir instâncias tanto na base de treinamento quanto na base de teste.
- As classes devem possuir pelo menos três instâncias na base de treinamento.

4.2 Pre-processamento

A etapa de pré-processamento consiste em analisar se existem dados ausentes ou se existem dados nominais que não são aceitos pelo modelo como entrada do modelo para treinamento. O conjunto de bases sintéticas não possuem problemas de dados faltantes e não possuem dados nominais, assim esta etapa é realizada para o conjunto de bases GO.

Os atributos no conjunto de bases GO são nominais e numéricos. Também existem dados ausentes, representados como '?' nos conjuntos de dados originais. Para lidar com os atributos ausentes, nesta etapa da pesquisa, foi aplicada a exclusão de instâncias com valores ausentes nos atributos (Grzymala-Busse; Grzymala-Busse, 2010). No entanto, quando a porcentagem de dados ausentes em algumas colunas ultrapassa 50%, a característica foi excluída.

Considerando que, neste trabalho, utilizaremos algoritmos que não conseguem lidar com atributos nominais, é necessário transformar essas características. Para isso, aplicamos uma transformação que divide um atributo nominal em múltiplos atributos binários, onde cada atributo binário corresponde a um rótulo específico.

A Figura 17 apresenta um exemplo com uma classe-alvo de animais e quatro exemplos.

Animals
Cat
Dog
Bird, Cat
Lizzard, Dog

Cat	Dog	Bird	Lizzard
1	0	0	0
0	**1**	0	0
1	0	**1**	0
0	**1**	0	**1**

Figura 17 – Exemplo de transformação de uma característica de texto em uma característica binária.

4.3 Transformação

Neste trabalho, lidamos com problemas de classificação hierárquica com classes organizadas em um DAG. De acordo com o Silla and Freitas (2011) os métodos de classificadores locais por nó (LCN) são os mais utilizados para resolução de classificação hierárquica. O método LCN possui seis políticas onde elas se diferenciam pela definição de exemplos positivos e negativos. Algumas política s excluem exemplos e outro incluem exemplos como positivos ou negativos. De acordo com Eisner et al. (2005) quanto mais inclusivo é a política sua performance se observa melhor. Assim para se iniciar a seleção de uma política foram excluídos os exclusivos *exclusive*, *less exclusive* e *exclusive siblings*. Ficando com as política s*inclusive* para seleção. O trabalho de Fagni and Sebastiani (2007) demonstrou que as políticas *less inclusive* e *siblings* possuíam as melhores performances e entre eles não existia um vencedor.

Apesar de as política s *less inclusive* e *siblings* possuírem performance parecidas de acordo com Fagni and Sebastiani (2007) este trabalho decidiu trabalhar com a política *less inclusive*. A política *less inclusive* considera seus ancestrais como exemplo negativos e já a política *siblings* desconsidera os seus ancestrais. Assim foi escolhido a política que permanece com a hierarquia no treinamento.

Para selecionar os exemplos positivos e negativos no treinamento, é necessário empregar uma política de treinamento para exemplos positivos e negativos (mais detalhes na Seção 2.2). É então realizado uma transformação antes de cada treinamento para se adequar a política escolhida.

Existem inúmeros métodos de *resampling* a serem empregados e estudados, este trabalho escolheu trabalhar com o método SMOTE. De acordo com Haixiang et al. (2017) e Spelmen and Porkodi (2018) o SMOTE é o método mais utilizado para problemas onde existe a necessidade de aumentar a classe minoritária. Uma vez que cada nó ou folha vai se transformar em um classificador, significa que cada caso vai passar por uma transformação da base pelo SMOTE. Assim, este método foi escolhido uma vez que desejamos aumentar a quantidade da classe minoritária dentro de cada base dos classificadores nó ou folhas.

Por fim, esta etapa começa com a seleção de um nó ou folha. É feito uma cópia da base. Depois a política *less inclusive* é aplicada e em seguida o método SMOTE é aplicado. Lembrando que esta base transformada é uma entrada para um classificador binário.

4.4 Mineração de Dados

Quando observamos o conjunto de bases nas Tabelas 3 e 4 sintéticas observamos que o número de instâncias não é alto, assim como o número de classes. Agora observando nas Tabelas 5 e 6 o conjunto de base da GO, podemos ver o número de instâncias passa dos milhares assim como o número médio de classes por base. Uma vez que utilizando a política local por nó

less inclusive foi treinado, utilizando o classificador Naive bayes, classificadores binários que representam cada nó na hierarquia das bases de dados. O classificador Naive Bayes foi escolhido por este ser mais simples e possuir um tempo de processamento menor comparado com outros classificadores de acordo com Krishnaiah, Narsimha and Chandra (2014).

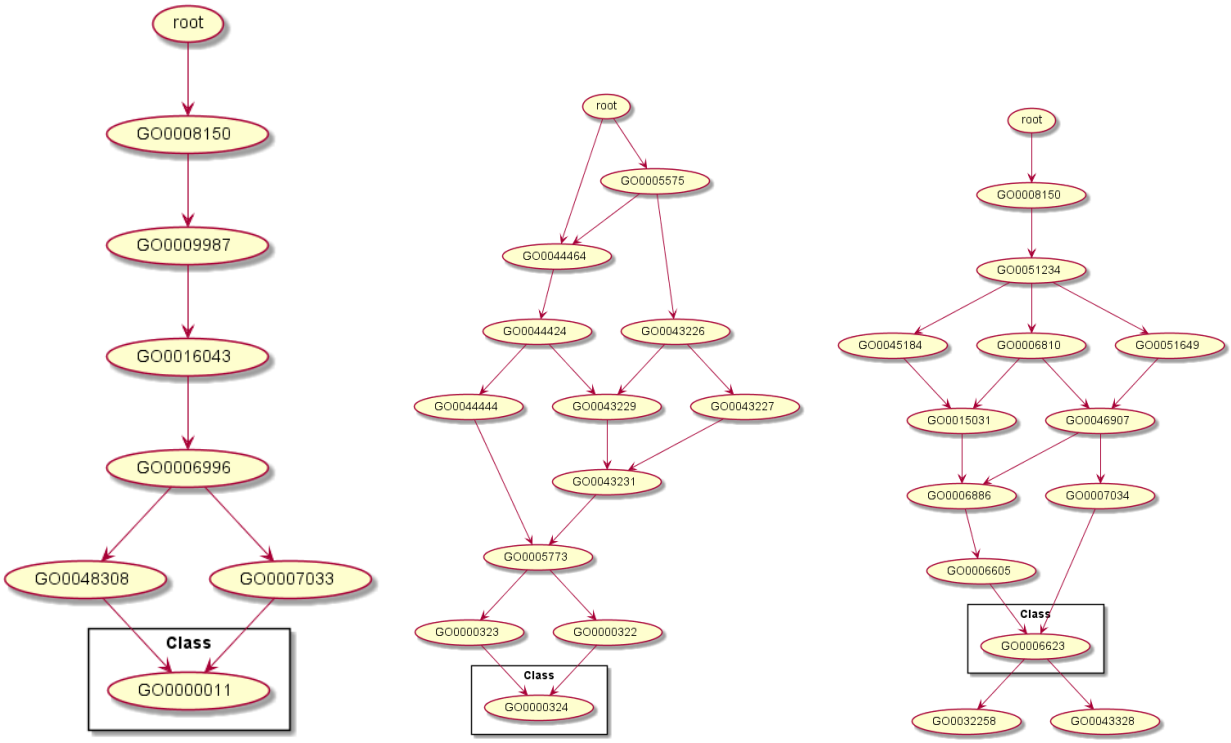
Nesta etapa foi realizado o treinamento utilizando o classificador Naive Bayes e o tratamento de inconsistências. Para ilustrar a sub-etapa de treinamento, utilizemos um exemplo real do conjunto de dados *church*: B, 20, 0.11, 0.1, 0.1, 0.14, -0.1, 0, 0, 0.11, 0.02, 0, 0, 0.15, 0.9, 0, 0.17, 0, 0, ?, 1, ?, ?, 5, 1, 4, 0, GO0005515 @ GO0000324 @ GO0016237 @ GO0000011 @ GO0042144 @ GO0006623. Conceitos da política de treinamento *less inclusive* será demonstrada para melhor entendimento desta sub-etapa.

Neste exemplo, a instância de dados possui seis classes: GO0000324 (Fig. 18b), GO0016237 (Fig. 18d), GO0000011 (Fig. 18a), GO0042144 (Fig. 18e), GO0006623 (Fig. 18c) e GO0005515 (Fig. 19 e 20).

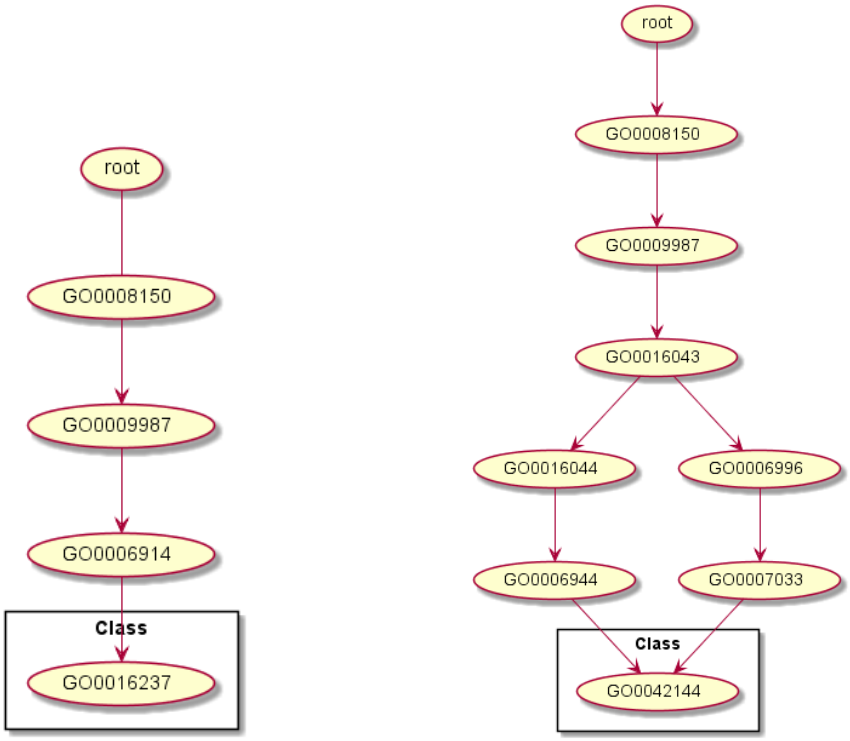
As Figuras 18 ilustram que cada classe aparece em diferentes partes da estrutura em DAG. Considerando que estamos utilizando a abordagem de classificador local por nó com a política de treinamento *less inclusive*, um classificador binário é treinado individualmente para cada uma dessas classes.

Por exemplo, para a classe GO0000011, todos os dados de sua classe e dos nós ancestrais (GO0007033, GO0048308, GO0006996, GO0016043, GO0009987 e GO0008150) são utilizados como exemplos positivos de treinamento. O restante dos dados foi utilizado como exemplos negativos de treinamento.

Note que, neste exemplo, a classe não possui nenhum nó filho; caso contrário, os dados dos nós filhos também seriam utilizados como parte dos exemplos positivos de treinamento.



(a) GO0000011 (b) GO0000324 (c) GO0006623



(d) GO0016237 (e) GO0042144

Figura 18 – Exemplos de hierarquia de classes.

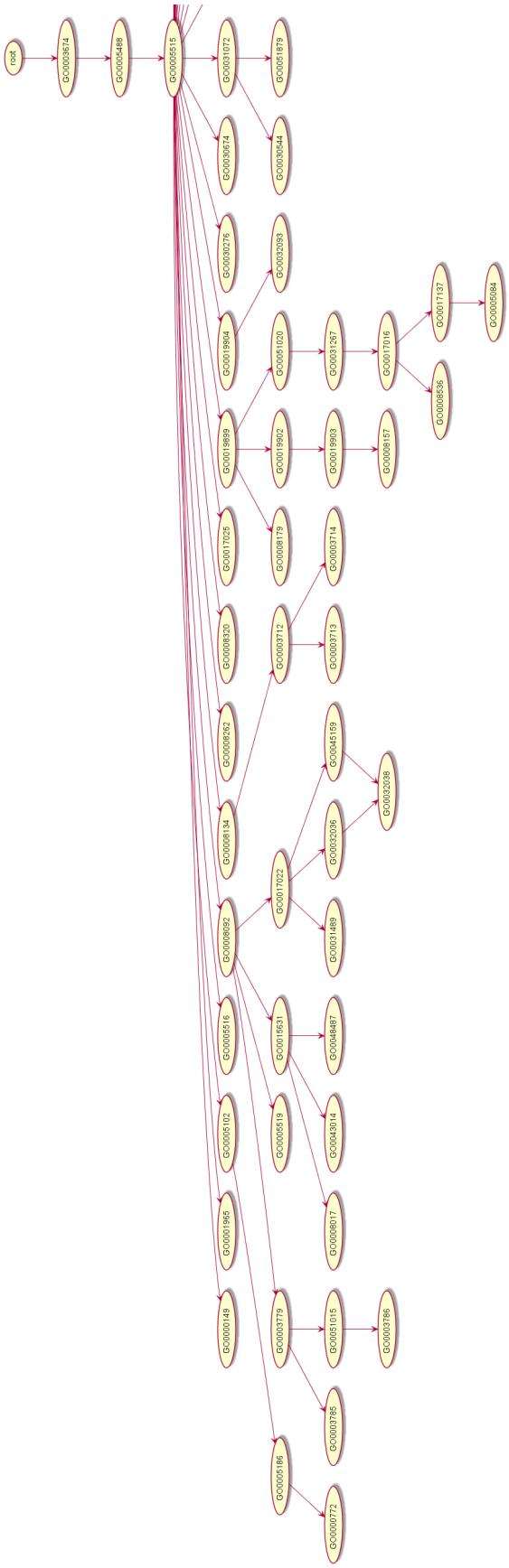


Figura 19 – Parte do exemplo de hierarquia de classes - GO0005515 lado esquerdo.

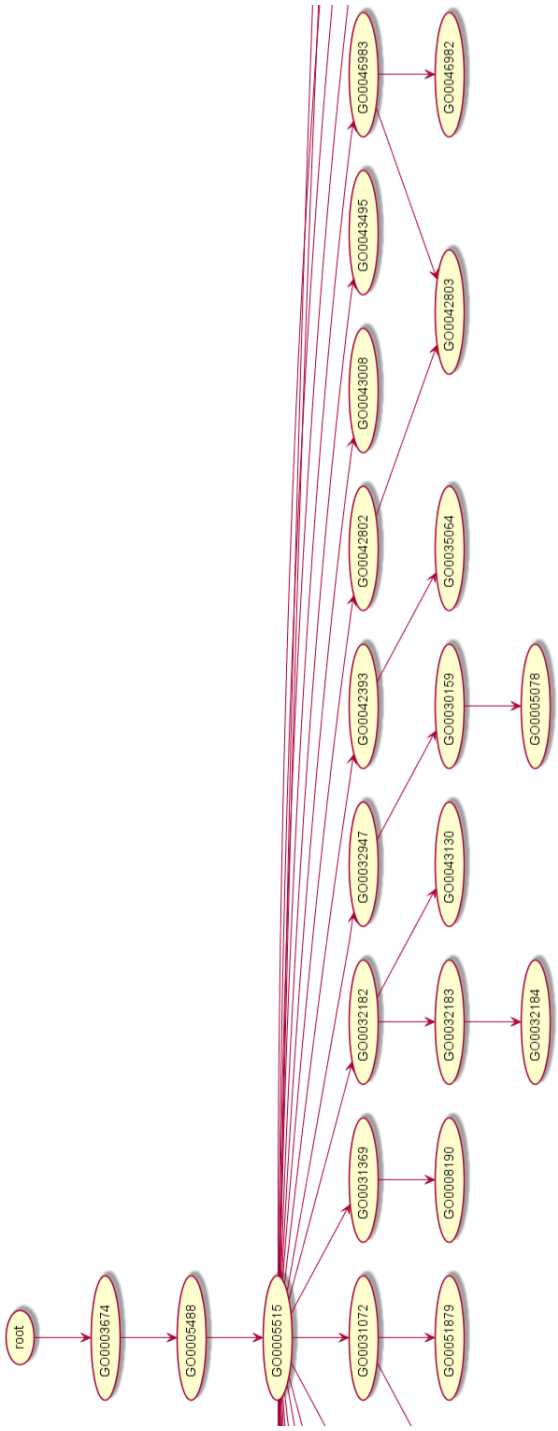


Figura 20 – Parte do exemplo de hierarquia de classes - GO0005515 lado direito.

Na fase de teste, cada um dos classificadores binários treinados durante a fase de treinamento foi utilizado para prever todas as possíveis classes para cada exemplo de teste.

Um exemplo será demonstrado para ilustrar a fase de teste. A Figura 18c será utilizada como exemplo, e GO0006623 será a classe-alvo.

Imagine que temos dois exemplos de teste. Cada exemplo passará por todos os classificadores binários. No entanto, é necessário analisar as respostas levando em conta a hierarquia de GO0006623, que consiste nos classificadores binários: GO00008150, GO0051234, GO0045184, GO00006810, GO0051649, GO0015031, GO0046907, GO0006886, GO0007034, GO0006605, GO0006623, GO0032258 e GO0043328.

Para analisar essas respostas, primeiro devemos verificar qual deve ser a resposta verdadeira. A Figura 21 representa o exemplo verdadeiro da classe GO0006623. As classes em azul devem receber uma resposta positiva do classificador binário, enquanto as classes em laranja devem receber uma resposta negativa do classificador binário.

Após serem apresentados às características, os dois exemplos demonstram as respostas aos classificadores binários.

O exemplo na Figura 22 demonstra uma instância onde GO00008150, GO0051234, GO0045184, GO00006810, GO0051649, GO0015031 e GO0046907 receberam respostas positivas, enquanto GO0006886, GO0006605, GO0007034, GO000660 e GO0006623 receberam respostas negativas. As respostas mostraram que apenas metade da hierarquia foi classificada como positiva, e a classe-alvo GO0006623 não foi prevista corretamente, sendo classificada como negativa.

O exemplo na Figura 23 demonstra uma instância onde GO00008150, GO0051234, GO0045184, GO00006810, GO0051649, GO0015031, GO0046907, GO0006886, GO0007034, GO000660 e GO0006623 foram classificados como positivos, enquanto GO0006605 foi classificado como negativo. Este exemplo demonstrou um tipo de inconsistência que pode ser encontrado quando apenas um classificador binário responde negativamente em toda a hierarquia.

Este trabalho utilizou o ajuste de inconsistência onde consideramos a resposta do classificador binário GO0006605 como positiva em vez de negativa, uma vez que a classe-alvo e todos os outros classificadores foram classificados como positivos.

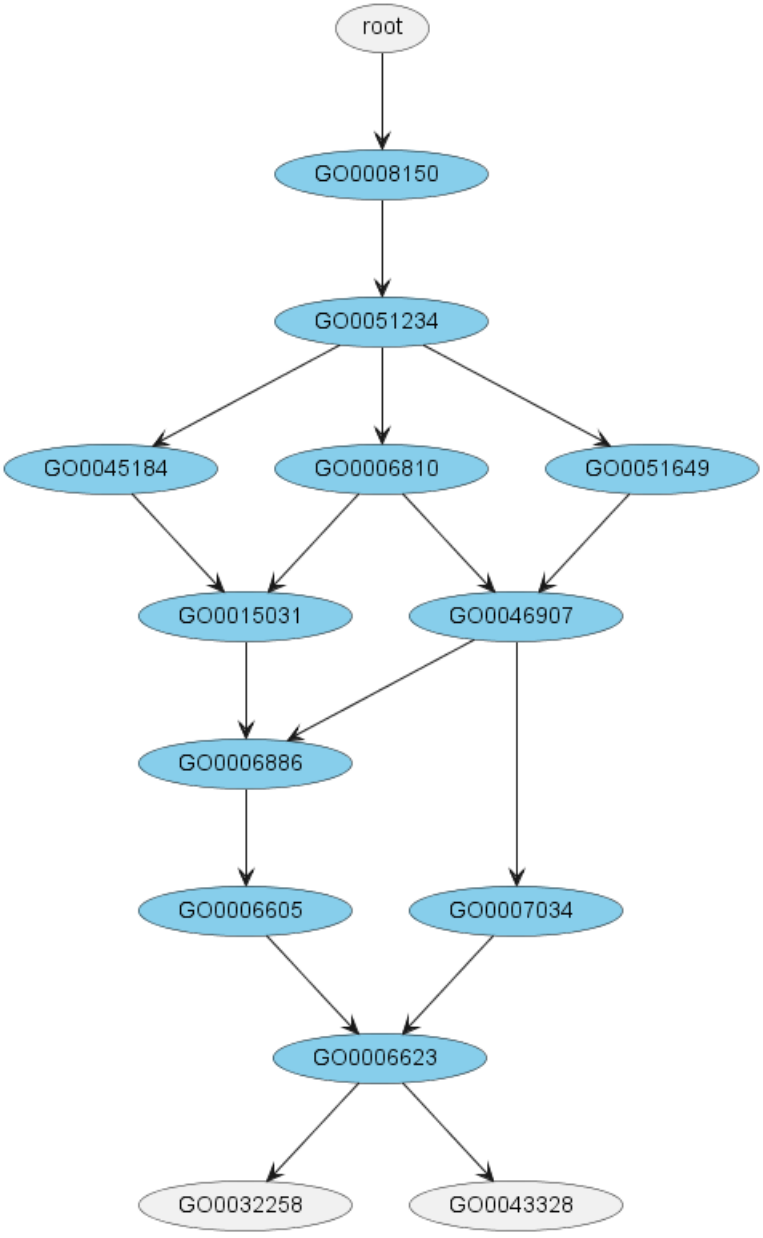


Figura 21 – GO0006623 - Exemplo verdadeiro de resposta

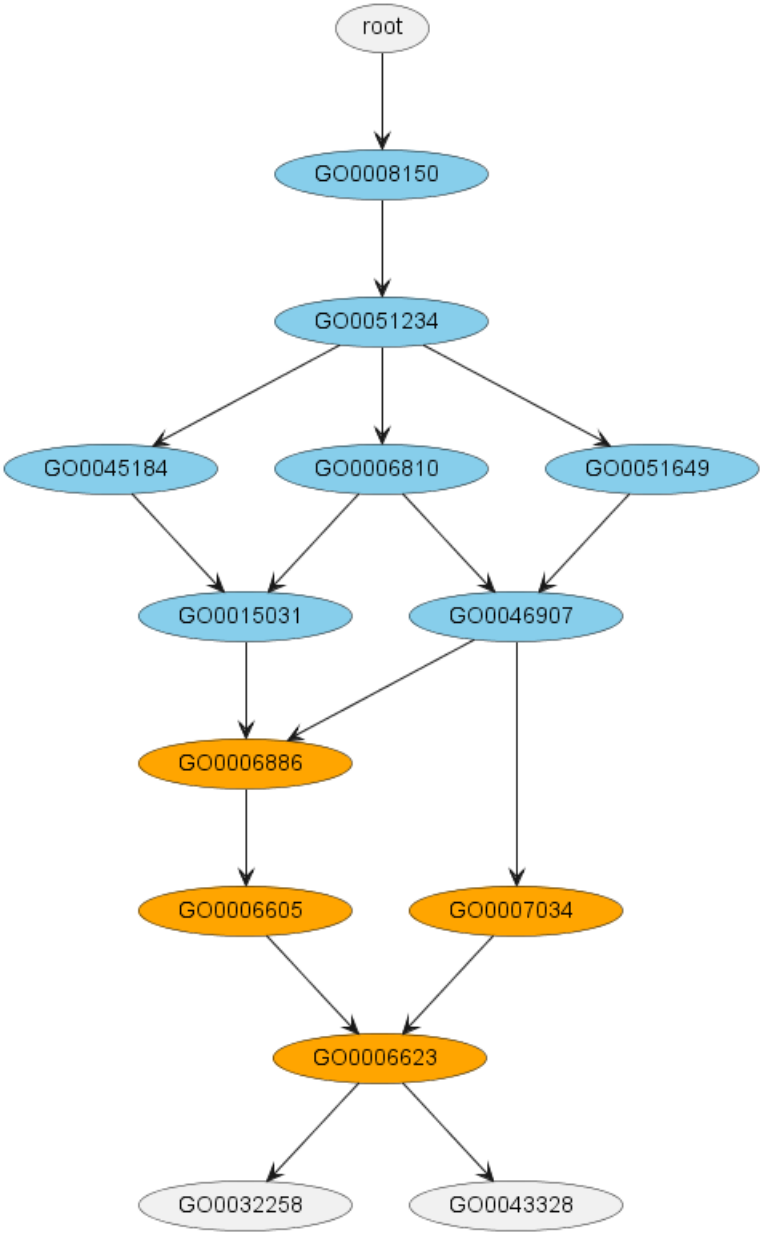


Figura 22 – GO0006623 - Exemplo 1 de resposta

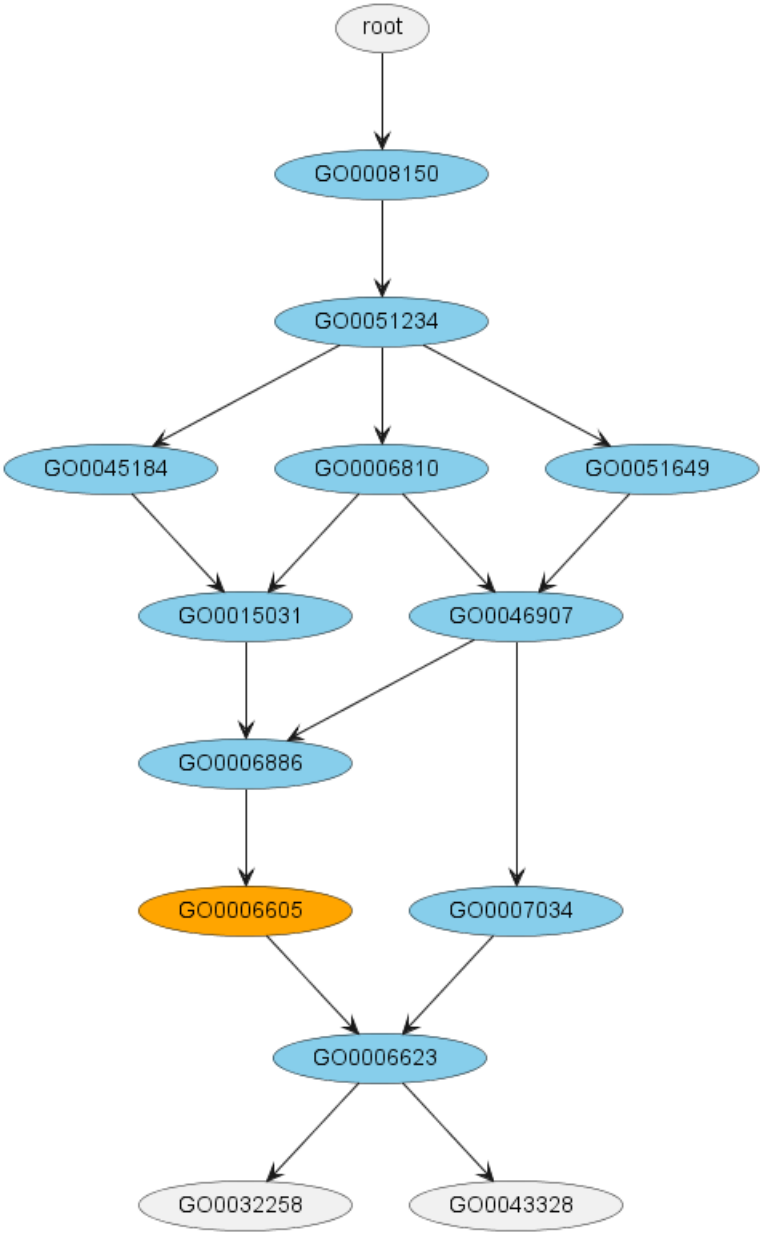


Figura 23 – GO0006623 - Exemplo 2 de resposta

4.5 Avaliação

Os resultados foram avaliados seguindo os seguintes passos:

- Análise do comportamento das métricas hierárquicas nos conjuntos de base de dados que utilizaram a política *less inclusive* e que utilizaram a combinação *less inclusive* com SMOTE.
- Avaliação estatística das métricas hierárquicas.
  - Para a avaliação estatística de pares foi utilizado o teste Wilcoxon. Onde os pares são os resultados das métricas hierárquicas nos conjuntos de dados que utilizaram apenas a política *less inclusive* e combinação *less inclusive* com SMOTE.
  - Para avaliação da comparação dos resultados das métricas hierárquicas deste trabalho com o estado da arte das bases de dados sintéticas foi utilizado o teste de Friedman, e caso a hipótese nula fosse rejeitada a utilização do teste de Nemenyi.

4.5.1 Métricas

A etapa de avaliação aplicou as métricas escolhidas e em seguida comparou os resultados do conjunto de bases sintéticas e GO. O resultado esperado é uma Tabela contendo todas as métricas hierárquicas, comparando os conjuntos de bases utilizando apenas a política de treinamento *less inclusive* e os resultados utilizando a combinação *less inclusive* e SMOTE.

De acordo com Silla and Freitas (2011) as melhores métricas para serem utilizadas em problemas de classificação hierárquica são chamadas de métricas hierárquicas e foram propostas por Kiritchenko et al. (2005). Estas métricas são extensões das métricas já conhecidas de precisão, revocação e *f-measure*. Elas foram desenvolvidas para que a hierarquia das classes seja levada em consideração durante o cálculo. As métricas hierárquicas como a precisão hierárquica, o *recall* hierárquico e a *f-measure* hierárquico, foram apresentadas na Seção 2.2.

Apesar da análise de comparação entre métricas hierárquicas dos conjuntos de bases o *f-measure* que será a base para analisar o objetivo deste trabalho. Quando observamos a métrica de precisão hierárquica ela demonstra do total das previsões quantas estavam corretas. Já a métrica de revocação hierárquica nos apresenta o quanto das classes o modelo acertou. Quando trabalhamos com desbalanceamento de classes não é ideal observar apenas a precisão ou apenas a revocação, é preciso observar a média harmônica entre elas. Esta média harmônica é caracterizada pelo *f-measure*.

As métricas serão demonstradas usando os exemplos das Figuras 21, 22 e 23. Primeiramente, observe atentamente o exemplo na Figura 22.

As métricas hierárquicas utilizam o conjunto de classes verdadeiras (*True classes*) e o conjunto de classes preditas (*Predicted classes*). Os exemplos verdadeiros são definidos pela

classe-alvo e seus ancestrais, neste caso, sendo as classes GO00008150, GO0051234, GO0045184, GO00006810, GO0051649, GO0015031, GO0046907, GO0006886, GO0006605, GO0007034 e GO0006623.

O conjunto predito consiste nas classes que foram previstas como positivas e todos os seus ancestrais. Neste exemplo, as classes preditas como positivas são GO00008150, GO0051234, GO0045184, GO00006810, GO0051649, GO0015031 e GO0046907.

Ao comparar o conjunto de classes verdadeiras (apresentado na Figura 21) com o conjunto de classes preditas (apresentado na Figura 22), podemos ver que sete classes foram corretamente preditas. As classes corretamente preditas estão destacadas em azul, enquanto as classes que não foram preditas corretamente estão em laranja.

É importante observar que este é um exemplo de predição de nó folha não obrigatória (*Non-Mandatory Leaf Node Prediction*). Portanto, as classes GO0032258 e GO0043328 não fazem parte do conjunto de classes verdadeiras (aquelas com fundo branco).

Para o exemplo 1:

A Equação 4.1 apresenta a precisão hierárquica (hP). A Equação 4.2 apresenta o *recall* hierárquico (hR). A Equação 4.3 apresenta a *f-measure* hierárquica (hF).

$$hP = \frac{7}{7} = 1 \tag{4.1}$$

$$hR = \frac{7}{11} = 0.63 \tag{4.2}$$

$$hF = \frac{2 * 1 * 0.63}{1 + 0.63} = 0.77 \tag{4.3}$$

Para outro exemplo de predição, assumamos que o conjunto de classes verdadeiras seja o apresentado na Figura 21 e que o conjunto de classes preditas seja o apresentado na Figura 23.

Com esses conjuntos, é possível calcular as métricas hierárquicas. As Equações 4.4, 4.5 e 4.6 mostram, respectivamente, a precisão hierárquica (hP), o *recall* hierárquico (hR) e a *f-measure* hierárquica (hF).

$$hP = \frac{10}{10} = 1 \tag{4.4}$$

$$hR = \frac{10}{11} = 0.90 \tag{4.5}$$

$$hF = \frac{2 * 1 * 0.9}{1 + 0.9} = 0.95 \tag{4.6}$$

4.5.2 Análise Estatística

Para realizar os teste das hipóteses declaradas no seção 1.2 foram utilizados teste de hipóteses estatísticos. Quando falamos em testar duas amostras dependentes, onde vamos analisar os resultados antes ou depois de um tratamento, podemos utilizar o teste de Wilcoxon desenvolvido por Woolson (2005) para este fim. Para analisar as hipóteses: "O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados da aplicação do classificador local por nó *less inclusive* e "O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados nas bases GO, pode superar os resultados da aplicação do classificador local por nó *less inclusive* vamos utilizar o este teste.

O teste pareado de Wilcoxon é um teste não paramétrico utilizado para comparar duas amostras pareadas. Nele a hipótese nula (H_0) é sinalizada pela não diferença entre os pares onde a distribuição deste pares de dados são simétricas em torno da mediana zero. Já na hipótese alternativa (H_A) existe uma diferença entre os pares de dados.

O teste pareado de Wilcoxon acontece em alguns passos, e são eles:

- Calcular a diferença entre os pares, conservando o sinal.
- Ranquear os valores de diferença ignorando o sinal da diferença.
- Somar os valores positivos (W^+) e os valores negativos (W^-).
- Determinar a estatística do teste W (Wilcoxon), que é calculada pelo menor valor das somas dos valores positivos e negativos.

$$W = min(W^+, W^-) \tag{4.7}$$

- Determinar o p-valor. Para amostras pequenas ($n \leq 25$), onde n é o número de amostras, o p-valor é obtido a partir de Tabelas da distribuição de Wilcoxon. Para amostras grande($n > 25$), podemos utilizar a distribuição normal utilizando a fórmula abaixo, onde CE = correção a ser usada se houver empates (se não houver empates $CE = 0$), onde $CE = \sum(t^3 - t)$ sendo t é o número de empates por ranque.

$$Z = \frac{W - \frac{n(n+1)}{4}}{\sqrt{\frac{n(n+1)(2n+1) - \frac{CE}{2}}{24}}} \tag{4.8}$$

O p-valor é calculado abaixo sendo P a probabilidade e Z distribuição normal padrão.

$$p = 2P(Z > |Z_{\text{calculado}}|) \tag{4.9}$$

- Verificar se o p-valor é menor que 0,05. Caso ele seja a hipótese nula H_0 é rejeitada. E caso o p-valor for maior que 0,05 não existe evidência suficiente para rejeitar a hipótese nula H_0.

Para realizar o teste de hipótese "O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados do estado da arte" foi utilizado o teste de Friedman (Friedman, 1940) e em seguida pelo teste de Nemenyi (Nemenyi, 1963).

O teste de Friedman é um teste estatístico não paramétrico utilizado para detectar diferenças em medições repetidas em três ou mais grupos. A hipótese nula (H_0) é representada pela não diferença estatística dos grupos, ou seja, suas medianas populacionais são iguais. Para a hipótese alternativa (H_A) existe pelo menos um grupo cuja distribuição é diferente dos demais, ou seja um dos grupos, pelo menos, possui uma mediana estatisticamente diferente de outro. O teste é apresentado nos passos a seguir.

- Organizar os dados em matriz onde cada linha representa um grupo e cada coluna uma condição deste grupo.
- Ranquear as condições de cada linha (grupo), e os valores são convertidos em números dos ranques (exemplo: 1ª, 2ª etc.).
- Somar os ranques de cada linha (grupo).
- Calcular o teste de estatística de Friedman onde n é o número de grupos, k é o número de condições, R_j é a soma dos ranques para cada condição.

$$Q = \frac{12}{nk(k+1)} \sum R_j^2 - 3n(k+1) \tag{4.10}$$

- Calcular o p-valor onde comparamos o teste de estatística de Friedman (Q) com a distribuição qui-quadrado (X^2) com $k-1$ graus de liberdade. Assim o cálculo do p-valor é definido pela função.

$$p = P\left(\chi^2_{df} > Q\right) \tag{4.11}$$

- Verificar se o p-valor é menor que 0,05. Caso ele seja a hipótese nula H_0 é rejeitada. E caso o p-valor for maior que 0,05 não existe evidência suficiente para rejeitar a hipótese nula H_0.

O teste de Nemenyi pode ser realizado caso a hipótese nula do teste de Friedman seja rejeitada, caso contrário não existe diferença significativa entre os grupos apresentados. O teste de Nemnyi compara pares de grupos utilizando a diferença entre seus ranques médios, e depois avaliando se a diferença é maior que um valor crítico. O teste é apresentado nos passos a seguir:

- Organizar os dados em matriz onde cada linha representa um grupo e cada coluna uma condição deste grupo.
- Ranquear as condições de cada linha (grupo), e os valores são convertidos em números dos ranques (exemplo:1ª,2ª etc.). Onde o ranque é iniciado do melhor resultado sendo o primeiro lugar, depois vem o segundo lugar e assim por diante.
- Calcular a média dos ranques para cada linha (grupo). Esta média normalmente é chamada de ranque médio do grupo.
- Calcular o valor crítico, onde q_α é o valor crítico da distribuição de Tukey retirado na Tabela de Nemenyi, k é o número de grupos e n é o número de condições por grupo.

$$CD = q_\alpha \cdot \sqrt{\frac{6n}{k(k+1)}} \tag{4.12}$$

- Calcular a diferença dos ranques médios de pares de grupos, caso essa diferença for maior que o valor crítico para um dos pares, então existe uma diferença significativa entre esses grupos.

5 Resultados

Os resultados dos conjuntos de dados sintéticos e do *Gene Ontology* são apresentados neste capítulo. A Seção 5.1.2 apresenta a comparação entre os resultados sem SMOTE (*baseline*) e os resultado com a aplicação do métodos SMOTE para os conjuntos de dados sintéticos. A Seção 5.1.1 mostra a diferença entre os resultados sem o SMOTE, os resultados utilizando o método SMOTE e o estado da arte apresentado no trabalho de Serrano-Pérez and Sucar (2021) para os conjuntos de bases de dados sintéticos desbalanceados. A Seção 5.2 apresenta a comparação entre a os resultados *baseline* e os resultado com a aplicação do métodos SMOTE para os conjuntos de bases de dados do GO.

5.1 Bases Sintéticas

5.1.1 *Resampling*

Esta seção analisa o objetivo específico "Comparar os resultados das métricas hierárquicas entre experimentos utilizando apenas o classificadores local por nó (LCN) *less inclusive* e a combinação do *less inclusive* com o método de *resampling* SMOTE em bases sintéticas desbalanceadas". Os resultados dos experimentos com os conjuntos de dados *easy* e desbalanceados são apresentados na Tabela 7. Nas colunas H *F-measure*, é possível observar um *ranking* dos conjuntos de bases de dados. Os melhores resultados são de conjuntos de bases de dados com seis atributos, cinco nós e duas profundidades máximas. À medida que o número de atributos aumenta, os resultados diminuem.

Datasets	**H Precision**		**H Recall**		**H F-Measure**	
	Baseline	**Smote**	***Baseline***	**Smote**	***Baseline***	**Smote**
DSE2	82,16	81,78	87,50	92,00	84,75(3)	**86,59(3)**
DSE4	77,22	75,10	84,33	87,56	80,62(4)	**80,85(4)**
DSE6	67,16	62,19	68,48	75,43	67,81(5)	**68,17(5)**
DSE8	64,72	55,34	63,31	73,57	**64,01(6)**	63,17(6)
DSE10	99,49	99,50	98,50	99,00	98,99(1)	**99,25(1)**
DSE12	90,79	90,48	95,39	96,31	93,03(2)	**93,30(2)**

Tabela 7 – Resultados de Precisão Hierárquica, *Recall* Hierárquica e *F-measure* Hierárquico para os conjuntos de dados *easy* desbalanceados com estrutura hierárquica de DAG.

Quando observamos a diferença das métricas hierárquicas de precisão (HP) da Tabela 7 foi possível observar que a aplicação do método SMOTE diminuiu o HP. Em alguns casos como DSE2, DS10 e DS12 não chegou a 1% esta diminuição, porém em casos como o DSE8

a diminuição do HP chegou a 9,38%. Assim levando a concluir que o SMOTE diminuiu o HP destas bases.

Agora quando observamos a diferença das métricas hierárquicas de *recall* (HP) na Tabela 7 pode ser visto um aumento dos resultados após a aplicação do método SMOTE. Para as bases DSE10 e DSE12 o aumento não chegou a 1%, porém quando observamos as outras bases o aumento foi maior. O aumento para as bases DSE2 e DSE4 ficou entre 1% e 5%, enquanto o aumento para as bases DSE6 E DSE8 ficou entre 5% e 11%.

Quando observamos a diferença entre as métricas hierárquica *F-measure* (HF) é possível observar que 83% das bases possuem um aumento do HF. A única base que não possui aumento do HF é a DS8 onde sua diminuição do HP foi próxima de 10% e seu aumento do HR foi próximo de 11%. Podemos perceber que o aumento do HR foi compensado pela diminuição da precisão, causando este não aumento no HF.

Os resultados foram analisados estatisticamente utilizando o teste de Wilcoxon, onde ele se aplica para dados que não estão distribuídos normalmente. A hipótese nula sendo o p-valor >= 0,05, onde não existe diferença significativa entre as amostras e a hipótese alternativa onde p-valor < 0,05, onde existe uma diferença significativa entre as amostras. A Tabela 8 apresenta os resultados de hipótese.

É possível observar que na Tabela 8 que para o HP o p-valor não é menor que 0,05 assim não existem evidências estatísticas para rejeitar a hipótese nula, ou seja não existe diferença significativa entre os valores com e sem SMOTE. Para o HR, como o p-valor é menor que $0,05$, assim rejeitamos a hipótese nula, ou seja existe uma diferença estatística significativa entre os valores com e sem SMOTE. E por fim para o HF é possível verificar que a hipótese nula não é rejeitada, assim não existe uma diferença entre os valores com e sem SMOTE.

Métrica	**Teste Usado**	**p-valor**	**Significativo (<0,05)**
Hierarchical Precision	Teste de Wilcoxon	0,06250	Verdadeiro
Hierarchical Recall	Teste de Wilcoxon	0,03125	Verdadeiro
Hierarchical F-measure	Teste de Wilcoxon	0,31250	Falso

Tabela 8 – Resultados do Teste de Wilcoxon para as Métricas Hierárquicas dos conjuntos de dados. *easy*

A Tabela 9 apresenta os resultados dos experimentos com os conjuntos de dados *hard* e desbalanceados. Nas colunas H *F-measure*, é possível identificar um *ranking* de desempenho para os conjuntos de bases de dados. Observa-se que, à medida que a profundidade máxima aumenta, os resultados dos conjuntos de bases de dados diminuem. O conjunto DSH2, com profundidade máxima de dois, ocupa a primeira posição no ranking utilizando SMOTE, enquanto o conjunto DSH12, com profundidade máxima de cinco, ocupa a última posição no *ranking*.

Na Tabela 9, assim como foi visto na Tabela 7, o HP das bases que utilizam SMOTE diminuem em relação aos resultados que não utilizam SMOTE. As bases DSH2, DSH4, DSH6 e DSH8 possuem uma diminuição de até 2%. Observando as bases DSH10 e DSH12, elas

Datasets	**H Precision**		**H Recall**		**H F-measure**	
	Baseline	**Smote**	***Baseline***	**Smote**	***Baseline***	**Smote**
DSH2	32,26	31,13	77,26	85,24	45,51(1)	**45,61(1)**
DSH4	31,21	29,39	73,47	85,36	**43,81(2)**	43,72(2)
DSH6	28,89	28,58	78,44	78,54	**42,23(3)**	41,91(3)
DSH8	24,81	24,29	80,70	81,03	**37,95(4)**	37,38(5)
DSH10	56,67	28,98	27,48	72,56	37,01(5)	**41,42(4)**
DSH12	56,51	24,79	23,07	70,76	32,76(6)	**36,72(6)**

Tabela 9 – Resultados de Precisão Hierárquica, *Recall* Hierárquica e F-measure Hierárquico para os conjuntos de dados *hard* desbalanceados com estrutura hierárquica de DAG.

apresentam uma diminuição mais expressiva de 27,70% e 31,72% respectivamente.

Quando observamos a diferença entre os resultados dO HR da Tabela 9, assim como foi visto na Tabela 7, existe um aumento. As bases DSH2, DSH4, DSH6 e DSH8 possuem um aumento de até 12%. Agora as bases DSH10 e DSH12 apresentam um aumento mais expressivo de 45,5% e 47,69% respectivamente.

O HF na Tabela 9 quando comparada seus resultados apresenta metade das bases com aumento utilizando o SMOTE. Os resultados do HF traduzem que o aumento do HR e pode ter compensado a diminuição do HP resultando em um número menor de aumento do HF, mesmo em bases como DSH10 e DSH12 onde o aumento do HR foi de 45,5% e 47,69% respectivamente.

Os resultados foram analisados estatisticamente utilizando o teste de Wilcoxon, a Tabela 10 apresenta os resultados de hipótese. Os valores da Tabela 10 para HF são semelhantes aos resultados da Tabela 8, onde o SMOTE não causou um impacto estatisticamente significativo no HF. Essa observação sugere que o HF não apresentou alterações significativas, o que pode indicar que os ganhos no HR foram compensados por perdas na HP.

Métrica	**Teste Usado**	**p-valor**	**Significativo (<0,05)**
Hierarchical Precision	Teste de Wilcoxon	0,03125	Verdadeiro
Hierarchical Recall	Teste de Wilcoxon	0,03125	Verdadeiro
Hierarchical F-measure	Teste de Wilcoxon	0,68750	Falso

Tabela 10 – Resultados do Teste de Wilcoxon para as Métricas Hierárquicas dos conjuntos de dados *hard*

Por fim a hipótese "O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados da aplicação do classificador local por nó *less inclusive*."não se tornou verdadeira. Quando observamos o teste de Wilcoxon para o HR nas Tabelas 10 e 8 pode-se observar que a diferença entre os pares é estatisticamente significativa. Porém, quando observamos o teste para o HF nestas mesmas tabelas observamos que a hipótese nula para HF não foi rejeitada. Assim, apesar de existir uma diferença entre os resultados da aplicação do método SMOTE e sem, eles não são estatisticamente significantes.

Neste trabalho foi utilizado apenas um método de *resampling*, o SMOTE. Esta decisão pode ter afetado os resultados, onde não foi possível comparar outros métodos de *resampling* que poderiam aumentar as métricas hierárquicas a ponto de se tornarem estatisticamente diferentes do método sem *resampling*. Uma sugestão seria aplicar outros métodos de *resampling* conhecidos no estados da arte e comparar com os resultados encontrados neste trabalho.

5.1.2 Discussão em Relação ao Estado da Arte

Este trabalho realizou experimentos com os conjuntos de bases de dados do trabalho de Serrano-Pérez and Sucar (2021). O trabalho apresenta os resultados dos seguintes métodos: *flat*, *top-down*, HBA (proposto por Barutcuoglu et al. (2008) e posteriormente adaptado por Serrano-Pérez and Sucar (2019)), CLUS-HMC (proposto por Vens et al. (2008)), CPE (proposto por Ramírez-Corona, Sucar and Morales (2016)), nLLCPN e LCPNB (propostos por Nakano et al. (2017)) e Redes Bayesianas com Classificadores Encadeados (propostas por Serrano-Pérez and Sucar (2019)).

Nos experimentos de Serrano-Pérez and Sucar (2021), os conjuntos de dados foram divididos em treinamento e teste, conforme mostrado nas Tabelas 3 e 4; este trabalho segue o mesmo protocolo de divisão. Isso possibilitou comparar o estado da arte com os resultados obtidos neste estudo. Assim satisfazendo os requisitos para a análise do objetivo "Comparar os resultados de *f-measure* dos estados da arte das bases sintéticas desbalanceadas e os experimentos que combinam a política *less inclusive* com o método de *resampling* SMOTE em bases sintéticas desbalanceadas".

Nas Tabelas 11 e 12 é possível observar a comparação dos conjuntos de base *easy* e *hard*, dos conjuntos desbalanceados configurados em taxonomia DAG com os resultados da linha de base (LCNB) e do SMOTE (LCNS). As análises estatísticas dos resultados foram baseadas no Friedman test (FT) e em seguida por Nemenyi test (NT).

DS	**FLAT**	**TD**	**CPE**	**C–H**	**HBA**	**HCP**	**HCA**	**HCC**	**LCNB**	**LCNS**
DSE2	87,09	83,87	73,02	78,49	84,38	85,12	81,79	**88,37**	84,75	86,59
DSE4	**86,9**	77,25	66,02	73,66	79,37	80,91	79,82	84,00	80,62	80,85
DSE6	68,14	65,37		68,14	**73,12**	69	71,34	66,80	67,81	68,17
DSE8	64,54	63,27		59,87	65,18	**66,73**	44,93	61,33	64,01	63,17
DSE10	99,25	94,74	80,85	94,15	**100**	**100**	**100**	99,25	98,99	99,25
DSE12	**94,55**	90,49	84,09	89,1	**94,55**	93,22	93,65	**94,55**	93,03	93,30

Tabela 11 – Resultados para a *f-measure* hierárquica (hF) de diferentes classificadores sobre conjuntos de dados *easy* desbalanceados com estrutura hierárquica de DAG. LCNB e LCNS são os resultados dos experimentos deste trabalho e estão destacados. O melhor resultado para cada conjunto de dados está em negrito. A implementação fornecida do CPE não funcionou em alguns conjuntos de dados. (DS: Bases de Dados; C–H: CLUS-HMC., LCNB: *Local Classifier per node less exclusive Baseline*, LCNS: *Local Classifier per node less exclusive SMOTE*).

DS	**FLAT**	**TD**	**CPE**	**C–H**	**HBA**	**HCP**	**HCA**	**HCC**	**LCNB**	**LCNS**
DSH2	54,76	53,71	57,14	56,08	60,6	**61,79**	60,83	45,24	45,51	45,61
DSH4	53,76	50,16	**56,26**	54,66	55,83	47,35	56,04	56,15	43,81	43,72
DSH6	58,67	41,43	25,31	54,77	56,4	**59,44**	39,27	37,92	42,23	41,91
DSH8	47,98	41,51	25,77	47,32	51,71	50,42	**55,02**	39,25	37,95	37,38
DSH10	44,81	36,31	20,81	45,84	**50,06**	45,36	38,53	40,22	37,01	41,42
DSH12	43,4	34,18		40,55	**45,42**	36,7	35,7	32,73	32,76	36,72

Tabela 12 – Resultados para a *f-measure* hierárquica (hF) de diferentes classificadores sobre conjuntos de dados *hard* desbalanceados com estrutura hierárquica de DAG. LCNB e LCNS são os resultados dos experimentos deste trabalho e estão destacados. O melhor resultado para cada conjunto de dados está em negrito. A implementação fornecida do CPE não funcionou em alguns conjuntos de dados. (DS: Bases de Dados; C–H: CLUS-HMC., LCNB: *Local Classifier per node less exclusive Baseline*, LCNS: *Local Classifier per node less exclusive SMOTE*).

Para análise da hipótese "O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados do método *flat* do estado da arte" esta seção foca na diferença entre os resultados dos métodos apresentados (LCNB e LCNS) com o resultado do método *FLAT* apresentado por Serrano-Pérez and Sucar (2021).

Observando os resultados da Tabela 11 que os resultados dos métodos LCNB e LCNS das bases DSE2, DSE4, DSE8 e DSE12 não superaram os valores de HF do método *FLAT*. Apesar destas bases não superarem os valores do método *FLAT* eles não são os métodos com o pior resultado, outros métodos apresentam resultados menores que os dos Métodos LCNB e LCNS. Para a base DSH6 o método LCNS superou em resultado o método *FLAT*. E para a base DSE10 o método LCNS se equiparou aos resultados do método *FLAT*.

Na Tabela 12 é possível observar que os resultados dos método LCNS e LCNB não superam os resultados do método *FLAT*. A base DSH4 apresenta o pior resultado dos métodos. O restante das bases não possuem os piores resultados nem os melhores resultados.

Para uma análise mais aprofundada desta diferença entre os métodos foi aplicado o FT e em seguida do NT. Para as Tabela 11 o teste de hipótese nula FT foi rejeitada sendo o p-valor igual a 0,0033. Indicando que existe uma diferença entre os diferentes métodos apresentados. Na Figura 24 é possível observar o *rank* médio dos resultados do trabalho 11 junto ao resultados deste trabalho, sendo eles os resultados de *Baseline* sem SMOTE (LCNB) e com SMOTE (LCNS). Como o sistema de *rank* LCNS teve um desempenho de 4,5 e LCNB de 5,83. O método *FLAT* apresentar o melhor desempenho observando o ranque médio de todos os métodos. O valor *critical difference* (CD) é de 7,8208. Este valor indica que nenhum método apresentou diferença estatística significativa em relação às outras.

Na Tabela 12 o teste de hipótese nula FT foi rejeitada sendo o p-valor igual a 0,0019. Indicando que existe uma diferença entre os diferentes métodos apresentados. Na Figura 25 é possível observar o ranque médio dos resultado do trabalho trabalho de Serrano-Pérez and Sucar (2021) que podem ser observados na Figura 12 junto ao resultados deste trabalho, sendo eles os resultados de *Baseline* sem SMOTE (LCNB) e com SMOTE (LCNS). O LCNS apresentou o desempenho de 7,0 e LCNB de 7,83. Ficando os dois abaixo do método *FLAT*. O valor *critical difference* (CD) é de 7,8208. Este valor indica que nenhum método apresentou diferença estatística significativa em relação às outras.

Assim a hipótese "O uso do método de *resampling* SMOTE, juntamente com um classificador local por nó *less inclusive* em problemas estruturados em DAG aplicados em bases artificiais desbalanceadas, podem superar os resultados do método *flat* do estado da arte" não foi validada. Uma vez que para as bases DSE, tanto para as bases DSH, a diferença entre os ranques médios não superou o valor *critical difference*, por fim não podendo inferir que os métodos utilizados neste trabalho foram melhores ou piores que os utilizados no trabalho de Serrano-Pérez and Sucar (2021).

Na seção anterior foi comentado que a utilização de outros métodos de *resampling* podem

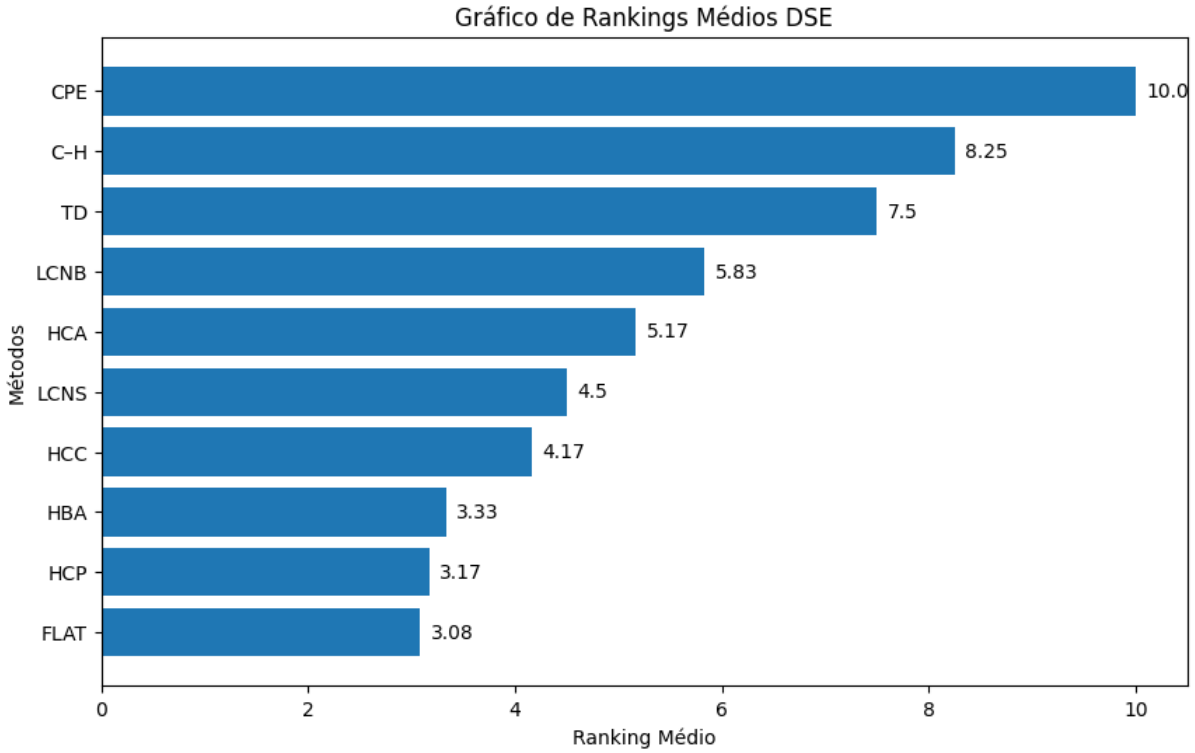


Figura 24 – Rank Médio Métodos DSE.

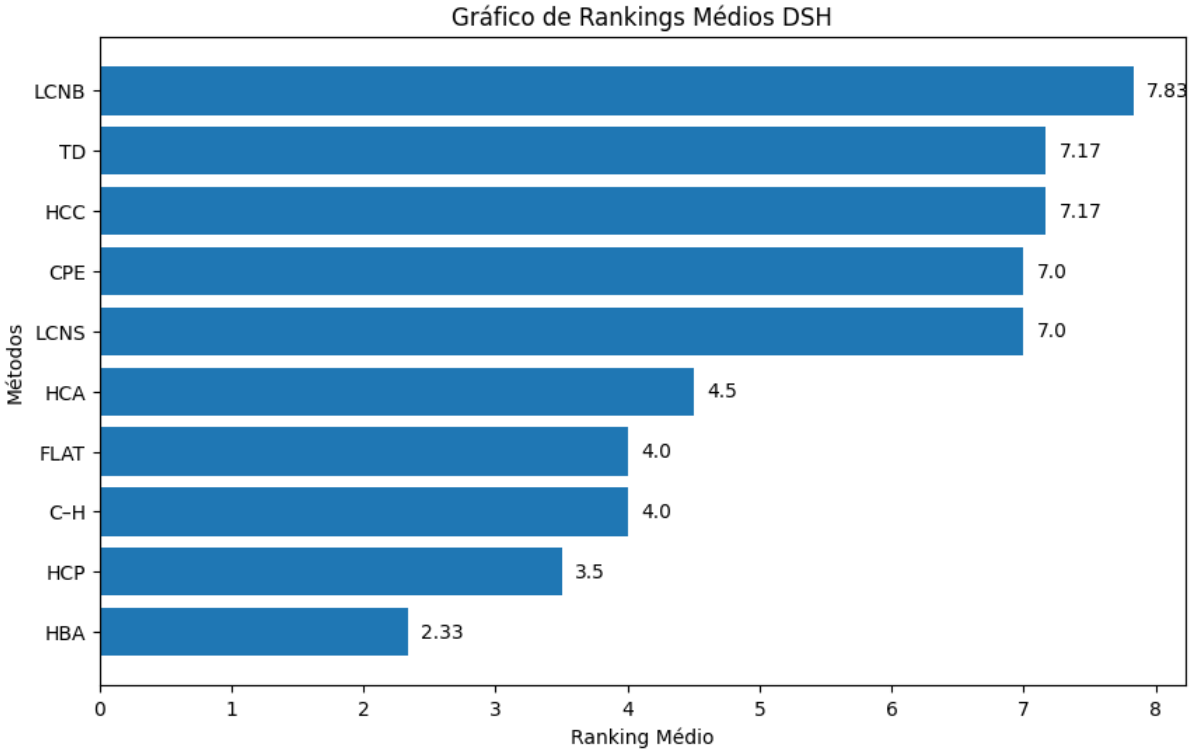


Figura 25 – Rank Médio Métodos DSH.

agregar ao trabalho desenvolvido. Além deste ponto podemos dizer que a escolha do classificador, Naive Bayes, pode ter afetado o desempenho dos resultados. Uma vez que adicionamos mais classes e profundidade observamos uma dificuldade do classificador de distinguir entre elas. Assim, sugere-se a aplicação de outros classificadores para análise de sua performance junto aos resultados deste trabalho.

5.2 *Gene Ontology*

Esta seção tem como objetivo realizar a analise do objetivo especifico "Comparar os resultados das métricas hierárquicas entre experimentos utilizando apenas o classificadores local por nó (LCN) *less inclusive* e a combinação do *less inclusive* com o método de *resampling* SMOTE em bases reais genéticas *Gene Ontology* (GO)".

Os conjuntos de dados do Gene Ontology (GO) propostos por Vens et al. (2008) demonstraram maior complexidade em comparação com as bases sintéticas apresentadas na última seção 5.1. Os conjuntos de dados do GO possuem, em média, 3997 classes e quatorze níveis de profundidade, conforme mostrado na Tabela 5.

A profundidade é mais que o dobro daquela encontrada nos conjuntos de dados sintéticos mais difíceis. Como observado na última seção 5.1, à medida que a profundidade e o número de nós aumentam, as métricas hierárquicas, especialmente a precisão hierárquica, tendem a diminuir. Portanto, este trabalho infere que os resultados das métricas hierárquicas dos conjuntos de dados do GO devem apresentar desempenho inferior às métricas com menos nós e menor profundidade. Os resultados podem ser encontrados na Tabela 13.

Datasets	**H Precision**		**H Recall**		**H F-measure**	
	Baseline	**Smote**	***Baseline***	**Smote**	***Baseline***	**Smote**
Church	2,23	3,16	40,48	47,53	4,23	**5,93**
Cellcyle	1,70	7,16	79,39	53,57	3,33	**12,62**
Derisi	4,95	3,23	29,76	42,47	**8,49**	6,00
Eisen	9,23	9,71	47,33	44,53	15,45	**15,95**
Expr	4,83	5,71	47,23	46,87	8,76	**10,18**
Gasch 1	6,41	7,07	51,93	48,49	11,42	**12,34**
Gasch 2	5,97	6,30	48,00	47,46	10,61	**11,12**
Hom	8,45	8,45	39,58	39,58	**13,93**	**13,93**
Pheno	3,52	3,27	46,35	42,12	**6,55**	6,07
Seq	4,98	3,10	35,10	49,87	**8,72**	5,84
Spo	3,37	3,61	39,48	43,35	6,21	**6,67**
Struc	3,43	2,31	13,20	54,22	**5,45**	4,43

Tabela 13 – Resultados de Precisão Hierárquica, “Recall” Hierárquica e F-measure Hierárquico para as bases GO.

Observando a Tabela 13 é possível observar que os resultados HP não são altos, porém algumas bases aumentam o HP quando aplicado o método SMOTE como por exemplo Cellcyle,

Eisen, Expr, Gasch1, Gasch2 e Spo. Outras bases foi possível observar a diminuição do HP como no caso da base Derisi, Pheno, Seq e Struc. Já a base Hom não apresentou diferenciação entre os pares com e sem SMOTE.

Quando analisamos os resultados da Tabela 13 é possível observar que metade das bases de dados possuem uma diminuição de HR, onde a base Cellcyle possui uma diminuição de 25,82 pontos entre o resultado baseline e com SMOTE. As bases Church, Derisi, Seq, Spo e Struc obtiveram um aumento de HR, onde a base Struc apresentou a maior diferença entre baseline e Smote de 41,02. Já a base Hom não apresentou diferenciação entre os pares com e sem SMOTE. Podendo sinalizar a dificuldade do classificador Naive Bayes em lidar com grandes quantidade de atributos.

Analisando o HF na Tabela 13 é possível perceber que as bases que sinalizaram uma diminuição de resultados no HP também apresentaram uma diminuição no HF, mesmo algumas possuindo um aumento em HR como as bases Derisi, Seq e Struc. Sinalizando que o aumento de HR não foi suficiente para ajustar os baixos valores de HP. Em casos onde houve o aumento de HP também ocorreu um aumento em HF mesmo com uma diminuição em HR, como nas bases Eisen, Expr, Gasch1 e Gasch2.

É possível perceber que para a base Hom seus resultados não apresentaram alterações devido ao SMOTE, este fato pode estar associado a escolha do classificador Naive Bayes. Por ser um classificador mais simples e rápido ele pode não ter aprendido adequadamente em uma base com muitos atributos. O classificador pode ter afetado o HP no geral onde ele não conseguiu prever corretamente as classes. A sugestão deste trabalho é que seja aplicado outros classificadores para comparação e verificação se esta hipótese é válida.

No geral o SMOTE apresentou melhores resultados para algumas bases e outras ele não obteve uma boa resposta. Neste ponto a escolha de utilizar apenas um método de *resampling* foi limitadora, uma vez que não se sabe ao certo se a resposta se deve ao classificador, ou ao método de *resampling*. Assim, este trabalho sugere a combinação de teste de novos classificadores com os métodos de *resampling* do estados da arte utilizando método de *undersampling*, *oversampling* e híbrido.

Para análise estatística foi utilizado o teste de Wilcoxon. Na Tabela 14 é possível encontrar os resultados deste teste. O teste de hipótese nula não foi rejeitado, ou seja o p-valor foi acima de 0,05 em todas as métricas. Isso significa que não existem evidências estatísticas para confirmar que existe diferença entre os métodos *baseline* e SMOTE. Assim, os dois métodos produziram resultados semelhantes em termos de H *Precision*, H *Recall* e H *F-measure*. A estatística de Wilcoxon mede a diferença entre os pares de valores. Como os valores não são extremos, reforça a ideia de que não existe uma discrepância entre *baseline* e SMOTE. Com base no teste de Wilcoxon, podemos dizer que o uso do SMOTE não trouxe uma diferença estatisticamente significativa nas métricas quando comparado ao *baseline*.

Os resultados deste trabalho estão alinhados com os estudos relacionados ao tema de classificação hierárquica desbalanceada, que demonstram a complexidade em lidar com bases de

Métrica	Estatística	p-valor
H Precision	29.0	0.722108
H Recall	26.0	0.533695
H F-measure	27.0	0.593712

Tabela 14 – Resultados do Teste de Wilcoxon para bases de dados *Gene Ontology*

dados hierárquicos naturalmente desbalanceadas. O trabalho de Vens et al. (2008), por exemplo, explorou a classificação hierárquica multi-rótulo em bases genéticas organizadas em taxonomias de árvores e DAGs sem utilizar técnicas de reamostragem. Seus resultados demonstraram que, embora diferentes abordagens estruturais impactem o balanceamento das classes, nenhuma abordagem explorou diretamente o impacto de técnicas de reamostragem, como o SMOTE, no desempenho das métricas hierárquicas.

Por outro lado, pesquisas posteriores, como as de Chen and Hu (2012) e Chen, Duan and Hu (2012), propuseram adaptações do SMOTE para a classificação hierárquica, como o Hierarchical SMOTE, que realiza *oversampling* nas classes folhas e propaga as amostras sintéticas aos nós superiores da taxonomia. Esses estudos relataram melhorias no desempenho dos classificadores ao tratar o desbalanceamento de maneira estruturada. No entanto, os resultados da Tabela 13 e 14 sugerem que a aplicação do SMOTE pode não ser suficiente para garantir melhorias estatisticamente significativas nas bases analisadas, reforçando a necessidade de adaptações do algoritmo para classificação hierárquica.

O estudo de Pereira, Costa and Silla Jr (2021a) reforça essa necessidade. O trabalho investigou diferentes abordagens para tratar o desbalanceamento em problemas de classificação hierárquica e propôs três esquemas de *resampling* adaptados para abordagens locais de classificação: Classificadores Locais por Nó (LCN), Classificadores Locais por Nível (LCL) e Classificadores Locais por Nó Pai (LCPN). O estudo demonstrou que diferentes políticas de seleção de amostras impactam diretamente o desempenho dos classificadores e que técnicas como SMOTE+ENN e SMOTE+TL foram mais eficazes em reduzir o desbalanceamento e melhorar as métricas de classificação. Entretanto, os resultados indicam que a efetividade dos métodos de *resampling* está ligada às configurações do classificador local utilizado.

Outro aspecto relevante é a comparação com trabalhos que aplicam aprendizado profundo para classificação hierárquica. O estudo de Stiller et al. (2024), por exemplo, relatou dificuldades na melhoria da performance devido ao forte desbalanceamento dos dados e à complexidade do modelo hierárquico.

Dessa forma, os resultados apresentados nas Tabelas 13 e 14 reforçam a necessidade de considerar abordagens de *resampling* adaptadas para taxonomias hierárquicas. Embora o SMOTE seja amplamente utilizado, sua aplicação isolada pode não ser suficiente para lidar com o desbalanceamento em classificação hierárquica, sendo necessário explorar variantes e combinações com outras técnicas para obter melhores resultados.

Um dos principais desafios de utilizar o método SMOTE com políticas LCN em um

cenário de classificação hierárquica é o tempo necessário para realizar os experimentos. Conforme mostrado na Tabela 15, quanto mais instâncias e atributos um conjunto de dados possui, maior é o tempo necessário para realizar o experimento.

Bases de Dados	**|D|**	**|A|**	**Dias por experimentos**
Church	3764	27	4
Cellcyle	3766	77	4
Derisi	3733	63	4
Eisen	2425	79	4
Expr	3788	551	10
Gasch1	3773	173	4
Gasch 2	3788	52	4
Hom	3867	47034	20
Pheno	1592	69	4
Seq	3932	478	10
Spo	3711	80	4
Struc	3851	19628	20

Tabela 15 – Propriedades dos conjuntos de bases de dados da GO, onde |D| é o número de instâncias e |A| é o número de atributos.

Por exemplo, o conjunto de dados Church levou quatro dias para cada experimento. O conjunto de dados Struc levou vinte dias, como indicado na Tabela 15. É importante ressaltar que não houve diferença significativa no tempo necessário entre os experimentos *baseline* e os experimentos com SMOTE, o que destaca o investimento consistente de tempo exigido por este método. No total, foram realizados vinte e quatro experimentos em dois computadores. Com 16 GB de RAM, processador Intel(R) Core(TM) i5-9300H CPU com 2,4 Hz e sistema operacional Windows 64 bits. Embora o poder de processamento dos computadores utilizados pareça adequado para modelos simples, como o Naive Bayes, esse tipo de problema demandaria maior poder computacional se implementado em um cenário real.

6 Conclusão e Trabalhos Futuros

Este trabalho propôs avaliar se o uso do método de *resampling* de dados SMOTE poderia melhorar os resultados de classificação em problemas de classificação hierárquica estruturados em DAG, utilizando a abordagem de classificador local por nó com a politica de treinamento de exemplos positivos e negativos denominada *less inclusive*.

Nos conjuntos de dados sintéticos, os resultados demonstraram que o SMOTE pode aumentar o *recall hierárquico*, melhorando a identificação de classes minoritárias. No entanto, em alguns casos, essa melhora ocorreu à custa da precisão hierárquica, aumentando falsos positivos e impactando negativamente o desempenho do modelo. Como consequência, a métrica *F-measure* Hierárquica não apresentou ganhos estatisticamente significativos. Observou-se também uma correlação entre o número de classes e a profundidade das taxonomias com a diminuição das métricas hierárquicas. Esse fenômeno sugere que a estrutura do problema pode afetar significativamente a eficácia de métodos de *resampling*.Neste caso a hipótese de que o uso do método de *resampling* SMOTE, juntamente com a política *less inclusive* superaria os resultados da aplicação apenas da política foi rejeitada.

Quando foi observado os dados das bases sintética em análise com o estado da arte pode-se perceber que a hipótese de que o uso do método *resampling* combinado com a política *less inclusive* superaria os resultados do método flat do estado da arte foi rejeitada. Foi realizado o teste estatístico de Friedman e de Nemenyi, e verifico-se que a diferença entre os ranques médios não se mostrou maior que o valor *critical difference*. Assim a aplicação do SMOTE isoladamente pode não ser suficiente para melhorar o desempenho geral dos resultados das métricas hierárquicas.

Em bases reais como *Gene Ontology*, os resultados reforçaram a tendência de que quanto maior a profundidade e a quantidade de classes menor as métricas hierarquicas. Evidenciando que a aplicação do SMOTE aumentou o *recall*, mas não proporcionou melhorias estatisticamente significativas na métrica *F-measure* Hierárquica. Além disso, a dificuldade do classificador Naive Bayes em lidar com a alta dimensionalidade dos dados pode ter influenciado os resultados, sugerindo que a aplicação de outros classificadores neste problema de classificação. Neste caso a hipótese de que o uso do método de *resampling* SMOTE, juntamente com a política *less inclusive* superaria os resultados da aplicação apenas da política foi rejeitada.

A principal vantagem do uso do método SMOTE com o classificador local por nó em um cenário de classificação hierárquica é sua flexibilidade. Esse método permite a exploração de outras políticas não utilizadas neste trabalho, capacitando os pesquisadores a adaptar sua abordagem às necessidades específicas de seus conjuntos de dados.

Dentre as limitações deste trabalho, destaca-se a dificuldade em encontrar conjuntos de bases de dados públicos com taxonomia em DAG, o que restringiu a comparação com abordagens

mais diversificadas. Uma vez que encontramos conjuntos de bases de dados reais amplamente utilizados na academia, que são altamente complexos, tornou o processo experimental demorado, já que os experimentos não foram realizados de uma só vez. A Tabela 15 mostra que cada teste pode ser muito demorado.

Este trabalho possui algumas limitações. Algumas foram exploradas durante os resultados e outras foram apontadas nesta seção. Uma limitação não explorada é a utilização de apenas uma política do classificador local por nó. Esta limitação pode gerar uma oportunidade que seria aplicar outras políticas com o método SMOTE.

O uso do SMOTE como o único método de reamostragem aplicado também é uma limitação, e este trabalho sugere a exploração outros métodos de *resampling*, combinando SMOTE com técnicas como Borderline-SMOTE (Han; Wang; Mao, 2005), ADASYN (He et al., 2008) ou métodos de *undersampling*. E investigar abordagens híbridas que combinam *resampling* com técnicas de aprendizado profundo. E por fim Explorar a adaptação do *Hierarchical* SMOTE para melhor adequação aos desafios específicos da classificação hierárquica multi-rótulo.

Alguns critérios definidos e utilizados neste trabalho também possuem limitações. O critério de dados ausente nos conjuntos de dados *Gene Ontology* é uma limitação, e este trabalho sugere que outros critérios sejam aplicados.

A última, mas não menos importante, limitação e possibilidade de trabalho futuro é o critério de inconsistência apresentado na seção 4.4. Neste trabalho, utilizamos um método que considerou todos os pais de uma classe como positivos quando existe uma inconsistência. No entanto, este trabalho deixa espaço para estudar e implementar outros critérios conhecidos na literatura.

Em resumo, este estudo contribui para a compreensão do impacto do SMOTE na classificação hierárquica e destaca a necessidade de abordagens personalizadas para lidar com o desbalanceamento em contextos hierárquicos complexos. O avanço nessa área pode levar ao desenvolvimento de técnicas mais eficazes para aplicações práticas em biologia, medicina e outras áreas que lidam com dados hierárquicos desbalanceados.

Referências

Ariyaratne, H. B.; Zhang, D. A Novel Automatic Hierachical Approach to Music Genre Classification. In: *2012 IEEE International Conference on Multimedia and Expo Workshops*. 2012. p. 564–569.

Ashburner, M. et al. Gene Ontology: tool for the unification of biology. *Nature Genetics*, v. 25, n. 1, p. 25–29, 2000. ISSN 1546-1718. Available on: <https://doi.org/10.1038/75556>.

Barutcuoglu, Z.; DeCoro, C.; Schapire, R. E.; Troyanskaya, O. G. *Bayesian aggregation for hierarchical classification*. 2008.

Batista, G. E.; Prati, R. C.; Monard, M. C. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, ACM New York, NY, USA, v. 6, n. 1, p. 20–29, 2004.

Charte, F.; Rivera, A. J.; Jesus, M. J. del; Herrera, F. MLSMOTE: Approaching imbalanced multilabel learning through synthetic instance generation. *Knowledge-Based Systems*, Elsevier, v. 89, p. 385–397, 2015.

Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; Kegelmeyer, W. P. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, v. 16, p. 321–357, 2002.

Chen, B. Hierarchical Multi-Label Classification Based on Over-Sampling and Hierarchy Constraint for Gene Function Prediction. n. i, p. 183–189, 2012.

Chen, B.; Duan, L.; Hu, J. Composite kernel based SVM for hierarchical multi-label gene function classification. In: IEEE. *The 2012 International Joint Conference on Neural Networks (IJCNN)*. 2012. p. 1–6.

Chen, B.; Hu, J. Hierarchical multi-label classification based on over-sampling and hierarchy constraint for gene function prediction. *IEEJ transactions on electrical and electronic engineering*, Wiley Online Library, v. 7, n. 2, p. 183–189, 2012.

De Barros, F. K. H.; Selleti, A. L. J.; Queiroz, V. A. P.; Pereira, R. M.; Silla, C. N. Analyzing the Impact of Resampling Approaches on Chest X-Ray Images for COVID-19 Identification in a Local Hierarchical Classification Scenario. In: IEEE. *2021 IEEE 21st International Conference on Bioinformatics and Bioengineering (BIBE)*. 2021. p. 1–6.

Eisner, R.; Poulin, B.; Szafron, D.; Lu, P.; Greiner, R. Improving Protein Function Prediction using the Hierarchical Structure of the Gene Ontology. In: *2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*. 2005. p. 1–10.

Fagni, T.; Sebastiani, F. On the selection of negative examples for hierarchical text categorization. In: *Proceedings of the 3rd language technology conference*. 2007. p. 24–28.

Friedman, M. A comparison of alternative tests of significance for the problem of m rankings. *The annals of mathematical statistics*, JSTOR, v. 11, n. 1, p. 86–92, 1940.

Grzymala-Busse, J. W.; Grzymala-Busse, W. J. Handling missing attribute values. *Data mining and knowledge discovery handbook*, Springer, p. 33–51, 2010.

Haixiang, G. et al. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, v. 73, p. 220–239, 2017. ISSN 0957-4174. Available on: <https://www.sciencedirect.com/science/article/pii/S0957417416307175>.

Han, H.; Wang, W.-Y.; Mao, B.-H. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Springer. *International conference on intelligent computing*. 2005. p. 878–887.

Hart, P. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, v. 14, p. 515–516, 1968.

He, H.; Bai, Y.; Garcia, E. A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: Ieee. *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. 2008. p. 1322–1328.

Kiritchenko, S.; Canada, C.; Famili, F. Functional Annotation of Genes Using Hierarchical Text Categorization. *NRC Publications Archive (NPArC) Archives des publications du CNRC (NPArC)*, n. January 2005, 2014.

Kiritchenko, S.; Matwin, S.; Famili, A. F.; Others. Functional annotation of genes using hierarchical text categorization. In: *Proc. of the ACL Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*. 2005.

Klungpornkun, M.; Vateekul, P. Hierarchical text categorization using level based neural networks of word embedding sequences with sharing layer information. *Walailak Journal of Science and Technology (WJST)*, v. 16, n. 2, p. 121–131, 2019.

Krishnaiah, V.; Narsimha, G.; Chandra, N. S. Survey of classification techniques in data mining. *International Journal of Computer Sciences and Engineering*, v. 2, n. 9, p. 65–74, 2014.

Montenegro, C.; Santana, R.; Lozano, J. A. Introducing multi-dimensional hierarchical classification: Characterization, solving strategies and performance measures. *Neurocomputing*, v. 533, p. 141–160, 2023. ISSN 0925-2312. Available on: <https://www.sciencedirect.com/science/article/pii/S0925231223001984>.

Mrozek, P.; Panneerselvam, J. Efficient Resampling for Fraud Detection During Anonymised Credit Card Transactions with Unbalanced Datasets. *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*, p. 426–433, 2020.

Nakano, F. K.; Pinto, W. J.; Pappa, G. L.; Cerri, R. Top-down strategies for hierarchical classification of transposable elements with neural networks. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. 2017. p. 2539–2546.

Nemenyi, P. B. *Distribution-free multiple comparisons.* : Princeton University, 1963.

Pereira, R. M.; Bertolini, D.; Teixeira, L. O.; Silla Jr, C. N.; Costa, Y. M. G. COVID-19 identification in chest X-ray images on flat and hierarchical classification scenarios. *Computer methods and programs in biomedicine*, Elsevier, v. 194, p. 105532, 2020.

Pereira, R. M.; Costa, Y. M. G.; Silla Jr, C. N. Dealing with imbalanceness in hierarchical multi-label datasets using multi-label resampling techniques. In: IEEE. *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*. 2018. p. 818–824.

Pereira, R. M.; Costa, Y. M. G.; Silla Jr, C. N. Handling imbalance in hierarchical classification problems using local classifiers approaches. *Data Mining and Knowledge Discovery*, Springer, v. 35, n. 4, p. 1564–1621, 2021.

Pereira, R. M.; Costa, Y. M. G.; Silla Jr, C. N. Toward hierarchical classification of imbalanced data using random resampling algorithms. *Information Sciences*, Elsevier, v. 578, p. 344–363, 2021.

Ramírez-Corona, M.; Sucar, L. E.; Morales, E. F. Hierarchical multilabel classification based on path evaluation. *International Journal of Approximate Reasoning*, v. 68, p. 179–193, 2016. ISSN 0888-613X. Available on: <https://www.sciencedirect.com/science/article/pii/S0888613X15001073>.

Ruepp, A. et al. The FunCat , a functional annotation scheme for systematic classification of proteins from whole genomes. v. 32, n. 18, p. 5539–5545, 2004.

Serrano-Pérez, J.; Sucar, L. E. Hierarchical classification with Bayesian networks and chained classifiers. In: *The Thirty-Second International Flairs Conference*. 2019.

Serrano-Pérez, J.; Sucar, L. E. Artificial datasets for hierarchical classification. *Expert Systems with Applications*, v. 182, n. January, p. 115218, 2021.

Silla, C. N.; Freitas, A. A. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, v. 22, n. 1, p. 31–72, 2011. ISSN 1573-756X. Available on: <https://doi.org/10.1007/s10618-010-0175-9>.

Spelmen, V. S.; Porkodi, R. A Review on Handling Imbalanced Data. In: *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*. 2018. p. 1–11.

Stein, R. A.; Jaques, P. A.; Valiati, J. F. An analysis of hierarchical text classification using word embeddings. *Information Sciences*, v. 471, p. 216–232, 2019. ISSN 0020-0255. Available on: <https://www.sciencedirect.com/science/article/pii/S0020025518306935>.

Stiller, S.; Dueñas, J. F.; Hempel, S.; Rillig, M. C.; Ryo, M. Deep learning image analysis for filamentous fungi taxonomic classification: Dealing with small datasets with class imbalance and hierarchical grouping. *Biology Methods and Protocols*, Oxford University Press, v. 9, n. 1, p. bpae063, 2024.

Tang, H.; Wang, Y.; Tang, S.; Chu, D.; Li, C. A Randomized Clustering Forest Approach for Efficient Prediction of Protein Functions. *IEEE Access*, v. 7, p. 12360–12372, 2019.

Tomek, I. Two Modifications of CNN. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6, n. 11, p. 769–772, 1976.

Vens, C.; Struyf, J.; Schietgat, L.; Dzeroski, S.; Blockeel, H. Decision Trees for Hierarchical Multi-label Classification. *Springer New York LLC*, v. 73, n. 2, p. 185–214, 2008.

Wohlin, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. New York, NY, USA: Association for Computing Machinery, 2014. (EASE '14). ISBN 9781450324762. Available on: <https://doi.org/10.1145/2601248.2601268>.

Woolson, R. F. Wilcoxon signed-rank test. *Encyclopedia of biostatistics*, Wiley Online Library, v. 8, 2005.

Wu, F.; Zhang, J.; Honavar, V. Learning Classifiers Using Hierarchically Structured Class Taxonomies. In: Zucker, J.-D.; Saitta, L. (Ed.). *Abstraction, Reformulation and Approximation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 313–320. ISBN 978-3-540-31882-8.

A Análise de Classes do Conjunto de Dados

Soma das classes com exemplos nos conjuntos de treino e teste	**Soma das classes com exemplos no conjunto de treino, mas não no de teste**	**Soma das classes com exemplos no conjunto de teste, mas não no de treino**
1084	560	788

Tabela 16 – Análise da quantidade de exemplos no conjunto de dados *church*

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0005634, GO0005737,	GO0006633, GO0005096,	GO0000162, GO0005381,
GO0005739, GO0003674,	GO0031930, GO0006998,	GO0016925, GO0000307,
GO0006412, GO0008150,	GO0043001, GO0000048,	GO0042026, GO0000372,
GO0003735, GO0005575,	GO0030532, GO0042710,	GO0007129, GO0000727,
GO0005886, GO0000723,	GO0006829, GO0001671,	GO0000144, GO0005619,
GO0005515, GO0005783,	GO0030137, GO0007155,	GO0009306, GO0016592,
GO0005829, GO0042254,	GO0019005, GO0030427,	GO0006620, GO0004634,
GO0006468, GO0003723,	GO0005993, GO0009098,	GO0031207, GO0005667,
GO0005730, GO0005743,	GO0015095, GO0015693,	GO0008655, GO0042721,
GO0007047, GO0005842,	GO0005537, GO0005089,	GO0005658, GO0004028,
GO0003677, GO0000398,	GO0000751, GO0005471,	GO0017148, GO0008152,
GO0007124, GO0009277,	GO0005385, GO0004555,	GO0003893, GO0004749,
GO0016021, GO0042493,	GO0004681, GO0006114,	GO0000433, GO0008622,
GO0005843, GO0031202,	GO0007029, GO0000509,	GO0006166, GO0003873,
GO0009060, GO0005789,	GO0015646, GO0003883,	GO0006417, GO0004708,
GO0030437, GO0005759,	GO0017108, GO0004824,	GO0005543, GO0000138,
GO0016563, GO0006888,	GO0045041, GO0008409,	GO0004652, GO0046688,
GO0006365, GO0005741,	GO0001320, GO0004026,	GO0016237, GO0045821,
GO0003924, GO0006406,	GO0006048, GO0032093,	GO0006646, GO0019220,
GO0005643, GO0005935,	GO0006824, GO0000170,	GO0009410, GO0042623,
GO0006897, GO0045944,	GO0004239, GO0031463,	GO0003941, GO0005960,
GO0005198, GO0006357,	GO0000213, GO0006283,	GO0015793, GO0000126,
GO0005762, GO0003899,	GO0008757, GO0017136,	GO0004375, GO0006892,
GO0016020, GO0006368,	GO0042542, GO0004066,	GO0006635, GO0000813,
GO0006950, GO0006511,	GO0004857, GO0015888,	GO0009396, GO0000749,
GO0030468, GO0005732,	GO0004864, GO0000709,	GO0005674, GO0000755,
GO0001403, GO0006366,	GO0019236, GO0005882,	GO0017196, GO0006614,
GO0003700, GO0051082,	GO0051180, GO0005753,	GO0006598, GO0004725,
GO0006281, GO0006623,	GO0045129, GO0009061,	GO0006110, GO0000916,
GO0005794, GO0004842,	GO0008308, GO0006760,	GO0006857, GO0045143,
GO0006457, GO0005200,	GO0004042, GO0019856,	GO0006003, GO0006546,
GO0005934, GO0005816,	GO0016071, GO0030527,	GO0005845, GO0006626,
GO0003704, GO0006611,	GO0005835, GO0045787,	GO0031417, GO0030915,
GO0005768, GO0006367,	GO0000404, GO0016831,	GO0008559, GO0030100,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0006979, GO0005840,	GO0006499, GO0047066,	GO0006525, GO0005769,
GO0016887, GO0007126,	GO0043166, GO0015082,	GO0005756, GO0000706,
GO0016251, GO0005622,	GO0030690, GO0004252,	GO0016156, GO0004827,
GO0005625, GO0006407,	GO0004437, GO0004176,	GO0015168, GO0000316,
GO0030466, GO0006999,	GO0000321, GO0006901,	GO0005823, GO0015893,
GO0006887, GO0046540,	GO0045016, GO0008897,	GO0004872, GO0015840,
GO0007015, GO0000329,	GO0004682, GO0006740,	GO0004518, GO0016075,
GO0007059, GO0000082,	GO0004736, GO0048309,	GO0000739, GO0045861,
GO0001302, GO0003743,	GO0006430, GO0006986,	GO0016077, GO0030529,
GO0006409, GO0006609,	GO0000152, GO0030242,	GO0000767, GO0005324,
GO0006610, GO0006408,	GO0004316, GO0016810,	GO0008556, GO0000025,
GO0006607, GO0006608,	GO0007088, GO0006164,	GO0031416, GO0017111,
GO0000086, GO0000131,	GO0031578, GO0016576,	GO0042174, GO0000921,
GO0006970, GO0016573,	GO0003989, GO0004314,	GO0004427, GO0004609,
GO0000027, GO0006487,	GO0000214, GO0000346,	GO0004430, GO0030125,
GO0006413, GO0000750,	GO0046037, GO0006529,	GO0006552, GO0050821,
GO0004674, GO0030036,	GO0000121, GO0030691,	GO0005950, GO0004049,
GO0006461, GO0030490,	GO0005825, GO0006241,	GO0008415, GO0018987,
GO0009408, GO0043332,	GO0008451, GO0031201,	GO0005631, GO0004099,
GO0005624, GO0000778,	GO0015798, GO0004450,	GO0032889, GO0030688,
GO0006810, GO0000070,	GO0018065, GO0000506,	GO0004128, GO0000743,
GO0006383, GO0006338,	GO0006811, GO0032777,	GO0008466, GO0000078,
GO0004004, GO0006355,	GO0045552, GO0015343,	GO0003910, GO0043065,
GO0006486, GO0000910,	GO0004088, GO0007001,	GO0009069, GO0003880,
GO0008565, GO0007121,	GO0016580, GO0000349,	GO0030037, GO0045046,
GO0006360, GO0007165,	GO0000370, GO0051382,	GO0008652, GO0005744,
GO0016567, GO0005355,	GO0000781, GO0006449,	GO0006990, GO0006010,
GO0008645, GO0007020,	GO0043148, GO0016274,	GO0008554, GO0008283,
GO0005666, GO0005685,	GO0004365, GO0015114,	GO0016298, GO0030050,
GO0005353, GO0015578,	GO0008194, GO0004151,	GO0004325, GO0006418,
GO0030472, GO0005681,	GO0004343, GO0004559,	GO0004558, GO0005838,
GO0005682, GO0000122,	GO0030192, GO0004807,	GO0046658, GO0008360,
GO0007131, GO0030476,	GO0004044, GO0015124,	GO0031902, GO0001522,
GO0004672, GO0006364,	GO0004592, GO0001558,	GO0006008, GO0015672,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0030433, GO0006303,	GO0003939, GO0045892,	GO0019483, GO0015149,
GO0006696, GO0004871,	GO0003861, GO0019170,	GO0005776, GO0009922,
GO0006893, GO0006379,	GO0019887, GO0004301,	GO0035101, GO0016587,
GO0007266, GO0007117,	GO0045703, GO0004039,	GO0019948, GO0045026,
GO0006270, GO0005758,	GO0004422, GO0000066,	GO0006285, GO0051016,
GO0045047, GO0007096,	GO0006221, GO0003885,	GO0043630, GO0015758,
GO0004407, GO0016575,	GO0051183, GO0015772,	GO0004371, GO0000339,
GO0005887, GO0006879,	GO0006308, GO0045786,	GO0051920, GO0016076,
GO0008094, GO0006289,	GO0015811, GO0015878,	GO0008605, GO0048250,
GO0004175, GO0007031,	GO0006570, GO0000215,	GO0008290, GO0008380,
GO0000002, GO0006974,	GO0006518, GO0000010,	GO0006108, GO0043634,
GO0005215, GO0007264,	GO0004035, GO0045910,	GO0007107, GO0019563,
GO0006896, GO0030176,	GO0004413, GO0004641,	GO0015075, GO0045003,
GO0007035, GO0006914,	GO0004168, GO0006077,	GO0003916, GO0005981,
GO0000022, GO0042645,	GO0043618, GO0016428,	GO0017102, GO0004614,
GO0000228, GO0003713,	GO0051225, GO0004121,	GO0006067, GO0004619,
GO0005777, GO0006890,	GO0045950, GO0008677,	GO0004866, GO0005078,
GO0005774, GO0006096,	GO0015883, GO0000373,	GO0005785, GO0019413,
GO0005933, GO0006508,	GO0012505, GO0008315,	GO0019843, GO0004489,
GO0007005, GO0005665,	GO0004089, GO0018364,	GO0000015, GO0005386,
GO0040020, GO0006506,	GO0004075, GO0031966,	GO0018279, GO0043021,
GO0006886, GO0030528,	GO0032265, GO0051219,	GO0009847, GO0017025,
GO0016566, GO0030478,	GO0019432, GO0009072,	GO0008106, GO0015847,
GO0003714, GO0005680,	GO0006269, GO0051598,	GO0045298, GO0006880,
GO0005576, GO0006526,	GO0004833, GO0030135,	GO0016036, GO0005853,
GO0030479, GO0006350,	GO0008412, GO0042762,	GO0004366, GO0000795,
GO0006906, GO0008361,	GO0009204, GO0008353,	GO0005536, GO0004826,
GO0006606, GO0004843,	GO0004809, GO0004326,	GO0005047, GO0000703,
GO0005763, GO0042626,	GO0001406, GO0042723,	GO0015489, GO0006431,
GO0006555, GO0006301,	GO0015805, GO0007092,	GO0004165, GO0030060,
GO0006865, GO0046695,	GO0006033, GO0000396,	GO0004696, GO0005801,
GO0005811, GO0006378,	GO0007025, GO0032138,	GO0006813, GO0004849,
GO0005819, GO0009228,	GO0004318, GO0000310,	GO0006458, GO0030870,
GO0003682, GO0005736,	GO0051972, GO0016291,	GO0005827, GO0003870,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0006094, GO0030674,	GO0000333, GO0030678,	GO0004635, GO0005095,
GO0007010, GO0048017,	GO0004317, GO0005621,	GO0042538, GO0016035,
GO0003697, GO0006388,	GO0030071, GO0000247,	GO0006266, GO0015410,
GO0003684, GO0016514,	GO0004244, GO0000403,	GO0000101, GO0000006,
GO0005856, GO0003688,	GO0008198, GO0004573,	GO0004379, GO0045015,
GO0000742, GO0006342,	GO0031415, GO0004852,	GO0006390, GO0015744,
GO0000183, GO0005740,	GO0004385, GO0004477,	GO0000285, GO0017004,
GO0000501, GO0000508,	GO0006183, GO0015890,	GO0003881, GO0004933,
GO0006817, GO0005635,	GO0004830, GO0015780,	GO0000298, GO0016308,
GO0006402, GO0007120,	GO0004475, GO0004024,	GO0006335, GO0051037,
GO0003729, GO0000001,	GO0009237, GO0031126,	GO0017061, GO0005293,
GO0016538, GO0005669,	GO0015620, GO0003810,	GO0051864, GO0004015,
GO0000124, GO0000139,	GO0006825, GO0015225,	GO0008251, GO0008234,
GO0000114, GO0007094,	GO0015664, GO0006675,	GO0000920, GO0008271,
GO0006333, GO0015986,	GO0006800, GO0006119,	GO0006545, GO0004020,
GO0016586, GO0005792,	GO0008615, GO0018283,	GO0045118, GO0003986,
GO0005628, GO0000030,	GO0004070, GO0004061,	GO0004392, GO0004356,
GO0006450, GO0042147,	GO0045333, GO0015123,	GO0015204, GO0004425,
GO0006298, GO0006310,	GO0000034, GO0000800,	GO0016559, GO0006438,
GO0046961, GO0000142,	GO0009743, GO0048471,	GO0042803, GO0004164,
GO0042273, GO0000788,	GO0046107, GO0006741,	GO0030696, GO0043138,
GO0016192, GO0006116,	GO0030636, GO0030259,	GO0004856, GO0009636,
GO0005778, GO0000154,	GO0004421, GO0030950,	GO0031072, GO0004825,
GO0005802, GO0030287,	GO0043529, GO0004772,	GO0016976, GO0000100,
GO0006873, GO0006631,	GO0015880, GO0046855,	GO0000810, GO0016806,
GO0008047, GO0007091,	GO0001310, GO0004585,	GO0005274, GO0004417,
GO0030148, GO0030515,	GO0019265, GO0006864,	GO0005302, GO0016227,
GO0016571, GO0008301,	GO0043328, GO0008859,	GO0008190, GO0005099,
GO0000184, GO0005832,	GO0005315, GO0016653,	GO0000060, GO0004451,
GO0031145, GO0008054,	GO0045142, GO0000186,	GO0006083, GO0016424,
GO0005975, GO0016564,	GO0017110, GO0007006,	GO0045836, GO0046029,
GO0016125, GO0000245,	GO0051730, GO0006666,	GO0004742, GO0045117,
GO0004402, GO0006098,	GO0009202, GO0015866,	GO0004506, GO0006014,
GO0000026, GO0008298,	GO0004013, GO0006591,	GO0000040, GO0010133,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0000118, GO0006259,	GO0006541, GO0000178,	GO0004043, GO0006469,
GO0016579, GO0000790,	GO0006549, GO0009313,	GO0005952, GO0004331,
GO0006400, GO0030150,	GO0046015, GO0007050,	GO0046084, GO0006973,
GO0006493, GO0006272,	GO0042994, GO0015431,	GO0000046, GO0000208,
GO0006891, GO0005656,	GO0004771, GO0006436,	GO0004395, GO0031422,
GO0000028, GO0019898,	GO0016649, GO0016906,	GO0004243, GO0004460,
GO0006260, GO0043044,	GO0008989, GO0000391,	GO0030611, GO0051014,
GO0005830, GO0006537,	GO0008134, GO0019777,	GO0005094, GO0004583,
GO0008654, GO0005844,	GO0006657, GO0010044,	GO0006264, GO0015440,
GO0008033, GO0032266,	GO0016846, GO0004056,	GO0051204, GO0019888,
GO0000902, GO0005686,	GO0005338, GO0017157,	GO0016299, GO0015837,
GO0006267, GO0045045,	GO0003961, GO0042256,	GO0008418, GO0051085,
GO0005773, GO0005751,	GO0004529, GO0000115,	GO0004691, GO0004577,
GO0004526, GO0000079,	GO0015757, GO0032979,	GO0035004, GO0051228,
GO0007076, GO0005675,	GO0007000, GO0007030,	GO0004832, GO0031087,
GO0007103, GO0006744,	GO0030623, GO0006275,	GO0003895, GO0005371,
GO0003724, GO0004129,	GO0045490, GO0015127,	GO0032042, GO0006542,
GO0001308, GO0000502,	GO0004019, GO0001682,	GO0045134, GO0048255,
GO0006099, GO0030163,	GO0004474, GO0004339,	GO0016409, GO0043541,
GO0008250, GO0004579,	GO0008643, GO0006295,	GO0001409, GO0008487,
GO0000932, GO0000715,	GO0004037, GO0004844,	GO0004123, GO0000123,
GO0000812, GO0030488,	GO0006538, GO0018195,	GO0004663, GO0001401,
GO0005876, GO0005524,	GO0005951, GO0015581,	GO0008810, GO0030620,
GO0000282, GO0008017,	GO0016651, GO0006527,	GO0046087, GO0006562,
GO0004693, GO0000032,	GO0046421, GO0005662,	GO0003988, GO0019509,
GO0003701, GO0000145,	GO0016279, GO0032139,	GO0006863, GO0001408,
GO0007118, GO0015662,	GO0004372, GO0046100,	GO0005375, GO0008841,
GO0035267, GO0030126,	GO0032137, GO0043007,	GO0015238, GO0000770,
GO0007067, GO0000221,	GO0031371, GO0015228,	GO0016556, GO0003889,
GO0005048, GO0042787,	GO0008977, GO0006734,	GO0006427, GO0006750,
GO0031307, GO0006665,	GO0004862, GO0004648,	GO0015105, GO0004536,
GO0000794, GO0046856,	GO0003869, GO0004335,	GO0000253, GO0006559,
GO0006808, GO0051233,	GO0042736, GO0045547,	GO0015077, GO0048025,
GO0007534, GO0006113,	GO0015079, GO0004113,	GO0004034, GO0008534,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0031588, GO0005849,	GO0030004, GO0004553,	GO0042167, GO0007008,
GO0006882, GO0006629,	GO0001407, GO0004310,	GO0005427, GO0042602,
GO0004679, GO0000710,	GO0000179, GO0004315,	GO0004741, GO0008612,
GO0005637, GO0000256,	GO0031932, GO0043334,	GO0003876, GO0045132,
GO0031146, GO0008535,	GO0015788, GO0031942,	GO0008079, GO0004081,
GO0001402, GO0000209,	GO0004630, GO0004853,	GO0004079, GO0006613,
GO0000175, GO0016455,	GO0000262, GO0015157,	GO0009071, GO0015817,
GO0000747, GO0030134,	GO0019677, GO0015819,	GO0004657, GO0005869,
GO0006189, GO0007064,	GO0030136, GO0016532,	GO0006085, GO0006597,
GO0006470, GO0000074,	GO0006073, GO0004785,	GO0000906, GO0004401,
GO0042144, GO0000722,	GO0004187, GO0016259,	GO0006833, GO0008135,
GO0005654, GO0006207,	GO0004146, GO0003951,	GO0005057, GO0006481,
GO0003755, GO0006414,	GO0003862, GO0004178,	GO0019547, GO0004655,
GO0005085, GO0003702,	GO0043488, GO0007580,	GO0015848, GO0009074,
GO0000288, GO0006904,	GO0000500, GO0004697,	GO0016765, GO0008113,
GO0000119, GO0006644,	GO0006654, GO0000318,	GO0004148, GO0043139,
GO0015918, GO0000083,	GO0004351, GO0051292,	GO0000087, GO0006227,
GO0000783, GO0003678,	GO0010286, GO0015218,	GO0009436, GO0004363,
GO0016944, GO0030234,	GO0042148, GO0015087,	GO0019858, GO0008597,
GO0000300, GO0030008,	GO0006358, GO0000036,	GO0008137, GO0000774,
GO0051087, GO0045324,	GO0019627, GO0004038,	GO0000278, GO0006231,
GO0006895, GO0000780,	GO0032212, GO0000246,	GO0005854, GO0009353,
GO0042393, GO0016023,	GO0016971, GO0008793,	GO0008311, GO0015856,
GO0008540, GO0016485,	GO0004163, GO0016303,	GO0004620, GO0004644,
GO0006109, GO0016593,	GO0000395, GO0004750,	GO0031569, GO0003918,
GO0000171, GO0015940,	GO0031211, GO0040008,	GO0004694, GO0051220,
GO0000172, GO0007004,	GO0004754, GO0009062,	GO0006536, GO0015220,
GO0031011, GO0042138,	GO0017173, GO0008383,	GO0003978, GO0004590,
GO0000011, GO0005655,	GO0030414, GO0004851,	GO0004349, GO0016758,
GO0007070, GO0006488,	GO0005745, GO0019439,	GO0006916, GO0016768,
GO0006446, GO0019237,	GO0004462, GO0004333,	GO0004743, GO0003867,
GO0005664, GO0008379,	GO0005697, GO0004358,	GO0004496, GO0004766,
GO0005852, GO0006325,	GO0006345, GO0015801,	GO0004755, GO0033062,
GO0032045, GO0043141	GO0000095, GO0015230,	GO0000107, GO0009097,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0006616, GO0007231,	GO0009446, GO0019915,	GO0000267, GO0016772,
GO0005199, GO0009651,	GO0016040, GO0051300,	GO0015275, GO0006574,
GO0000158, GO0004012,	GO0031931, GO0042765,	GO0004160, GO0004638,
GO0005724, GO0043330,	GO0004680, GO0042719,	GO0006084, GO0007116,
GO0030435, GO0005798,	GO0006404, GO0016423,	GO0004354, GO0004930,
GO0006123, GO0006621,	GO0046975, GO0004586,	GO0004040, GO0032041,
GO0006006, GO0006625,	GO0000140, GO0015319,	GO0051312, GO0008295,
GO0016281, GO0043614,	GO0004615, GO0004332,	GO0000169, GO0046970,
GO0000159, GO0006839,	GO0000235, GO0006801,	GO0003866, GO0003894,
GO0003993, GO0006396,	GO0009049, GO0008202,	GO0006363, GO0005290,
GO0007114, GO0008121,	GO0004105, GO0009337,	GO0017062, GO0000319,
GO0008614, GO0006617,	GO0042177, GO0000277,	GO0004370, GO0018444,
GO0006271, GO0006464,	GO0004751, GO0031234,	GO0051268, GO0008174,
GO0008023, GO0005525,	GO0004057, GO0030120,	GO0030337, GO0007068,
GO0015677, GO0016558,	GO0006915, GO0042392,	GO0003942, GO0000062,
GO0004338, GO0006397,	GO0000406, GO0000775,	GO0043539, GO0004731,
GO0000411, GO0009086,	GO0016072, GO0006751,	GO0019789, GO0008159,
GO0006995, GO0005750,	GO0004550, GO0000254,	GO0016889, GO0016460,
GO0005786, GO0007119,	GO0008289, GO0051754,	GO0006900, GO0051269,
GO0019722, GO0019773,	GO0003923, GO0042800,	GO0004861, GO0015191,
GO0004888, GO0006359,	GO0008686, GO0045835,	GO0032217, GO0005948,
GO0015171, GO0006874,	GO0007568, GO0004637,	GO0045876, GO0004822,
GO0031384, GO0015146,	GO0045103, GO0003720,	GO0046470, GO0019654,
GO0006312, GO0000182,	GO0008276, GO0046027,	GO0006152, GO0004810,
GO0045722, GO0008104,	GO0004347, GO0000210,	GO0005337, GO0004196,
GO0015914, GO0043254,	GO0009966, GO0000400,	GO0019274, GO0006423,
GO0000059, GO0004602,	GO0006032, GO0004608,	GO0005052, GO0005288,
GO0008526, GO0009409,	GO0030692, GO0003972,	GO0015729, GO0050263,
GO0000754, GO0006592,	GO0043035, GO0004587,	GO0004512, GO0004604,
GO0006401, GO0017171,	GO0004345, GO0007093,	GO0015212, GO0009229,
GO0007089, GO0000113,	GO0004783, GO0000155,	GO0004018, GO0008144,
GO0006102, GO0007346,	GO0004806, GO0004313,	GO0019949, GO0006343,
GO0006513, GO0006273,	GO0001510, GO0005098,	GO0015720, GO0006287,
GO0005770, GO0007049,	GO0008453, GO0031055,	GO0008601, GO0046677,

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0005977, GO0005485, GO0003709, GO0000177, GO0000077, GO0046933, GO0000176, GO0006384, GO0000147, GO0000243, GO0005847, GO0004722, GO0031386, GO0051015, GO0006855, GO0006612, GO0045182, GO0000103, GO0006261, GO0000753, GO0007039, GO0005782, GO0005881, GO0005486, GO0000112, GO0006878, GO0046685, GO0003712, GO0006913, GO0006369, GO0003777, GO0030503, GO0008599, GO0006038, GO0019878, GO0030173, GO0006796, GO0008092, GO0005688, GO0016050, GO0042175, GO0006265, GO0006783, GO0015631, GO0000799, GO0005086, GO0017112, GO0000293, GO0006302, GO0000054, GO0030846, GO0006972, GO0005828, GO0005498, GO0016272, GO0009102, GO0005100, GO0000164, GO0000092, GO0006656, GO0000127, GO0005663, GO0045053, GO0000328, GO0005885, GO0008142, GO0009082, GO0045332,	GO0004455, GO0006352, GO0006435, GO0019344, GO0000907, GO0003922, GO0016574, GO0031026, GO0004312, GO0050826, GO0006106, GO0003977, GO0000163, GO0016255, GO0003871, GO0009142, GO0006812, GO0004362, GO0016598, GO0015184, GO0005462, GO0008398, GO0003786, GO0006165, GO0008117, GO0042283, GO0004488, GO0008486, GO0004792, GO0006353, GO0015173, GO0004250, GO0042124, GO0008299, GO0003879, GO0016791, GO0004072, GO0003992, GO0031491, GO0019776, GO0006031, GO0016480, GO0000729, GO0015723, GO0007242, GO0042576, GO0004329, GO0017022, GO0008934, GO0001578, GO0031119, GO0008574, GO0004847, GO0051599, GO0008168, GO0015719, GO0008173, GO0042720, GO0004650, GO0019408, GO0008670, GO0018738, GO0042221, GO0003721, GO0004581, GO0004476, GO0015959, GO0043085,	GO0000299, GO0004053, GO0016078, GO0031982, GO0015861, GO0046967, GO0005821, GO0045022, GO0004781, GO0006120, GO0005267, GO0051666, GO0008262, GO0006624, GO0045955, GO0051908, GO0004409, GO0030031, GO0046820, GO0031071, GO0017125, GO0051646, GO0003983, GO0043137, GO0004078, GO0004376, GO0030687, GO0009328, GO0004048, GO0030272, GO0000802, GO0017057, GO0000915, GO0003980, GO0003906, GO0004839, GO0043144, GO0016789, GO0030543, GO0005998, GO0007007, GO0019988, GO0050072, GO0008175, GO0006235, GO0008495, GO0003991, GO0045144, GO0006011, GO0004647, GO0005469, GO0043023, GO0004482, GO0003838, GO0015721, GO0003962, GO0051436, GO0015909, GO0003850, GO0009305, GO0009826, GO0004514, GO0048313, GO0005053, GO0004789, GO0009164, GO0030145, GO0042292,
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0006268, GO0007021, GO0004406, GO0004730, GO0000132, GO0009749, GO0031385, GO0006465, GO0004022, GO0005967, GO0000120, GO0030897, GO0030469, GO0006334, GO0043565, GO0009051, GO0009113, GO0003676, GO0001100, GO0000009, GO0042843, GO0009268, GO0007109, GO0009272, GO0005354, GO0005519, GO0015035, GO0003756, GO0003689, GO0000146, GO0000108, GO0009231, GO0005641, GO0006530, GO0005824, GO0031321, GO0005509, GO0005979, GO0001324, GO0006336, GO0005946, GO0007535, GO0000137, GO0007130, GO0000014, GO0001101, GO0005956, GO0006356, GO0004067, GO0030127, GO0004439, GO0015892, GO0032447, GO0001405, GO0016602, GO0000785, GO0008177, GO0005992, GO0006790, GO0006122, GO0003779, GO0019568, GO0030508, GO0000084, GO0005742, GO0005871, GO0005937, GO0006276,	GO0000064, GO0006146, GO0030627, GO0048487, GO0006944, GO0032977, GO0015685, GO0005529, GO0019318, GO0032299, GO0048188, GO0015809, GO0006846, GO0008843, GO0005365, GO0009298	GO0004340, GO0006987, GO0004798, GO0004306, GO0043631, GO0042790, GO0004142, GO0005968, GO0004076, GO0004642, GO0042326, GO0004435, GO0004684, GO0008419, GO0000227, GO0005727, GO0009127, GO0004428, GO0003716, GO0000017, GO0005672, GO0017005, GO0005784, GO0046482, GO0046969, GO0015198, GO0003934, GO0043161, GO0004226, GO0015806, GO0004134, GO0030619, GO0015125, GO0010142, GO0007024, GO0008138, GO0015824, GO0044419, GO0030261, GO0031310, GO0004424, GO0006561, GO0004582, GO0015174, GO0008441, GO0004848, GO0016180, GO0004305, GO0015813, GO0006797, GO0004794, GO0015849, GO0004571, GO0015344, GO0043014, GO0043291, GO0004934, GO0004080, GO0006314, GO0045140, GO0042624, GO0006421, GO0007188, GO0008444, GO0015495, GO0000942, GO0010009, GO0015186,
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0000056, GO0006013, GO0005955, GO0030847, GO0042274, GO0000133, GO0043130, GO0006627, GO0019897, GO0009269, GO0009435, GO0016439, GO0000903, GO0035064, GO0000136, GO0007062, GO0030123, GO0006354, GO0045941, GO0042729, GO0008143, GO0006566, GO0030869, GO0000417, GO0042255, GO0003690, GO0005788, GO0015937, GO0030295, GO0006749, GO0000724, GO0004540, GO0006448, GO0007034, GO0006826, GO0019243, GO0019541, GO0000928, GO0000772, GO0006501, GO0046982, GO0005677, GO0007097, GO0005834, GO0004467, GO0008233, GO0006078, GO0000148, GO0007329, GO0031110, GO0005822, GO0031204, GO0003825, GO0004709, GO0006000, GO0006012, GO0018063, GO0031262, GO0015926, GO0030489, GO0030140, GO0006144, GO0046020, GO0042149, GO0031124, GO0019706, GO0010038, GO0000117,		GO0004845, GO0004788, GO0004399, GO0004311, GO0004418, GO0004130, GO0006337, GO0045141, GO0004497, GO0015871, GO0046907, GO0048280, GO0048278, GO0000415, GO0005874, GO0004073, GO0007186, GO0018008, GO0051177, GO0020037, GO0004817, GO0004618, GO0015039, GO0004591, GO0050790, GO0017120, GO0005275, GO0004819, GO0005771, GO0032453, GO0016577, GO0009063, GO0006168, GO0004765, GO0000018, GO0032218, GO0006550, GO0032872, GO0005506, GO0006233, GO0004747, GO0006830, GO0007026, GO0004664, GO0015188, GO0004799, GO0001321, GO0045040, GO0003785, GO0006432, GO0015233, GO0000408, GO0015858, GO0004382, GO0000356, GO0000412, GO0004132, GO0004135, GO0004322, GO0006835, GO0003984, GO0009041, GO0015810, GO0003856, GO0004721, GO0009070, GO0005352, GO0045033,
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0043486, GO0031684, GO0005034, GO0009267, GO0031518, GO0004633, GO0006512, GO0000161, GO0000735, GO0045727, GO0032786, GO0017056, GO0000347, GO0019933, GO0018318, GO0033100, GO0004723, GO0032806, GO0000725, GO0000096, GO0000105, GO0042162, GO0005657, GO0006284, GO0006118, GO0045449, GO0004596, GO0009055, GO0003746, GO0008320, GO0006280, GO0030447, GO0005938, GO0043162, GO0005545, GO0000407, GO0015846, GO0030473, GO0006828, GO0030015, GO0030276, GO0001300, GO0008541, GO0006081, GO0000289, GO0006090, GO0007323, GO0000784, GO0007033, GO0045039, GO0009073, GO0019740, GO0000220, GO0019774, GO0003887, GO0004520, GO0000717, GO0007265, GO0005978, GO0000135, GO0005971, GO0007532, GO0005884, GO0006020, GO0000049, GO0015559, GO0030491, GO0005384,		GO0032473, GO0050516, GO0004131, GO0015887, GO0008277, GO0031499, GO0004077, GO0015700, GO0003987, GO0030246, GO0032184, GO0003855, GO0016197, GO0004106, GO0030968, GO0006505, GO0046135, GO0030174, GO0003841, GO0017103, GO0008374, GO0004639, GO0000250, GO0004045, GO0005247, GO0019368, GO0019187, GO0015741, GO0000165, GO0015804, GO0003864, GO0004764, GO0004487, GO0006661, GO0015997, GO0016763, GO0005516, GO0006428, GO0006089, GO0043008, GO0006534, GO0004141, GO0030974, GO0003959, GO0000814, GO0016966, GO0008184, GO0005483, GO0006045, GO0003890, GO0007624, GO0007190, GO0005310, GO0004479, GO0006220, GO0031428, GO0006842, GO0006167, GO0006425, GO0000102, GO0003994, GO0000031, GO0015250, GO0003985, GO0030866, GO0010107, GO0015718, GO0004610,
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0008270, GO0005342, GO0030121, GO0005787, GO0006474, GO0005296, GO0000776, GO0008028, GO0008536, GO0019655, GO0006997, GO0004169, GO0001400, GO0015802, GO0015696, GO0046854, GO0030482, GO0007017, GO0004521, GO0008623, GO0000033, GO0015239, GO0004748, GO0006730, GO0006827, GO0005851, GO0000335, GO0005940, GO0004003, GO0006075, GO0000151, GO0005088, GO0018456, GO0051377, GO0030188, GO0006807, GO0008519, GO0045002, GO0006885, GO0005749, GO0046323, GO0003891, GO0019388, GO0006688, GO0005868, GO0006816, GO0008154, GO0006490, GO0015392, GO0000076, GO0000055, GO0015071, GO0030026, GO0017183, GO0000150, GO0005880, GO0016233, GO0000149, GO0043248, GO0032040, GO0046686, GO0016070, GO0006452, GO0016584, GO0000938, GO0050839, GO0004739, GO0000226,		GO0004115, GO0006163, GO0004601, GO0043140, GO0008330, GO0005477, GO0004502, GO0015386, GO0047964, GO0006850, GO0048193, GO0005846, GO0008333, GO0046488, GO0004811, GO0004016, GO0043115, GO0004821, GO0000248, GO0008131, GO0016530, GO0043495, GO0016114, GO0004327, GO0004107, GO0008835, GO0050177, GO0006039, GO0003706, GO0045898, GO0004793, GO0000354, GO0015923, GO0043626, GO0015078, GO0000225, GO0003717, GO0008186,GO0000173, GO0004737
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0005761, GO0006121, GO0004805, GO0043625, GO0003711, GO0030904, GO0004177, GO0031505, GO0000798, GO0015992, GO0004197, GO0018348, GO0000080, GO0008408, GO0045116, GO0019538, GO0007090, GO0030371, GO0000290, GO0016226, GO0008310, GO0015606, GO0032197, GO0000110, GO0005507, GO0006567, GO0004032, GO0030029, GO0000752, GO0004364, GO0045721, GO0004190, GO0006097, GO0000276, GO0015359, GO0045121, GO0051083, GO0015680, GO0042555, GO0006066, GO0000280, GO0004100, GO0008272, GO0005980, GO0016973, GO0000266, GO0003843, GO0004445, GO0006814, GO0006279, GO0006605, GO0000297, GO0000707, GO0050000, GO0004017, GO0004651, GO0016972, GO0004373, GO0015867, GO0004622, GO0006405, GO0008156, GO0000274, GO0043132, GO0000731, GO0030969, GO0004449, GO0030467,		
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0016429, GO0005388, GO0007234, GO0005678, GO0030014, GO0003872, GO0016616, GO0004092, GO0005083, GO0047429, GO0004828, GO0007127, GO0004534, GO0004523, GO0016407, GO0004408, GO0006532, GO0008483, GO0005965, GO0043458, GO0006376, GO0009003, GO0045009, GO0017059, GO0004616, GO0008821, GO0006883, GO0004660, GO0005875, GO0004588, GO0016363, GO0009117, GO0017087, GO0005779, GO0000302, GO0045740, GO0015234, GO0006434, GO0046513, GO0030162, GO0009437, GO0031206, GO0043200, GO0016926, GO0019781, GO0015116, GO0008553, GO0006420, GO0004815, GO0000350, GO0000943, GO0015031, GO0032044, GO0000818, GO0030497, GO0000075, GO0006564, GO0006467, GO0043224, GO0000736, GO0004478, GO0019395, GO0042134, GO0003937, GO0005286, GO0003849, GO0004535, GO0000111,		
Continuação da página anterior		

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0005097, GO0017119, GO0019266, GO0003747, GO0015088, GO0005953, GO0016929, GO0006476, GO0015908, GO0000097, GO0004096, GO0030530, GO0003964, GO0030042, GO0006101, GO0006491, GO0000386, GO0000019, GO0006672, GO0004069, GO0019354, GO0000817, GO0019912, GO0006370, GO0016565, GO0004103, GO0016126, GO0000038, GO0004396, GO0004758, GO0004643, GO0046294, GO0031941, GO0000301, GO0010032, GO0031490, GO0016481, GO0000168, GO0005478, GO0016301, GO0009083, GO0004033, GO0004662, GO0005673, GO0003680, GO0006071, GO0050291, GO0017076, GO0015183, GO0004367, GO0008608, GO0017050, GO0008237, GO0007023, GO0010008, GO0004823, GO0018347, GO0006086, GO0006296, GO0004420, GO0006784, GO0000217, GO0016591, GO0031429, GO0051724, GO0004240, GO0016074, GO0003774,		(2, 2, 0), (3, 1, 0)

Continuação da página anterior

Tabela 16 – continuação da página anterior

Classes com exemplos nos conjuntos de treino e teste	**Classes com exemplos no conjunto de treino, mas não no de teste**	**Classes com exemplos no conjunto de teste, mas não no de treino**
GO0004820, GO0004416, GO0007243, GO0007232, GO0004801, GO0048476, GO0008195, GO0005850, GO0004814, GO0005945, GO0004352, GO0004802, GO0006869, GO0003999, GO0032445, GO0003917, GO0009092, GO0009090, GO0030122, GO0045184, GO0006426, GO0005671, GO0006415, GO0045011, GO0000304, GO0006429, GO0017040, GO0030003, GO0006424, GO0004458, GO0000922, GO0005313, GO0004222, GO0004712, GO0030007, GO0004186, GO0003938, GO0031314, GO0004707, GO0004818, GO0004410, GO0031251, GO0016491, GO0046459, GO0008053, GO0004124, GO0003954, GO0004108, GO0000815, GO0045010, GO0031475, GO0007569, GO0004084, GO0006533, GO0000244, GO0004636, GO0004816, GO0006650, GO0031118, GO0030133, GO0004829, GO0005754, GO0000324, GO0006348		