A New Hierarchical Redundancy Eliminated Tree Augmented Naïve Bayes Classifier for Coping with Gene Ontology-based Features

Cen Wan[†,‡]   C.WAN@UCL.AC.UK
Alex A. Freitas[‡]   A.A.FREITAS@KENT.AC.UK
[†] Department of Computer Science, University College London, London, United Kingdom
[‡] School of Computing, Univerisity of Kent, Canterbury, United Kingdom

Abstract

The Tree Augmented Naïve Bayes classifier is a type of probabilistic graphical model that can represent some feature dependencies. In this work, we propose a Hierarchical Redundancy Eliminated Tree Augmented Naïve Bayes (HRE–TAN) algorithm, which considers removing the hierarchical redundancy during the classifier learning process, when coping with data containing hierarchically structured features. The experiments showed that HRE–TAN obtains significantly better predictive performance than the conventional Tree Augmented Naïve Bayes classifier, and enhanced the robustness against imbalanced class distributions, in aging-related gene datasets with Gene Ontology terms used as features.

1. Introduction

This work proposes a new type of Tree Augmented Naïve Bayes (TAN) classifier, namely the Hierarchical Redundancy Eliminated Tree Augmented Naïve Bayes (HRE–TAN) algorithm, which is designed for coping with features organized into a hierarchy (e.g., a tree or a DAG – directed acyclic graph). In this paper the features are DAG-structured Gene Ontology (GO) terms, in datasets where instances represent genes to be classified into pro-longevity or anti-longevity genes. However, the proposed algorithm can also be applied to other classification datasets with hierarchical features.

Tree Augmented Naïve Bayes (TAN) is a type of semi-Naïve Bayes classifier that relaxes Naïve Bayes' feature independence assumption, by allowing each feature to depend on at most one non-class variable feature. This type of tree structure-based Bayesian classifier is able to

represent some feature dependencies and scales to large datasets more efficiently than other Bayesian classifiers that represent more complex feature dependencies. In this work, we focus on one of the most computationally efficient TAN classifiers (Friedman et al., 1997; Keogh & Pazzani, 1999; Jiang et al., 2005; Zhang & Ling, 2001), which essentially computes the conditional mutual information (CMI) for each pair of features given the class attribute and then builds a Maximum Weight Spanning Tree (MST), where an edge's weight is given by its CMI (Friedman et al., 1997). Then, a randomly selected vertex of the MST acts as the tree's root, and the edge directions are propagated accordingly.

2. Background

2.1. The Gene Ontology and Hierarchical Redundancy

The Gene Ontology (GO) uses unified and structured vocabularies to describe gene functions (The Gene Ontology Consortium, 2000). Most GO terms are hierarchically structured by an "is-a" relationship, where each GO term is a specialization of its ancestor (more generic) terms. For example, GO:0003674 (molecular function) is the root of the DAG for molecular function terms, and it is also the parent of GO:0003824 (catalytic activity), which is in turn the parent of GO:0004803 (transposase activity).

This feature hierarchy has two types of hierarchical redundancy. First, if a GO term (feature) takes the value "1" for a given instance (gene), this implies its ancestor terms in the GO DAG also take the value "1" for that instance. Conversely, if the GO term takes the value "0" for a given instance, its descendants in the DAG also take the value "0" for that instance. In order to cope with those types of hierarchical redundancy, in our previous works (Wan & Freitas, 2013; Wan et al., 2015; Wan, 2015; 2016), three types of filter hierarchical feature selection algorithms were proposed, i.e., MR, HIP and the hybrid HIP-MR. Those three algorithms eliminate/alleviate the

above types of hierarchical redundancy in a data pre-processing phase, before learning the classifier. In contrast, the proposed HRE–TAN eliminates the hierarchical redundancy during the classifier learning phase.

2.2. Lazy Learning

A "lazy" learning method performs the learning process in the testing phase, building a specific classification model for each testing instance to be classified (Aha, 1997; Pereira et al., 2011), rather than building a general classifier for all testing instances. The newly proposed TAN classifier in this work is based on lazy learning, since it selects features for each testing instance separately.

3. Hierarchical Redundancy Eliminated Tree Augmented Naïve Bayes (HRE–TAN)

This is a new type of tree-based Bayesian classifier based on the lazy learning approach, and it performs an embedded hierarchical feature redundancy elimination, rather than in a pre-processing step. As mentioned in Section 1, a conventional TAN method builds a MST to detect dependencies among features, but it assumes that the features are "flat", not hierarchical. In contrast, the proposed method eliminates the hierarchical redundancy between features when it builds the MST for each testing instance. As discussed in Section 2.1, two vertices are hierarchically redundant if one of them is an ancestor or descendant of the other and they have the same feature value ("1" or "0"). In essence, HRE–TAN checks the status of each edge before adding it into the Undirected Acyclic Graph (UDAG) that will be transformed into the MST later. The status of an edge will be set to "Unavailable" if either of the vertices connected by the edge is hierarchically redundant, with respect to the vertices that have already been included in the UDAG. The pseudocode of HRE–TAN is described in Algorithms 1 and 2.

In Algorithm 1, in the first part of the HRE–TAN algorithm (lines 1–12), HRE–TAN firstly generates the Directed Acyclic Graph (DAG) for the current dataset with a corresponding set of vertices (features) $\mathbb{X}$ and set of edges $\mathbb{E}$. Then it generates the set of ancestor and descendant features for each feature x_i, denoted $\mathbb{A}(x_i)$ and $\mathbb{D}(x_i)$, respectively. $\mathbf{Status}_{<E>}(x_i, x_j)$, which is initialized as "Available", denotes the selection status of the edge connecting vertices x_i and x_j. $\mathbf{CMI}_{<E>}(x_i, x_j)$ denotes the value of CMI (conditional mutual information) for the edge $\mathbf{E}(x_i, x_j)$. All edges are sorted in descending order of their CMI value (a greater CMI value means a higher priority of adding the edge into the UDAG). In the second part of the HRE–TAN algorithm (lines 13–21), the tree $\mathbb{T}$ will be built for each testing instance (adopting a lazy

learning approach) by calling the procedure HRE–MST() for building the Hierarchical Redundancy Eliminated Maximum Weight Spanning Tree (HRE–MST). Then the **TrainSet** and the current testing instance $\mathbf{Inst}_{<w>}$ will be re-created with the set $\mathbb{X}'$ of the features included in the tree, so that only those features will be used for classifying the re-created testing instance. The re-created **TrainSet_T** and **Inst_T**$_{<w>}$ with tree $\mathbb{T}$ are then used to build a lazy TAN model that classifies $\mathbf{Inst}_{<w>}$ in line 17. Finally, in lines 18–20 all edges in the DAG have their status re-assigned to "Available", as a preparation to process the next testing instance.

Algorithm 1 Lazy Hierarchical Redundancy Eliminated Tree Augmented Naïve Bayes (HRE–TAN)

1: Initialize **DAG** with all features in Dataset;
2: Initialize **TrainSet**;
3: Initialize **TestSet**;
4: **for** each feature $x_i \in \mathbb{X}$ **do**
5:     Initialize $\mathbb{A}(x_i)$ in **DAG**;
6:     Initialize $\mathbb{D}(x_i)$ in **DAG**;
7: **end for**
8: **for** each $\mathbf{E}(x_i, x_j) \in \mathbb{E}$ **do**
9:     Calculate $\mathbf{CMI}_{<E>}(x_i, x_j)$ using **TrainSet**;
10:     Initialize $\mathbf{Status}_{<E>}(x_i, x_j) \leftarrow$ *"Available"*;
11: **end for**
12: Sort all $\mathbf{E}(x_i, x_j) \in \mathbb{E}$ by descending order of **CMI**;
13: **for** each instance $\mathbf{Inst}_{<w>} \in \mathbf{TestSet}$ **do**
14:     $\mathbb{T} = $ HRE–MST(**DAG**, $\mathbf{Inst}_{<w>}$, $\mathbb{A}(\mathbb{X})$, $\mathbb{D}(\mathbb{X})$, $\mathbb{E}$);
15:     Re-create **TrainSet_T** with feature set $\mathbb{X}' \in \mathbb{T}$;
16:     Re-create **Inst_T**$_{<w>}$ with feature set $\mathbb{X}' \in \mathbb{T}$;
17:     Classify by TAN($\mathbb{T}$, **TrainSet_T**, **Inst_T**$_{<w>}$);
18:     **for** each $\mathbf{E}(x_i, x_j) \in \mathbb{E}$ **do**
19:         Re-assign $\mathbf{Status}_{<E>}(x_i, x_j) \leftarrow$ *"Available"*;
20:     **end for**
21: **end for**

Algorithm 2 shows the pseudocode for building the HRE–MST. $\mathbf{NR}(x_i, x_j, \mathbf{Inst}_{<w>}, \mathbf{DAG})$ is a Boolean function that returns "True" if nodes x_i and x_j are non-hierarchically-redundant in the current testing instance $\mathbf{Inst}_{<w>}$, given the feature DAG. $\mathbf{NoCycle}(\mathbf{E}(x_i, x_j), \mathbf{UDAG})$ is a Boolean function that returns "True" if there is no cycle in the **UDAG** after adding edge $\mathbf{E}(x_i, x_j)$. If the edge satisfies all the conditions in line 3 of Algorithm 2, it will be added into the **UDAG** (line 4). Once the algorithm has added the edge $\mathbf{E}(x_i, x_j)$ to the **UDAG**, for each of the two nodes connected by that edge, denoted as x_g (line 5), the algorithm will consider each of the nodes which is either
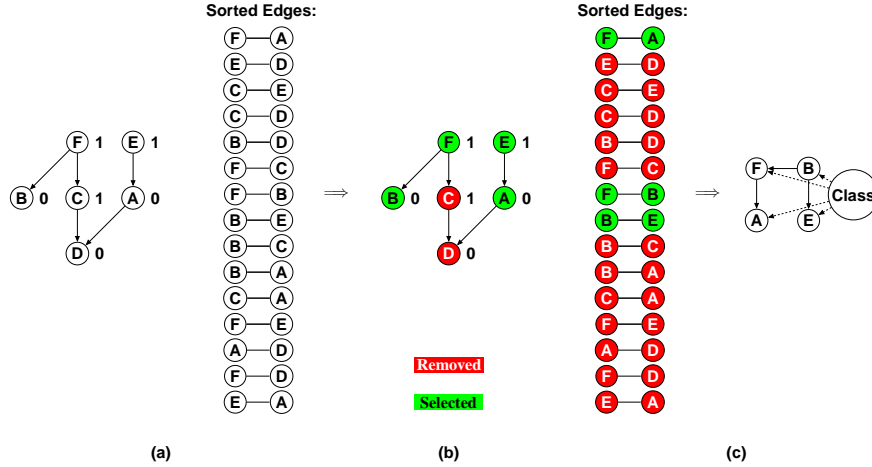
Figure 1. Example of HRE–TAN operation initially with a set of features structured as a DAG

an ancestor or a descendant of x_g in the feature **DAG**, denoting each such ancestor/descendant as x_h (line 6). If feature x_g and its ancestor/descendant feature x_h have the same value in the current testing instance $\mathbf{Inst}_{<w>}$ (line 7), indicating a hierarchical redundancy in that pair of features, then the *for each* loop in lines 8–10 will set to "Unavailable" the status of all edges where one of the nodes is x_h – line 8, where the symbol "$*$" is a wildcard matching any node. In other words, among the set of hierarchically-redundant nodes (features) with the same value, HRE–TAN selects the node included in the edge having higher conditional mutual information (CMI), since Algorithm 2 processes edges in descending order of CMI.

To explain how Algorithms 1 and 2 work, we use the example DAG shown in Figure 1.a, where the left part is a feature hierarchy consisting of three paths from a root to a leaf node of the **DAG**, i.e., node F to node B; node F to node D; and node E to node D. The right part of Figure 1.a shows the edges (for all pair of nodes) in descending order of **CMI**. HRE–TAN firstly adds edge $\mathbf{E}(F, A)$ into the UDAG, since its selection status is "*Available*"; nodes F and A are not hierarchically-redundant; and there is no cycle in the **UDAG** after adding edge $\mathbf{E}(F, A)$. Then, Algorithm 2 will delete all edges that contain hierarchically redundant nodes with respect to node F or node A, in order to minimize feature redundancy. Node C is redundant with respect to node F, because both of them have value "1" and are located in the same path in Figure 1.a. So, all edges containing node C (i.e., $\mathbf{E}(C, E)$, $\mathbf{E}(C, D)$, $\mathbf{E}(F, C)$, $\mathbf{E}(B, C)$, $\mathbf{E}(C, A)$) will be unavailable to be added into the **UDAG**. Also, node D is redundant with respect to node A, because both of them have value "0" and are located in the same path. Then, all edges containing node

D (i.e., $\mathbf{E}(E, D)$, $\mathbf{E}(C, D)$, $\mathbf{E}(B, D)$, $\mathbf{E}(A, D)$, $\mathbf{E}(F, D)$) will be unavailable to be added into the **UDAG**. Note that this hierarchical redundancy elimination process will dramatically reduce the size of the search space of candidate TAN structures.

Algorithm 2 Hierarchical Redundancy Eliminated Maximum Weight Spanning Tree (HRE–MST)

1: Initialize an Empty **UDAG**;

2: **for** each $\mathbf{E}(x_i, x_j) \in \mathbb{E}$ **do**

3:     **if** $\{\mathbf{Status}_{<E>}(x_i, x_j) = \text{"Available"}\} \wedge$
        $\{\mathbf{NR}(x_i, x_j, \mathbf{Inst}_{<w>}, \mathbf{DAG})\} \wedge$
        $\{\mathbf{NoCycle}(\mathbf{E}(x_i, x_j), \mathbf{UDAG})\}$ **then**

4:         add $\mathbf{E}(x_i, x_j)$ into **UDAG**;

5:         **for** each x_g in $\{x_i, x_j\}$ **do**

6:             **for** each x_h in $\{\mathbb{A}(x_g) \cup \mathbb{D}(x_g)\}$ **do**

7:                 **if** $\mathbf{V}(x_g, \mathbf{Inst}_{<w>}) = \mathbf{V}(x_h, \mathbf{Inst}_{<w>})$ **then**

8:                     **for** each $\mathbf{E}(x_h, *)$ **do**

9:                         $\mathbf{Status}_{<E>}(x_h, *) \leftarrow$ "*Unavailable*";

10:                 **end for**

11:                 **end if**

12:             **end for**

13:         **end for**

14:     **end if**

15: **end for**

16: Choose **Root** by Randomly selecting vertex x in **UDAG**;

17: Build the tree ($\mathbb{T}$) by marking direction of all edges from the **Root** outwards to other vertices;

18: Return $\mathbb{T}$;

Table 1. Sensitivity (± standard error), specificity (± standard error) and GMean values obtained by HRE–TAN and TAN over 28 datasets

	Caenorhabditis elegans Datasets						*Drosophila melanogaster Datasets*					
	HRE-TAN			**TAN**			**HRE-TAN**			**TAN**		
	Sens.	*Spec.*	*GMean*	*Sens.*	*Spec.*	*GMean*	*Sens.*	*Spec.*	*GMean*	*Sens.*	*Spec.*	*GMean*
BP	41.1 ± 2.4	76.8 ± 2.1	**56.2**	34.0 ± 3.2	79.6 ± 2.3	52.0	86.8 ± 3.2	30.6 ± 10.2	**51.5**	92.3 ± 2.9	19.4 ± 8.4	42.3
MF	23.1 ± 4.8	75.3 ± 5.4	41.7	37.2 ± 5.8	61.4 ± 5.0	**47.8**	86.8 ± 3.4	41.2 ± 8.8	**59.8**	91.2 ± 3.3	20.6 ± 5.0	43.3
CC	24.5 ± 3.6	80.8 ± 3.0	44.5	39.8 ± 3.0	78.2 ± 2.2	**55.8**	75.8 ± 5.8	28.6 ± 9.7	46.6	90.3 ± 3.6	32.1 ± 11.6	**53.8**
BP+MF	42.3 ± 2.3	80.0 ± 2.6	**58.2**	35.2 ± 1.9	80.3 ± 2.2	53.2	87.0 ± 3.3	31.6 ± 6.5	52.4	92.4 ± 3.3	23.7 ± 6.9	46.8
BP+CC	44.6 ± 3.0	74.4 ± 3.6	57.6	42.7 ± 3.1	81.7 ± 2.7	**59.1**	84.6 ± 2.4	32.4 ± 10.6	52.4	86.8 ± 4.0	18.9 ± 7.6	40.5
MF+CC	32.4 ± 3.3	79.8 ± 3.2	50.8	40.6 ± 3.4	74.4 ± 3.6	**55.0**	87.1 ± 4.4	39.5 ± 5.5	**58.7**	90.6 ± 3.3	31.6 ± 5.0	53.5
BP+MF+CC	44.2 ± 3.9	79.3 ± 2.9	**59.2**	39.5 ± 2.8	80.1 ± 2.6	56.2	82.6 ± 3.4	47.4 ± 8.7	**62.6**	92.4 ± 2.4	18.4 ± 5.3	41.2
	Mus musculus Datasets						*Saccharomyces cerevisiae Datasets*					
	HRE-TAN			**TAN**			**HRE-TAN**			**TAN**		
	Sens.	*Spec.*	*GMean*	*Sens.*	*Spec.*	*GMean*	*Sens.*	*Spec.*	*GMean*	*Sens.*	*Spec.*	*GMean*
BP	86.8 ± 5.5	47.1 ± 4.7	**63.9**	89.7 ± 3.7	41.2 ± 4.9	60.8	20.0 ± 7.4	93.5 ± 1.7	**43.2**	3.3 ± 3.3	98.9 ± 1.1	18.1
MF	83.1 ± 3.3	42.4 ± 9.3	**59.4**	89.2 ± 4.0	33.3 ± 9.4	54.5	0.0 ± 0.0	96.9 ± 1.7	0.0	0.0 ± 0.0	97.7 ± 1.2	0.0
CC	86.4 ± 4.0	41.2 ± 9.7	**59.7**	75.8 ± 4.4	41.2 ± 8.3	55.9	12.5 ± 6.1	93.5 ± 2.9	34.2	16.7 ± 7.0	95.9 ± 2.1	**40.0**
BP+MF	83.8 ± 4.5	41.2 ± 6.8	**58.8**	86.8 ± 3.4	35.3 ± 5.4	55.4	26.7 ± 10.9	95.8 ± 1.5	**50.6**	3.3 ± 3.3	99.0 ± 0.7	18.1
BP+CC	79.4 ± 4.9	47.1 ± 9.7	61.2	88.2 ± 3.6	47.1 ± 9.7	**64.5**	26.7 ± 6.7	94.1 ± 2.1	**50.1**	10.0 ± 5.1	99.0 ± 0.7	31.5
MF+CC	89.7 ± 3.0	35.3 ± 9.6	56.3	88.2 ± 4.2	41.2 ± 10.0	**60.3**	10.3 ± 6.1	95.4 ± 1.9	**31.3**	5.0 ± 5.0	98.5 ± 0.8	22.2
BP+MF+CC	85.3 ± 3.7	44.1 ± 8.9	61.3	91.2 ± 3.2	41.2 ± 8.6	61.3	23.3 ± 7.1	96.2 ± 1.4	**47.3**	0.0 ± 0.0	99.0 ± 0.6	0.0

After edges with node C or D had their selection status set to "*Unavailable*", edge $\mathbf{E}(F, B)$ – the next one available in the sorted list – will be added into the **UDAG**, since nodes F and B are not redundant (although both of them are in the same path in Figure 1.a, their values are different), and there is no cycle in the **UDAG** after adding that edge. Node B is not redundant with respect to any other node, so no edge has its status set to "Unavailable" in this step. Then, $\mathbf{E}(B, E)$ will be added into the **UDAG** as the next available edge in the sorted edge list, since this edge also satisfies all conditions in line 3 of Algorithm 2. Then, $\mathbf{E}(B, A)$, $\mathbf{E}(F, E)$ and $\mathbf{E}(E, A)$ will be processed in turn. However, none of them will be added into the **UDAG**, since this would create a cycle in that **UDAG**. Figure 1.b shows the selection status of features after processing all edges, while green color denotes the features were kept and included in the UDAG, whereas red color denotes features were removed and not included in the UDAG. Finally, HRE–TAN randomly selects a node as the root, which is used to mark directions of all edges in order to build the MST. Figure 1.c shows the final tree classifier including all selected features, with choosing feature B as the root. After finding the HRE–MST (i.e., tree $\mathbb{T}$), the training dataset and current testing instance will be re-created, and the testing instance will be classified using the built tree (line 17 in Algorithm 1). Then the selection status of all edges will be re-assigned as "Available" in line 19 of Algorithm 1, as a preparation for processing the next testing instance.

4. Computational Experiments

4.1. Aging-related Genes Datasets

We adopted the aging-related genes datasets used by our previous work (Wan & Freitas, 2015). The datasets consist of aging-related genes as instances and 7 different types of combination of Gene Ontology terms as features, e.g., biological process (BP) terms with molecular function (MF) terms, or molecular function (MF) terms with cellular component (CC) terms. The aging-related genes information was about 4 different modal organisms, i.e., *Caenorhabditis elegans* (CE), *Drosophila melanogaster* (DM), *Mus musculus* (MM) and *Saccharomyces cerevisiae* (SC), obtained from Human Ageing Genomic Resources (HAGR) GenAge database (Tacutu et al., 2013). Therefore, in total we have 28 different datasets (4 model organisms times 7 types of GO terms combinations).

4.2. Experimental Results and Discussion

We conducted a head-to-head comparison between the newly proposed HRE–TAN and the conventional TAN classifier based on their predictive accuracy. We used a well-known 10-fold cross validation procedure to evaluate the predictive accuracy measured by the geometric mean (Gmean) of Sensitivity and Specificity, i.e., the square root of the product of Sensitivity and Specificity. Sensitivity denotes the proportion of positive (pro-longevity) genes correctly classified as positive; whilst Specificity denotes

the ratio of negative (anti-longevity) genes correctly classified as negative. Table 1 displays the experimental results of HRE–TAN and TAN on the 28 datasets. The figures in bold denote higher GMean value among the two algorithms in each dataset. Overall, HRE–TAN and TAN obtained the higher GMean value in 18 and 8 datasets, respectively, with the GMean result being a tie in the other two datasets. According to the two-tailed Wilcoxon signed-rank test, HRE–TAN significantly outperformed TAN at the 0.05 significance level.

Notably, the class distribution on the datasets is imbalanced. Hence, we evaluated the robustness of HRE–TAN and TAN against imbalanced class distributions, by calculating the correlation coefficient r between GMean and the degree of class imbalance $\mathbf{D}$, given by: $\mathbf{D}$ (i.e., $\mathbf{D} = 1 - \frac{\#(Minor)}{\#(Major)}$, where $\#(Minor)$ denotes the number of instances belonging to the minority class and $\#(Major)$ denotes the number of instances belonging to the majority class. The values of $\mathbf{D}$ range from 0.234 (CE–MF dataset) to 0.856 (SC–BP+MF+CC dataset). The linear relationship between GMean and $\mathbf{D}$ values is shown in the scatter plots in Figure 2, where the red straight lines denote the fitted linear regression models. Obviously, HRE–TAN has better robustness against class imbalance than TAN, since HRE–TANs GMean decreases more slowly with an increase in $\mathbf{D}$ than TAN.
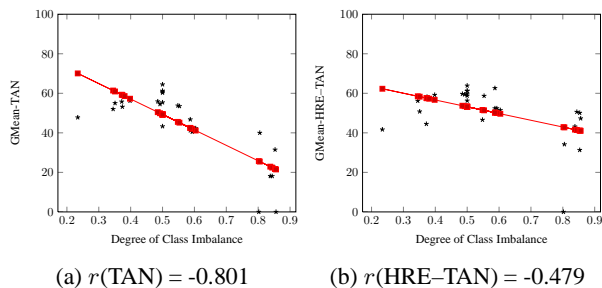

(a) r(TAN) = -0.801   (b) r(HRE–TAN) = -0.479

Figure 2. Linear relationship between $\mathbf{D}$ and GMean Values

5. Conclusion and Future Research Directions

In this work, we proposed and evaluated a new type of TAN classifier, i.e., HRE–TAN, which considers eliminating the hierarchical redundancy between features (hierarchical Gene Ontology terms) during the construction of the tree of features that is used as part of the classifier. The experiments show that HRE–TAN significantly outperforms the conventional TAN classifier on the tasks of classifying aging-related genes into pro-longevity or anti-longevity genes. In future work, we will further evaluate the performance of HRE–TAN in other datasets of hierarchical features and exploit other criteria to eliminate hierarchical redundancy during the classifier learning phase.

References

Aha, D. W. *Lazy Learning*. Kluwer Academic Publishers, Norwell, MA, 1997.

Friedman, N., Geiger, D., and Goldszmidt, M. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, November 1997.

Jiang, L., Zhang, H., Cai, Z., and Su, J. Learning tree augmented naive bayes for ranking. *Database Systems for Advanced Applications*, pp. 688–698, January 2005.

Keogh, E. J. and Pazzani, M. J. Learning augmented bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *Proc. the seventh international workshop on artificial intelligence and statistics*, pp. 225–230, Florida, USA, January 1999.

Pereira, R. B., Plastino, A., Zadrozny, B., de C. Merschmann, L. H., and Freitas, A. A. Lazy attribute selection: Choosing attributes at classification time. *Intelligent Data Analysis*, 15 (5):715–732, August 2011.

Tacutu, R., Craig, T., Budovsky, A., Wuttke, D., Lehmann, G., Taranukha, D., Costa, J., Fraifeld, V. E., and de Magalhães, J. P. Human ageing genomic resources: Integrated databases and tools for the biology and genetics of ageing. *Nucleic Acids Research*, 41(D1):D1027–D1033, January 2013.

The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, May 2000.

Wan, C. *Novel Hierarchical Feature Selection Methods for Classification and Their Application to Datasets of Ageing-Related Genes*. PhD thesis, University of Kent, 2015.

Wan, C. Novel hierarchical feature selection algorithms for predicting genes' aging-related function. *AI Matters*, 2:23–24, 2016.

Wan, C. and Freitas, A. A. Prediction of the pro-longevity or anti-longevity effect of *Caenorhabditis Elegans* genes based on Bayesian classification methods. In *Proc. of IEEE International Conference on Bioinformatics and Biomedicine (BIBM 2013)*, pp. 373–380, Shanghai, China, December 2013.

Wan, C. and Freitas, A. A. Two methods for constructing a gene ontology-based feature selection network for a Bayesian network classifier and applications to datasets of aging-related genes. In *Proc. of the Sixth ACM Conference on Bioinformatics, Computational Biology and Health Informatics (ACM-BCB 2015)*, pp. 27–36, Atlanta, USA, Sept. 2015.

Wan, C., Freitas, A. A., and de Magalhães, J. P. Predicting the pro-longevity or anti-longevity effect of model organism genes with new hierarchical feature selection methods. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 12(2):262–275, March 2015.

Zhang, H. and Ling, C. X. An improved learning algorithm for augmented naive bayes. *Advances in Knowledge Discovery and Data Mining*, 2035:581–586, April 2001.