Shared Ensembles using Multi-trees

Vicent Estruch, Cèsar Ferri, José Hernández-Orallo, M. José Ramírez-Quintana
{vestruch, cferri, jorallo, mramirez}@dsic.upv.es
Dep. de Sistemes Informàtics i Computació,
Universitat Politècnica de València,
Valencia, Spain

VII Conferencia Iberoamericana de Inteligencia Artificial
Sevilla, 12-15 November 2002
Introduction

- Machine Learning techniques that construct a model/hypothesis (e.g. ANN, DT, SVM, ...):
  - usually devoted to obtain one single model:
    - As accurate as possible (close to the “target” model).
    - Other (presumably less accurate) models are discarded.
  - An old alternative has recently been popularised:
    - “Every consistent hypothesis should be taken into account”

But... How?
Ensemble Methods (1/3)

- Ensemble Methods (Multi-classifiers):
  - Generate multiple (and possibly) heterogeneous models and then combine them through voting or other fusion methods.

- Much better results (in terms of accuracy) than single models when the number and variety of classifiers is high.
Ensemble Methods (Multi-classifiers):

- Different topologies: simple, stacking, cascading, ...

- Different generation policies: boosting, bagging, randomisation, ...

- Different fusion methods: majority voting, average, maximum, ...
Main drawbacks:

- **Computational costs**: huge amounts of memory and time are required to obtain and store the set of hypotheses (ensemble).
- **Throughput**: the application of the combined model is slow.

The solution of these drawbacks would boost the applicability of ensemble methods in machine learning applications.
Ensembles of Decision Trees

- **Decision Tree:**
  - Each internal node represents a condition.
  - Each leaf assigns a class to the examples that fall under that leaf.

- **Forest:** several decision trees can be constructed.
  - Many trees have common parts.
  - Traditional ensemble methods repeat those parts:
    - memory and time ↑↑↑.
    - comprehensibility is lost.
Decision Tree Shared Ensembles

- **Shared ensemble:**
  - Common parts are shared in an AND/OR tree structure.

- Construction space and time resources are highly reduced.

- Throughput is also improved by this technique.
Decision Tree Shared Ensembles

- **Previous work:**
  - Multiple Decision Trees (Kwok & Carter 1990)
  - Option Decision Trees (Buntine 1992)
    - The AND/OR tree structure is populated (partially) breadth-first.
  - Combination has been performed:
    - Using weighted combination (Buntine 1992).
    - Using majority voting combination (Kohavi & Kunz 1997).
  - Different conclusions on where alternatives are especially beneficial:
    - At the bottom of the tree (Buntine).
      - Trees are quite similar → Accuracy improvement is low.
    - At the top of the tree (Kohavi & Kunz).
      - Trees share few parts → Space resources are exhausted as in other non-shared ensembles (boosting, bagging, ...).
Previous work:

- **Drawbacks of Option Decision Trees:**
  - The number of alternative options is very difficult to be determined during the construction stage → size of the AND/OR structure is mostly unpredictable.
  - The fusion strategy (weighted, majority) determines the policy and number of alternative trees to be explored.
  - An “option factor” is required. The appropriate value highly depends on each particular dataset.
    - For option factor values such as 0.4, some datasets suffer an exponential increase of the number of nodes.
    - “Soybean was the extreme case, which increased from 68 nodes to 203,577 nodes” (Kohavi & Kunz 1997).
Multi-tree Construction

- **New Way of Populating the AND/OR Tree:**
  - The first tree is constructed in the classical eager way.
  - Discarded alternative splits are stored in a list.
    - Repeat $n$ times:
      - Once a tree is finished, the best alternative split (according to a “wakening” criterion) is chosen.
      - The branch is finished using the classical eager way.
  - This amounts to a ‘beam’ search → Anytime algorithm.
    - Extensions and alternatives can happen at any part of the tree (top, bottom).
    - The populating strategy can be easily changed.
    - The fusion strategy can also be flexibly modified.
    - The desired size of the AND/OR tree can be specified quite precisely.
Fusion Methods

- Combination on the Multi-tree:
  - The number of trees grows exponentially wrt. the number of alternative OR-nodes explored:
    - Advantages: ensembles are now much bigger with a constant increase of resources. Presumably, the combination will be more accurate.
    - Disadvantages: the combination at the top is unfeasible.
  - Global fusion techniques would be prohibitive.
Local Fusion

- First Stage. Classical top-down:
  - Each example to be predicted is distributed top-down into many alternative leaves.
  - The example is labelled in each leaf (class vector).

- Second Stage. The fusion goes bottom-up:
  - Whenever an OR-node is found. The (possibly) inconsistent predictions are combined through a local fusion method:

- Fusion of millions or billions of trees can be performed efficiently.
Local Fusion Methods

- Class vector transformation:
  - Good loser, bad loser, majority, difference, ...

- Fusion strategy
  - Sum, arithmean, product, geomean, max, min, ...

- When the fusioned vector reaches the top, the class with the greatest value is chosen.

- Examples:

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Good loser</th>
<th>Bad loser</th>
<th>Majority</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>{ 40, 10, 30 }</td>
<td>{ 80, 0, 0 }</td>
<td>{ 40, 0, 0 }</td>
<td>{ 1, 0, 0 }</td>
<td>{ 0, -60, -20 }</td>
</tr>
<tr>
<td></td>
<td>{ 7, 2, 10 }</td>
<td>{ 0, 0, 19 }</td>
<td>{ 0, 0, 10 }</td>
<td>{ 0, 0, 1 }</td>
<td>{ -5, -15, 1 }</td>
</tr>
</tbody>
</table>

MIN:  

- { 7, 2, 10 }  
  - c  
  - a b c  
  - a b c  
  - a b c  
  - c
Experiments (1/4)

Experimental setting:

- 15 datasets from the UCI repository.
- Multi-tree implemented in the SMILES system.
- Splitting criterion: GainRatio (C4.5).
- Second node selection criterion (wakening criterion): random.
- Boosting and Bagging from WEKA.
Experiments (2/4)

- Comparison between fusion techniques

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80.69</td>
<td>5.01</td>
<td>81.24</td>
<td>4.66</td>
<td>76.61</td>
</tr>
<tr>
<td>2</td>
<td>91.22</td>
<td>2.25</td>
<td>91.25</td>
<td>2.26</td>
<td>83.38</td>
</tr>
<tr>
<td>3</td>
<td>94.17</td>
<td>4.06</td>
<td>94.34</td>
<td>3.87</td>
<td>89.06</td>
</tr>
<tr>
<td>4</td>
<td>80.09</td>
<td>6.26</td>
<td>79.91</td>
<td>6.13</td>
<td>76.97</td>
</tr>
<tr>
<td>5</td>
<td>95.63</td>
<td>3.19</td>
<td>95.77</td>
<td>3.18</td>
<td>93.28</td>
</tr>
<tr>
<td>6</td>
<td>94.53</td>
<td>5.39</td>
<td>94.20</td>
<td>5.66</td>
<td>94.00</td>
</tr>
<tr>
<td>7</td>
<td>99.67</td>
<td>1.30</td>
<td>99.71</td>
<td>1.18</td>
<td>81.00</td>
</tr>
<tr>
<td>8</td>
<td>73.35</td>
<td>5.86</td>
<td>73.73</td>
<td>5.82</td>
<td>74.53</td>
</tr>
<tr>
<td>9</td>
<td>97.87</td>
<td>2.00</td>
<td>97.91</td>
<td>1.80</td>
<td>97.58</td>
</tr>
<tr>
<td>10</td>
<td>94.52</td>
<td>4.25</td>
<td>93.76</td>
<td>5.10</td>
<td>92.05</td>
</tr>
<tr>
<td>11</td>
<td>62.50</td>
<td>16.76</td>
<td>63.25</td>
<td>16.93</td>
<td>61.63</td>
</tr>
<tr>
<td>12</td>
<td>97.50</td>
<td>8.33</td>
<td>97.50</td>
<td>9.06</td>
<td>97.75</td>
</tr>
<tr>
<td>13</td>
<td>63.60</td>
<td>12.59</td>
<td>64.33</td>
<td>11.74</td>
<td>62.00</td>
</tr>
<tr>
<td>14</td>
<td>81.73</td>
<td>3.82</td>
<td>82.04</td>
<td>3.78</td>
<td>78.93</td>
</tr>
<tr>
<td>15</td>
<td>94.06</td>
<td>6.00</td>
<td>93.88</td>
<td>6.42</td>
<td>91.47</td>
</tr>
<tr>
<td>Geomean</td>
<td>85.83</td>
<td>4.72</td>
<td>85.99</td>
<td>4.71</td>
<td>82.53</td>
</tr>
</tbody>
</table>
Experiments (3/4)

- Combination Accuracy compared to other Ensemble Methods:

![Graph showing accuracy over iterations for different ensemble methods, including Boosting, Bagging, and Multitree.]
Experiments (4/4)

- Combination Resources compared to other Ensemble Methods:

![Graph showing comparison of Seconds vs Iterations for different methods: Boosting, Bagging, and Multi-tree.](image-url)
Conclusions

- Multi-tree as an alternative to other population strategies for shared decision tree ensembles:
  - Anytime character
    - The first tree is obtained in the same way as classical eager decision tree learning.
    - We ask for further solutions on demand.
  - Population (and hence resources) is scalable and easy to be controlled.
  - Fusion strategies are flexible.
    - Maximum fusion strategy seems to be the best one.
- Same or even better accuracy results than other ensemble methods with significantly lower resource consumption.
Conclusions

- Some further improvements:
  - Forgetting: not all the alternative OR-nodes are stored. Memory and time requirements are reduced even further with the same accuracy results.
  - Other uses of the multi-tree structure: extraction of the “best” single tree (Occam, archetype, ...).

- **SMILES** is freely available at: