Overview

Junjie Zhang

April 2020

1 Introduction

Overview of the graph-based deep forest

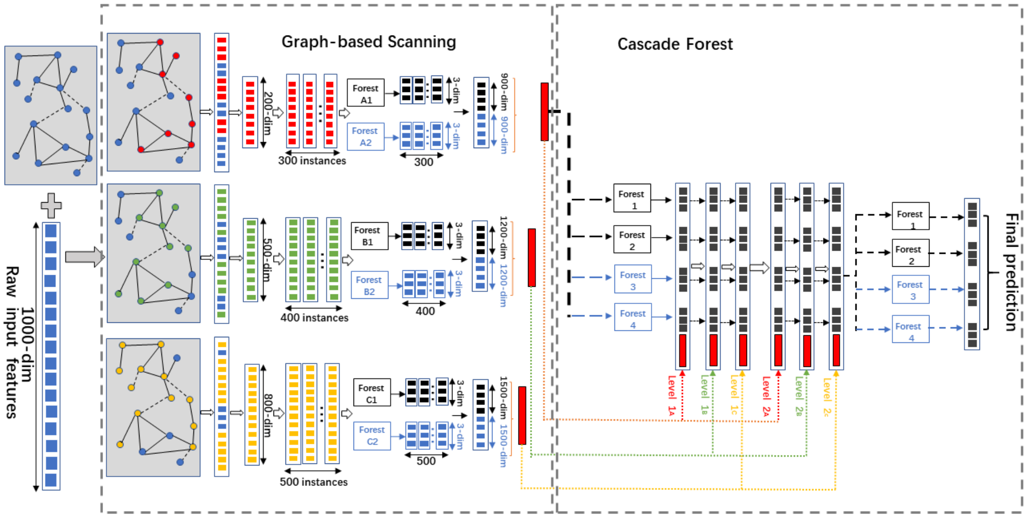
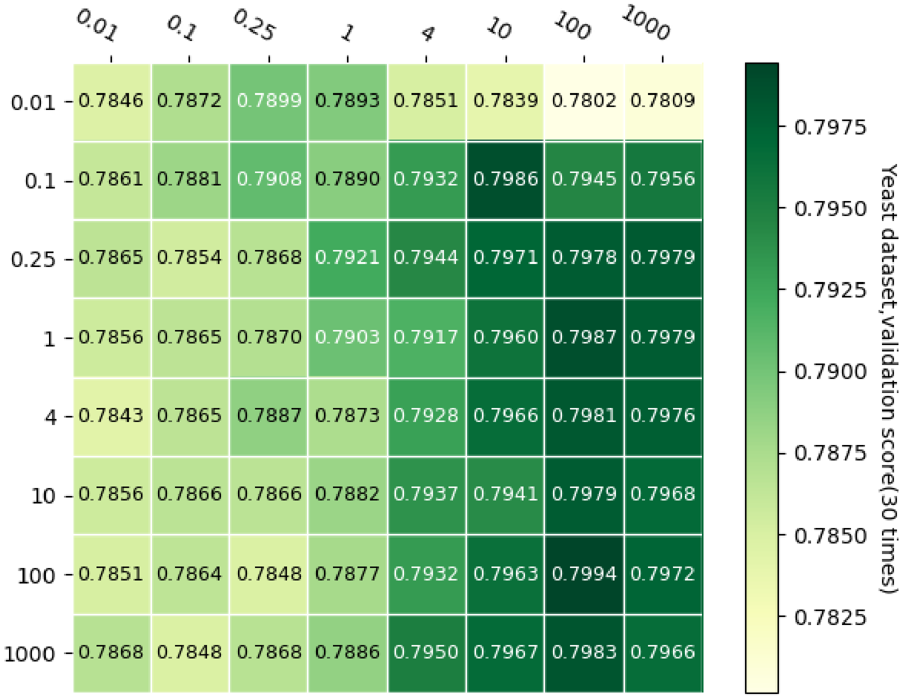


Figure 1: The overall procedure of the graph-based deep forest.Suppose there are three classes to predict,raw features are 1000-dim and three graph-based search strategy are used(length of a walk:(200,400,500),the number of walks:(300,400,500),respectively.

Algorithm 1 Graph-based Deep Forest

```
Input: Graph $G = (V, E, \pi)$,Walks per node $r$,Walk Length set $L$,Scale set $S$,Return $p$,In-out $q$,Raw Input $D$
function Graph-basedDeepForest
    Initialize $Transformed\quad dataset\quad T$ to Empty
    for all $l^{'} \in L$,all $s^{'} \in S$ do
        $t \leftarrow$ Graph-basedScanning($G, r, l^{'}, s^{'}, p, q, D$)
        Append $t$ to $T$
    end for
    $res \leftarrow$ CascadeForest(T)
    return $res$
end function
Input: Graph $G = (V, E, \pi)$,Walks per node $r$,Walk Length $l$,Scale $s$,Return $p$,In-out $q$,Raw Input $D$
function Graph-basedScanning
    $\pi$ = PreprocessModifiedWeights($G, p, q$)
    Graph $G^{'} = (V, E, \pi)$
    Initialize $walks$ to Empty
    for $iter \leftarrow 1$ to $l$ do
        for all node $u \in V$ do
            $walk \leftarrow$ Node2Walk($G^{'}, u, l$)
            Append $walk$ to $walks$
        end for
    end for
    $t \leftarrow$ ScanningGraph($D, walks$)
    return $t[: s]$
end function
Input: Graph $G^{'} = (V, E, \pi)$,Start node $u$,Length $l$
function Node2Walk
    Initialize $walk$ to $[u]$
    for $walk\_iter \leftarrow 1$ to $l$ do
        $curr \leftarrow walk[-1]$
        $V_{curr} \leftarrow$ GetNeighbors($curr, G^{'}$)
        $s \leftarrow$ AliasSample($V_{curr}, \pi$)
        Append $s$ to $walk$
    end for
    return $walk$
end function
```



	0.01	0.1	0.25	1	4	10	100	1000
0.01	0.7846	0.7872	0.7899	0.7893	0.7851	0.7839	0.7802	0.7809
0.1	0.7861	0.7881	0.7908	0.7890	0.7932	0.7986	0.7945	0.7956
0.25	0.7865	0.7854	0.7868	0.7921	0.7944	0.7971	0.7978	0.7979
1	0.7856	0.7865	0.7870	0.7903	0.7917	0.7960	0.7987	0.7979
4	0.7843	0.7865	0.7887	0.7873	0.7928	0.7966	0.7981	0.7976
10	0.7856	0.7866	0.7866	0.7882	0.7937	0.7941	0.7979	0.7968
100	0.7851	0.7864	0.7848	0.7877	0.7932	0.7963	0.7994	0.7972
1000	0.7868	0.7848	0.7868	0.7886	0.7950	0.7967	0.7983	0.7966

Figure 2: Validation score of the yeast dataset.60% for training,20% for validating,$iter\_num$=30,n_trees=25,p value along the y axis,q value along the x axis.

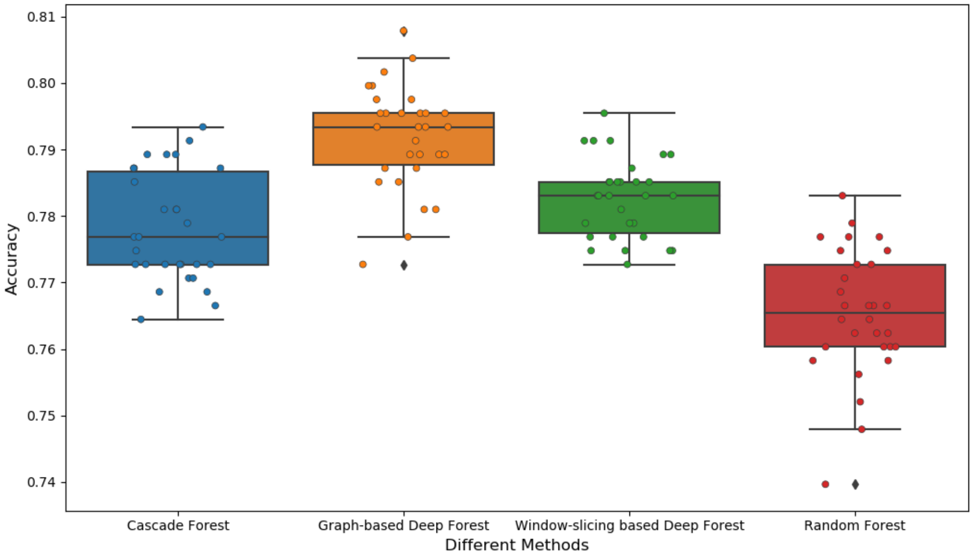


Figure 3: The accuracy's comparison of four methods by using remaining testing data.For graph-based deep forest,we used p=100,q=100.$Walk\_length\_set$= $(d/16, d/8, d/4)$,$window\_slicing\_size$=$(d/16, d/8, d/4)$,where d=number of raw input's feature.We have guaranteed that graph-based scanning and window slicing scanning generated features of same size.