STUDENT: AYUSH BADOLA

Convolution kernels for natural language

Michael Collins
mcollins@research.att.com
AT&T
Labs-Research

Nigel Duffy
nigeduff@cse.ucse.edu
Department of
Computer Science
University of
California

Kernel methods have been used in many algorithms like perceptron support vector machines and PCA. This paper discusses the use of convolution kernel(tree kernel) in Natural Language Processing Task .
NLP tasks involves complex structures like trees,queues,strings,etc.These structures need to be converted into feature vectors.The authors describes how the tree kernel(convolution kernels) can be used to track such complex and high dimensional features,like parse tree in case of parsing task .They also show how the computations are reduced using tree kernel.
A tree can be represented with the help of n-dimensional vector as
h(T)=(h1(T),h2(T),h3(T),.....hn(T))

Where h1(T),h2(T)...hn(T) is the number of occurrences of i'th tree fragment in the tree T.
Now the tree kernel is defined as the inner dot product between two trees T1 and T2 as K(T1,T2)=h(T1).h(T2).
Let the set of nodes in trees T1 and T2 be N1 and N2.Now an indicator function is defined which gives 1 if the subtree is seen rooted at node n and 0 in other case.

$hi(T1)=\sum_{n1\in N1} Ii(n1)$ and $hi(T2)=\sum_{n2\in N2} Ii(n2)$

$h(T1).h(T2)=\sum_{n1\in N1}\sum_{n2\in N2} C(n1,n2)$ where $C(n1,n2)=\sum_{i} Ii(n1).Ii(n2)$..
If the production at n1 and n2 are different C(n1,n2)=0
If the production at n1 and n2 are same and n1 and n2 are pre-terminals ,then C(n1,n2)=1
Else If the production at n1 and n2 are same and n1 and n2 are not pre-terminals

Then $C(n1,n2)=\prod_{j=1}^{nc(n1)} (1+C(ch(n1,j),ch(n2,j)))$

Where nc(n1) is the number of children of n1 in the tree.(this is the recursive definition of out kernel function)

Further there were some issues like the value of kernel was depending greatly on size of the tree which was solved by normalising kernel as
$K'(T1,T2)=K(T1,T2)/\sqrt{K(T1,T1)K(T2,T2)}$
Secondly the tree fragments were scaled according to their relative importance which depends on the size of the tree.This was done by introducing parameter $0<\lambda <= 1$.
This change was modified in the recursive definition of the kernel function.
C(n1,n2)=λ

$C(n1,n2)=\lambda.\prod_{j=1}^{nc(n1)} (1+C(ch(n1,j),ch(n2,j)))$

It was found that the time complexity was also reduced since in this case the time complexity does not depend on the number of nodes in a parse tree(which could be exponentially large) but it depends on the kernel function.

To check the utility of the convolution kernel it was applied to the problem of parsing in ATIS corpus.a PCFG was trained on the data and a voted perceptron with the tree kernel and the results were obtained for both cases.The voted perceptron performed much better than the PCFGs with the tree kernel that was applied to it.