GaKCo: a Fast GApped k-mer string Kernel using COunting

Ritambhara Singh, Arshdeep Sekhon, Kamran Kowsari, Jack Lanchantin, Beilun Wang, Yanjun Qi

University of Virginia, Department of Computer Science

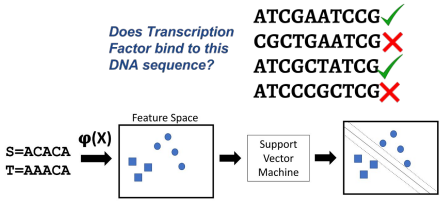
WiML 2017

1. Overview

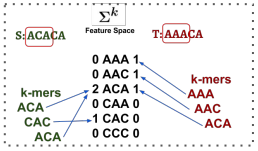
String Kernel techniques, especially those using gapped k-mers as features, have obtained great success in classifying sequences like DNA, protein, and text. However, the state-of-the-art gk-SVM runs extremely slow when we increase the dictionary size (Σ) or allow more mismatches (M). We propose a fast algorithm for calculating Gapped k-mer Kernel using Counting (GaKCo)-

1. *Faster* than state-of-the-art gk-SVM
2. Independent of dictionary size Σ and can *scale up* to large values of M and Σ.
3. *Parallelizable*

2. String Kernel + SVM framework

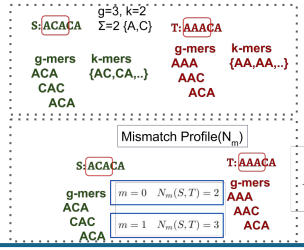


3. Spectrum Kernel

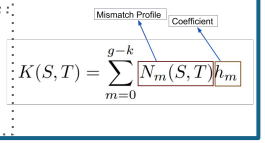


$$K(S,T) = \sum_{\gamma \in \Gamma_k} c_S(\gamma) \cdot c_T(\gamma)$$

4. Gapped k-mer Kernel



$$K(S,T) = \sum_{\gamma \in \Theta_g} c_S(\gamma) \cdot c_T(\gamma)$$



5. GaKCo

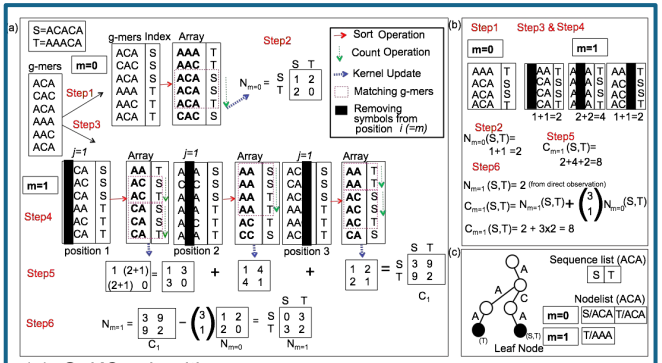
-*Algorithm*: GaKCo performs g-mer based cumulative counting of cooccurence to calculate N_m (independent of dictionary size(Σ)).

-*Parallelization*: GaKCo groups computations for each value of m into an independent function, making it naturally parallelizable.

Algorithm 1 GaKCo

Require: L, g, k ▨ L=Array list of g-mers

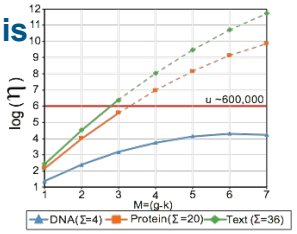
```
procedure CalculateKernel (L, g, k)
    M = g − k
    N = MismatchProfile (L, g, M)
    K = 0
    for m : 0 ..... M do
        h_m = C(g−k, m)
        K = K + N_m · h_m
procedure MismatchProfile (L, g, M)
    for m = 0 ...M do                ▨ Parallel threads
        C_m = 0                      ▨ Cumulative Profile
        n_pos = C(g, m)              ▨ Number of positions
        for i : 0..... n_pos do
            C^i_m = 0
            L^i = removePosition (L, i)
            L^i = sort (L^i)
            C^i_m = countAnd Update (L^i)
            C_m = C_m + C^i_m
    for m : 0 = M do
        N_m = C_m
        for j : 0 = m − 1 do
            N_m = N_m − C(g−j, m−j) N_j
    return N                         ▨ N = [N_0, ..., N_M]
Ensure: K                            ▨ Kernel Matrix
```



(a). GaKCo algorithm
(b). Overcounting when calculating cumulative mismatch profile for m=1.
(c). *gkm-SVM* uses a trie based data structure and a separate nodelist at each leafnode for calculating N_m.

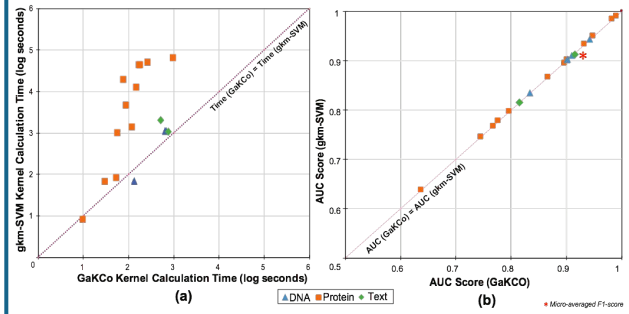
6. Theoretical Analysis

	GaKCo	gkm-SVM
Pre-processing	c_{gk} gNI	gNI + ηug
Kernel updates	c_{gk} zN ^{2}	ηuN ^{2}



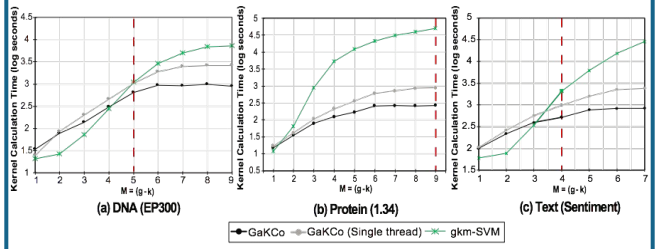
7. Experiments and Results

Data: We perform 19 different classification tasks to evaluate the performance of GaKCo. These tasks belong to three categories: (1) Transcription Factor (TF) binding site prediction (DNA dataset), (2) Remote Protein Homology prediction (protein dataset), and (3) Character based English text classification (text dataset).



Comparisons:

- (a). Kernel Calculation times (log(seconds)) of GaKCo(X-axis) vs gkm-SVM(Y-axis).*GaKCo is faster for 16/19 datasets.*
- (b). Empirical performance for the same 19 datasets of GaKCo (X-axis) versus gkm-SVM (Y-axis). *GaKCo achieves the same AUC-scores as gkm-SVM.*



Kernel calculation times with varying mismatches (M): scales well with increasing Σ and M.

References

1. Mahmoud Ghandi, Dongwon Lee, Morteza Mohammad-Noori, and Michael A Beer. Enhanced regulatory sequence prediction using gapped k-mer features. PLoS Comput Biol, 10(7):e1003711, 2014.
2. Mahmoud Ghandi, Morteza Mohammad-Noori, and Michael A Beer. Robust k-mer frequency estimation using gapped k-mers. Journal of mathematical biology, 69(2):469–500, 2014