Bioinformatics, 32(14), 2016, 2196–2198
doi: 10.1093/bioinformatics/btw142
Advance Access Publication Date: 15 March 2016
Applications Note

OXFORD

Sequence analysis

LS-GKM: a new gkm-SVM for large-scale datasets

Dongwon Lee

McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University, Baltimore, MD 21205, USA

Associate Editor: John Hancock

Received on December 10, 2015; revised on February 23, 2016; accepted on March 9, 2016

Abstract

Summary: gkm-SVM is a sequence-based method for predicting and detecting the regulatory vocabulary encoded in functional DNA elements, and is a commonly used tool for studying gene regulatory mechanisms. Here we introduce new software, LS-GKM, which removes several limitations of our previous releases, enabling training on much larger scale (LS) datasets. LS-GKM also provides additional advanced gapped *k*-mer based kernel functions. With these improvements, LS-GKM achieves considerably higher accuracy than the original gkm-SVM.
Availability and implementation: C/C ++ source codes and related scripts are freely available from http://github.com/Dongwon-Lee/lsgkm/, and supported on Linux and Mac OS X.
Contact: dwlee@jhu.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

We have previously introduced a sequence-based method, kmer-SVM (Fletez-Brant *et al.*, 2013; Lee *et al.*, 2011) to predict regulatory elements from DNA sequence and epigenetic data using support vector machines (SVM) (Vapnik, 1995). It has been successfully applied to studies of regulatory elements in different cellular contexts (Gorkin *et al.*, 2012; Pimkin *et al.*, 2014), and further improved by using gapped *k*-mers as new features (gkm-SVM (Ghandi *et al.*, 2014)). We have also recently demonstrated its ability to predict regulatory sequence variants (Lee *et al.*, 2015). Since then, gkm-SVM has gained increasing attention (Setty and Leslie, 2015; Zhou and Troyanskaya, 2015). Our general strategy is to build an SVM classifier that distinguishes regulatory sequences from non-regulatory genomic sequences in the *k*-mer or gapped *k*-mer frequency feature vector space. Training of SVM involves evaluation of a kernel matrix (or Gram matrix), defined as an *n*-by-*n* matrix of all possible inner products (or kernel functions) between a set of vectors of *n* training examples. However, as *n* increases, direct computation of the kernel matrix quickly becomes impractical with gapped *k*-mers as features. To resolve this issue, gkm-SVM employs an efficient algorithm that calculates a full kernel matrix with a runtime that linearly scales with n instead of n^2. An SVM classifier is then trained using the pre-calculated kernel matrix and standard SVM training methods. Yet, the full kernel matrix evaluation required in the original implementation has hindered optimal training of gkm-SVM on larger datasets because it needs substantial memory resources proportional to n^2. Sub-sampling strategies to circumvent this issue can be helpful (Ghandi *et al.*, 2014), but training on smaller datasets may yield sub-optimal SVM classifiers.

To tackle this problem, I have developed an improved software, LS-GKM, by implementing gapped *k*-mer kernel (gkm-kernel) functions within the LIBSVM framework (Chang and Lin, 2011). Most SVM tools such as LIBSVM utilize decomposition methods for SVM training. It iteratively finds and solves a small subset SVM problem that only needs a partial kernel matrix. For example, LIBSVM evaluates just two columns of the kernel matrix in each of the problem solving steps in its sequential minimal optimization algorithm (Fan *et al.*, 2005). Therefore, replacing the LIBSVM kernel routines with the gkm-kernel functions can essentially solve the memory resource issue and, consequently, allows us to train SVM on much larger datasets. To this end, I adopted and modified the original gkm-kernel algorithm, substituting for the original LIBSVM kernels so that it can efficiently evaluate one column of the kernel matrix in the same manner as the original gkm-SVM does for the full matrix. Multi-thread functionality is also implemented in LS-GKM for further speed-up (Supplementary Methods for more details).

I first compared runtime and memory usage of the new software to the original gkm-SVM by varying the training set size *n* (Supplementary Methods, Fig. S1). As expected, our previous method exhibits quadratic growth of memory usage as *n* increases. *gkmtrain* (the new SVM training module of LS-GKM) with the large

cache (8 Gb) also exhibits quadratic memory expansion when $n < 60\,000$. However, the overall memory usage is much less than the original method (~20%). Moreover, once the cache is full, the memory only linearly increases. Regarding runtime, the original method initially shows better computational efficiency than *gkmtrain* with the default setting (1 thread, 100 Mb). However, with either the large cache or the four threads, the new program can run faster than the original one. Furthermore, it can run almost 5× faster when both options are used. Most notably, we can now regularly train LS-GKM on much larger datasets with reasonable time and memory.

In addition to the integration of kernel functions into LIBSVM, three new kernel functions have been developed. Analogous to the basic RBF kernel, *gkmrbf-kernel* is defined as the radial basis function, in the space of gapped *k*-mer frequency vectors (Supplementary Methods). The second option, denoted as center weighted gkm-kernel or *wgkm-kernel*, is inspired by the observation that most ChIP-seq and DNaseI-seq signals are concentrated in the central regions within peaks. In this new kernel, the gapped *k*-mers are differentially weighted based on their distances from the center of the peak (Supplementary Methods, Fig. S1). The last kernel, *wgkmrbf-kernel*, is the combination of the previous two.

To demonstrate the utility of LS-GKM, I assessed how much classification accuracy can be improved by LS-GKM for predicting regulatory elements. Uniformly processed 322 ENCODE ChIP-seq datasets (The ENCODE Project Consortium, 2012) containing at least 5000 regions were considered, and standard training and test procedures developed in the previous studies (Ghandi *et al.*, 2014; Lee *et al.*, 2011) were applied with some modifications (Supplementary Methods). First, the new models trained on the whole datasets exhibit considerably better AUC than the models trained on the sub-sampled sets ($n = 10\,000$), especially when the training set is large ($n > 60\,000$) (Supplementary Methods, Fig.S2A).

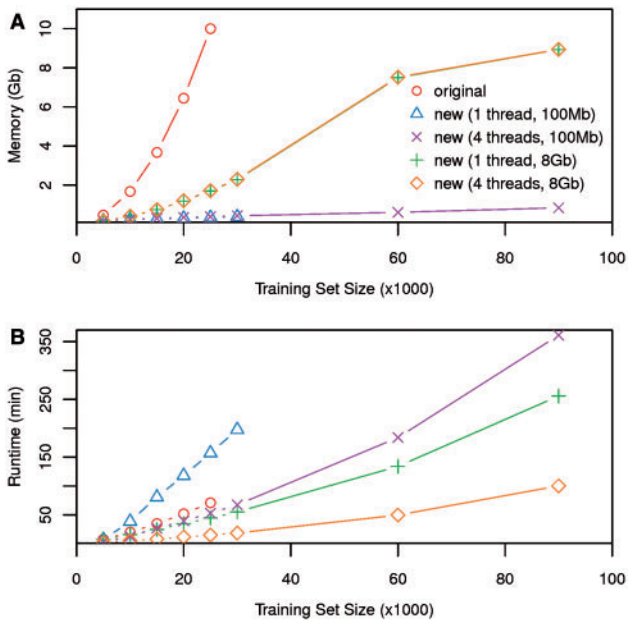

Fig. 1. LS-GKM is highly scalable in comparison to gkm-SVM. The maximum memory usage (**A**) and runtimes (**B**) are compared as training set sizes vary. Eight sub-sampled sets from H1hescCtcf (5k, 10k, 15k, 20k, 25k, 30k, 60k and 90k) were evaluated. All SVM trainings were done on a single machine equipped with Intel Core i7 processor (2.8 GHz, 4 cores) and 16 GB RAM for equivalent comparison

A grid search of the *C* parameter on selected datasets confirms that this result is not an artifact caused by a sub-optimal choice of *C* (Supplementary Table S1). In fact, our default value ($C = 1$) was optimal or near optimal in almost all cases we tested. Second, the model trained with *gkmrbf-kernel* further increases the AUC as compared to the original gkm-kernel in every case, but the improvement is marginal (Supplementary Methods, Fig.S2B). This result implies that the advantage of using non-linear decision boundaries is limited with gkm-kernel. Third, *wgkm-kernel* can also significantly improve the AUC, when the datasets already exhibit AUC values >0.9 (Supplementary Methods, Fig.S2C). Closer investigation reveals that most of the less predictive datasets (AUC < 0.9) are ChIP-seq on Pol2 and its related factors (TAF1, TAF7 and TBP). This suggests that many of these peaks may represent transient binding of the factors and, thus, contain less predictive sequence features. Fourth, similar to the *gkmrbf-kernel*, *wgkmrbf-kernel* marginally improves AUC when compared to *wgkm-kernel* (Supplementary Methods, Fig.S2D). Note that, in some cases such as Pol2 ChIP-seq, the best AUCs are achieved by *gkmrbf-kernel* not by *wgkmrbf-kernel*. Therefore, for the final comparison, the better kernels were chosen based on classification performance with independent training and evaluation (Supplementary Methods, Fig. S3). Supplementary Figure S4 compares the baseline AUCs (trained on 10 000 regions with gkm-kernel) from the original gkm-SVM to the best AUCs achieved by LS-GKM. The average gain of AUC is significant (0.912 → 0.941). If Pol2 and the related ChIP-seq datasets are removed, the average of the best AUCs is remarkably high, 0.960.

To determine whether LS-GKM can also improve deltaSVM, a major application of gkm-SVM for predicting regulatory sequence variants (Lee *et al.*, 2015), the dsQTL test set was reanalyzed using the new LS-GKM models trained on a larger GM12878 DHS dataset (Supplementary Methods, Fig. S5). The Precision Recall curves show that the new models consistently outperform the original model. However, no further improvement is achieved with new kernels, suggesting that larger datasets primarily contribute to the improvement of deltaSVM accuracy.

In this study, I have presented new and improved software, LS-GKM, which offers several new functionalities and considerably improves the classification accuracy on predicting regulatory elements. We strongly encourage all users of our software to train models on the largest datasets available, which can produce significantly more accurate predictions. In this regard LS-GKM should improve the performance considerably, and in combination with its enhanced functions, LS-GKM is expected to significantly contribute to our understanding of gene regulation.

Acknowledgements

The author thanks Dr. Aravinda Chakravarti for his support and helpful discussions. The author also thanks Dr. Michael A. Beer, and Dr. Mahmoud Ghandi for providing useful comments. The author thanks The Maryland Advanced Research Computing Center (MARCC) for use of its computing resources.

Funding

This work has been supported by NIH grants HL128782, HL086694 and GM104469.

Conflict of Interest: none declared.

References

Chang,C.C. and Lin,C.J. (2011) LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, **2**, 27. 1–27:27.

Fan,R.E. *et al*. (2005) Working set selection using second order information for training support vector machines. *J. Mach. Learn. Res.*, **6**, 1889–1918.

Fletez-Brant,C. *et al*. (2013) kmer-SVM: a web server for identifying predictive regulatory sequence features in genomic data sets. *Nucleic Acids Res.*, **41**, W544–W556.

Ghandi,M. *et al*. (2014) Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS Comput. Biol.*, **10**, e1003711.

Gorkin,D.U. *et al*. (2012) Integration of ChIP-seq and machine learning reveals enhancers and a predictive regulatory sequence vocabulary in melanocytes. *Genome Res.*, **22**, 2290–2301.

Lee,D. *et al*. (2011) Discriminative prediction of mammalian enhancers from DNA sequence. *Genome Res.*, **21**, 2167–2180.

Lee,D. *et al*. (2015) A method to predict the impact of regulatory variants from DNA sequence. *Nat. Genet.*, **47**, 955–961.

Pimkin,M. *et al*. (2014) Divergent functions of hematopoietic transcription factors in lineage priming and differentiation during erythro-megakaryopoiesis. *Genome Res.*, **24**, 1932–1944.

Setty,M. and Leslie,C.S. (2015) SeqGL identifies context-dependent binding signals in genome-wide regulatory element maps. *PLoS Comput. Biol.*, **11**, e1004271.

The ENCODE Project Consortium. (2012) An integrated encyclopedia of DNA elements in the human genome. *Nature*, **489**, 57–74.

Vapnik, V.N. (1995) *The Nature of Statistical Learning Theory*. Springer, New York.

Zhou,J. and Troyanskaya,O.G. (2015) Predicting effects of noncoding variants with deep learning-based sequence model. *Nat. Methods*, **12**, 931–934.