An Empirical Analysis of Domain Adaptation Algorithms for Genomic Sequence Analysis

Authored by:

Bernhard Sch?lkopf
Gunnar R?tsch
Gabriele Schweikert
Christian Widmer

Abstract

We study the problem of domain transfer for a supervised classification task in mRNA splicing. We consider a number of recent domain transfer methods from machine learning, including some that are novel, and evaluate them on genomic sequence data from model organisms of varying evolutionary distance. We find that in cases where the organisms are not closely related, the use of domain adaptation methods can help improve classification performance.

1 Paper Body

Ten years ago, an eight-year lasting collaborative effort resulted in the first completely sequenced genome of a multi-cellular organism, the free-living nematode Caenorhabditis elegans. Today, a decade after the accomplishment of this landmark, 23 eukaryotic genomes have been completed and more than 400 are underway. The genomic sequence builds the basis for a large body of research on understanding the biochemical processes in these organisms. Typically, the more closely related the organisms are, the more similar the biochemical processes. It is the hope of biological research that by analyzing a wide spectrum of model organisms, one can approach an understanding of the full biological complexity. For some organisms, certain biochemical experiments can be performed more readily than for others, facilitating the analysis of particular processes. This understanding can then be transferred to other organisms, for instance by verifying or refining models of the processes?at a fraction of the original cost. This is but one example of a situation where transfer of knowledge across domains is fruitful. In machine learning, the above information transfer is called domain adaptation, where one aims to use data or a model of a well-analyzed source domain to obtain or refine a model for a less analyzed target domain. For supervised classification, this corresponds to the case where there are ample labeled examples (xi , yi), i = 1, . . . , m for the source domain, but only few such examples (xi , yi), i = m + 1, . . . , m + n for the target domain (n ? m). The examples are assumed to be drawn independently from the joint probability distributions PS (X, Y) and PT (X, Y), respectively. The distributions PS (X, Y) = PS (Y —X) ? PS (X) and PT (X, Y) = PT (Y —X) ? PT (X) can differ in several ways: (1) In the classical covariate shift case, it is assumed that only the distributions of the input features P (X) varies between the two domains: PS (X) 6= PT (X). The conditional, however, remains 1

These authors contributed equally.

invariant, PS (Y —X) = PT (Y —X). For a given feature vector x the label y is thus independent of the domain from which the example stems. An example thereof would be if a function of some biological material is conserved between two organisms, but its composition has changed (e.g. a part of a chromosome has been duplicated). (2) In a more difficult scenario the conditionals differ between domains, PS (Y —X) 6= PT (Y —X), while P (X) may or may not vary. This is the more common case in biology. Here, two organisms may have evolved from a common ancestor and a certain biological function may have changed due to evolutionary pressures. The evolutionary distance may be a good indicator for how well the function is conserved. If this distance is small, we have reason to believe that the conditionals may not be completely different, and knowledge of one of them should then provide us with some information also about the other one. While such knowledge transfer is crucial for biology, and performed by biologists on a daily basis, surprisingly little work has been done to exploit it using machine learning methods on biological databases. The present paper attempts to fill this gap by studying a realistic biological domain transfer problem, taking into account several of the relevant dimensions in a common experimental framework: ? methods ? over the last years, the field of machine learning has seen a strong increase in interest in the domain adaptation problem, reflected for instance by a recent NIPS workshop ? domain distance ? ranging from close organisms, where simply combining training sets does the job, to distant organisms where more sophisticated methods can potentially show their strengths ? data set sizes ? whether or not it is worth transferring knowledge from a distant organism is expected to depend on the amount of data available for the target system With the above in mind, we selected the problem of mRNA splicing (see Figure A1 in the Appendix for more details) to assay the above dimensions of domain adaptation on a task which is relevant to modern biology. The paper is organized as follows: In Section 2, we will describe the experimental design including the datasets, the underlying classification model, and the model selection and evaluation procedure. In Section 3 we will briefly review a number of known algorithms for domain adaptation, and propose certain variations. In Section 4 we show the results of our comparison with a brief discussion.

2

Experimental Design

2.1 A Family of Classification Problems We consider the task of identifying so-called acceptor splice sites within a large set of potential splice sites based on a sequence window around a site. The idea is to consider the recognition of splice sites in different organisms: In all cases, we used the very well studied model organism C. elegans as the source domain. As target organisms we chose two additional nematodes, namely, the close relative C. remanei, which diverged from C. elegans 100 million years ago [10], and the more distantly related P. pacificus, a lineage which has diverged from C. elegans more than 200 million years ago [7]. As a third target organism we used D. melanogaster, which is separated from C. elegans by 990 million years [11]. Finally, we consider the plant A. thaliana, which has diverged from the other organisms more than 1,600 million years ago. It is assumed that a larger evolutionary distance will likely also have led to an accumulation of functional differences in the molecular splicing machinery. We therefore expect that the differences of classification functions for recognizing splice sites in these organisms will increase with increasing evolutionary distance. 2.2 The Classification Model It has been demonstrated that Support Vector Machines (SVMs) [1] are well suited for the task of splice site predictions across a wide range of organisms [9]. In this work, the so-called Weighted Degree kernel has been used to measure the similarity between two example sequences x and x? of

fixed length L by counting co-occurring substrings in both sequences at the same position: L?l+1 ?

1 X X k?wd (x, x?) = ?d I x[l:l+d] = x?[l:l+d] L l=1

(1)

d=1

where x[l:l+d] is the substring of length d of x at position l and ?d = 2 ??d+1 ?2 +? is the weighting of the substring lengths. In our previous study we have used sequences of length L = 140 and substrings of length ? = 22 for splice site detection [9]. With the four-letter DNA sequence alphabet {A, C, G, T } this leads to a very high dimensional feature space (¿ 1013 dimensions). Moreover, to archive the best classification performance, a large number of training examples is very helpful ([9] used up to 10 million examples). For the designed experimental comparison we had to run all algorithms many times for different training set sizes, organisms and model parameters. We chose the source and target training set as large as possible?in our case at most 100,000 examples per domain. Moreover, not for all algorithms we had efficient implementations available that can make use of kernels. Hence, in order to perform this study and to obtain comparable results, we had to restrict ourselves to a case were we can explicitly work in the feature space, if necessary (i.e. ? not much larger than two). We chose ? = 1. Note, that this choice does not limit the generality of this study, as there is no strong reason, why efficient implementations that employ kernels could not be developed for all methods. The development of large scale methods, however, was not the main focus of this study. Note that the above choices required an equivalent of about 1500 days of computing time on state-ofthe-art CPU cores. We therefore refrained from including more methods, examples or dimensions. 2.3 Splits and Model Selection In the first set of experiments we randomly selected a source dataset of 100,000 examples from C. elegans, while data sets of sizes 2,500, 6,500, 16,000, 40,000 and 100,000 were selected for each target organism. Subsequently we performed a second set of experiments where we combined several sources. For our comparison we used 25,000 labeled examples from each of four remaining organisms to predict on a target organism. We ensured that the positives to negatives ratio is at 1/100 for all datasets. Two thirds of each target set were used for training, while one third was used for evaluation in the course of hyper-parameter tuning.1 Additionally, test sets of 60,000 examples were set aside for each target organism. All experiments were repeated three times with different training splits (source and target), except the last one which always used the full data set. Reported will be the average area under the precision-recall-curve (auPRC) and its standard deviation, which is considered a sensible measure for imbalanced classification problems. The data and additional information will be made available for download on a supplementary website.2

3 Methods for Domain Adaptation Regarding the distributional view that was presented in Section 1, the problem of splice site prediction can be affected by both evils simultaneously, namely PS (X) 6= PT (X) and PS (Y —X) 6= PT (Y —X), which is also the most realistic scenario in the case of modeling most biological processes. In this paper, we will therefore drop the classical covariate shift assumption, and allow for different predictive functions PS (Y —X) 6= PT (Y —X). 3.1 Baseline Methods (SVMS and SVMT) As baseline methods for the comparison we consider two methods: (a) training on the source data only (SVMS) and (b) training on the target data only (SVMT). For SVMS we use the source data for training however we tune the hyper-parameter on the available target data. For SVMT we use the available target data for training (67%) and model selection (33%). The resulting functions are fS (x) = h?(x), wS i + bS and fT (x) = h?(x), wT i + bT . 1 2

Details on the hyper-parameter settings and tuning are shown in Table A2 in the appendix. http://www.fml.mpg.de/raetsch/projects/genomedomainadaptation

3.2 Convex Combination (SVMS +SVMT) The most straightforward idea for domain adaptation is to reuse the two optimal functions fT and fS as generated by the base line methods SVMS and SVMT and combine them in a convex manner: F (x) = ?fT (x) + (1 ? ?)fS (x). Here, ? ? [0, 1] is the convex combination parameter that is tuned on the evaluation set (33%) of the target domain. A great benefit of this approach is its efficiency. 3.3 Weighted Combination (SVMS+T) Another simple idea is to train the method on the union of source and target data. The relative importance of each domain is integrated into the loss term of the SVM and can be adjusted by setting domain-dependent cost parameters CS and CT for the m and n training examples from the source and target domain, respectively: min w,?

s.t.

m m+n X X 1 2 kwk + CS ?i + CT ?i 2 i=1 i=m+1

yi (hw, ?(xi)i + b) ? 1 ? ?i ?i ? 0 ?i ? [1, m + n]

(2)

?i ? [1, m + n]

This method has two model parameters and requires training on the union of the training sets. Since the computation time of most classification methods increases super-linearly and full model selection may require to train many parameter combinations, this approach is computationally quite demanding. Dual-task Learning (SVMS,T)

3.4

One way of extending the weighted combination approach is a variant of multi-task learning [2]. The idea is to solve the source and target classification problems simultaneously and couple the two solutions via a regularization term. This idea can be realized by the following optimization problem: min

wS ,wT ,?

s.t.

m+n X 1 ?i kwS ? wT k2 + C 2 i=1

yi (hwS , ?(xi)i + b) ? 1 ? ?i yi (hwT , ?(xi)i + b) ? 1 ? ?i ?i ? 0

(3) ?i ? 1, . . . , m ?i ? m + 1, . . . , m + n ?i ? 1, . . . , m + n

Please note that now wS and wT are optimized. The above optimization problem can be solved using a standard QP-solver. In a preliminary experiment we used the optimization package CPLEX to solve this problem, which took too long as the number of variables is relatively large. Hence, we decided to approximate the soft-margin loss using the logistic loss l(f (x), y) = log(1+exp(?yf (x))) and to use a conjugate gradient method3 to minimize the resulting objective function in terms of wS and wT . 3.5 Kernel Mean Matching (SVMS?T) Kernel methods map the data into a reproducing kernel Hilbert space (RKHS) by means of a mapping ? : X ? H related to a positive definite kernel via k(x, x?) = h?(x), ?(x?)i. Depending on the choice of kernel, the space of H may be spanned by a large number of higher order features of the data. In such cases, higher order statistics for a set of input points can be computed in H by simply taking the mean (i.e., the first order statistics). In fact, it turns out that for a certain class of kernels, the mapping n 1X ? : (x1 , . . . , xn) 7? ?(xi) n i=1 3

We used Carl Rasmussen?s minimize function.

is injective [5] ? in other words, given knowledge of (only) the mean (the right hand side), we can completely reconstruct the set of points. For a characterization of this class of kernels, see for instance [4]. It is often not necessary to retain all information (indeed, it may be useful to specify which information we want to retain and which one we want to disregard, see [8]). Generally speaking, the higher dimensional H, the more information is contained in the mean. In [6] it was proposed that one could use this for covariate shift adaptation, moving the mean of a source distribution (over the inputs only) towards the mean of a target distribution by re-weighting the source training points. We have applied this to our problem, but found that a variant of this approach performed better. In this variant, we do not re-weight the source points, but rather we translate each point towards the mean of the target inputs: ! m m+n X X 1 1 ? j) = ?(xj) ? ? ?(x ?(xi) ? ?(xi) ?j = 1, . . . , m. m i=1 n i=m+1

This also leads to a modified source input distribution which is statistically more similar to the target distribution and which can thus be used to improve performance when training the target task. Unlike [6], we do have a certain amount of labels also for the target distribution. We make use of them by performing the shift separately for each class y ? {?1}: ! m m+n X X 1 1 ? j) = ?(xj) ? ? ?(x [[yi = y]]?(xi) ? [[yi = y]]?(xi) my i=1 ny i=m+1 for all j = m + 1, . . . , m + n with yj = y, where my and ny are the number of source and target examples with label y, respectively. The shifted examples can now be used in different ways to obtain a final classifier. We decided to use the weighted combination with CS = CT for comparison. 3.6 Feature Augmentation (SVMS?T)

In [3] a method was proposed that augments the features of source and target examples in a domainspecific way: ? ?(x) = (?(x), ?(x), 0)? for i = 1, . . . , m ? ?(x) = (?(x), 0, ?(x))? for i = m + 1, . . . , m + n. The intuition behind this idea is that there exist one set of parameters that models the properties common to both sets and two additional sets of parameters that model the specifics of the two domains. It can easily be seen that the kernel for the augmented feature space can be computed as:

2h?(xi), ?(xj)i if [[i ? m]] = [[j ? m]] kAU G (xi , xi) = h?(xi), ?(xj)i otherwise This means that the ?similarity? between two examples is two times as high, if the examples were drawn from the same domain, as if they were drawn from different domains. Instead of the factor 2, we used a hyper-parameter B in the following. 3.7 Combination of Several Sources Most of the above algorithms can be extended in one way or another to integrate several source domains. In this work we consider only three possible algorithms: (a) convex combinations of several domains, (b) KMM on several domains and (c) an extension of the dual-task learning approach to multi-task learning. We briefly describe these methods below: Multiple Convex Combinations (M-SVMS +SVMT) The most general version would be to optimize all convex combination coefficients independently. If done in a grid-search-like manner, it becomes prohibitive for more than say three source domains. In principle, one can optimize these coefficients also by solving a linear program. In preliminary experiments we tried both approaches and they typically did not lead to better results than the following combination: 1 X F (x) = ?fT (x) + (1 ? ?) fS (x), —S— S?S

where S is the set of all considered source domains. We therefore only considered this way of combining the predictions.

Multiple KMM (M-SVMS?T) Here, we shift the source examples of each domain independently towards the target examples, but by the same relative distance (?). Then we train one classifier on the shifted source examples as well as the target examples. Multi-task Learning (M-SVMS,T) We consider the following version of multi-task learning: min

{wD }D?D ,?

s.t.

X 1 X X ?D1 ,D2 kwD1 ? wD2 k2 + ?i 2 i

(4)

D1 ?D D2 ?D

yi (hwDj , ?(xi)i + b) ? 1 ? ?i ?i ? 0

(5)

for all examples (xi , yi) in domain Dj ? D, where D is the set of all considered domains. ? is a set of regularization parameters, which we parametrized by two parameters CS and CT in the following way: ?D1 ,D2 = CS if D1 and D2 are source domains and CT otherwise.

4
Experimental Results
We considered two different settings for the comparison. For the first experiment we assume that there is one source domain with enough data that should be used to improve the performance in the target domain. In the second setting we analyze whether one can benefit from several source domains. 4.1 Single Source Domain Due to space constraints, we restrict ourselves to presenting a summary of our results with a focus on best and worst performing methods. The detailed results are given in Figure A2 in the appendix, where we show the median auPRC of the methods SVMT , SVMS , SVMS?T , SVMS+T , SVMS +SVMT , SVMS?T and SVMS,T for the considered tasks. The summary is given in Figure 1, where we ? illustrate which method performed best (green), similarly well (within a confidence interval of ?/ n) as the best (light green), considerably worse than the best (yellow), not significantly better than the worst (light red) or worst (red). From these results we can make the following observations: 1. Independent of the task, if there is very little target data available, the training on source data performs much better than training on the target data. Conversely, if there is much target data available then training on it easily outperforms training the source data. 2. For a larger evolutionary distance of the target organisms to source organism C. elegans, a relatively small number of target training examples for the SVMT approach is sufficient to achieve similar performance to the SVMS approach, which is always trained on 100,000 examples. We call the number of target examples with equal source and target performance the break-even point. For instance, for the closely related organism C. remanei one needs nearly as many target data as source data to achieve the same performance. For the most distantly related organism A. thaliana, less than 10% target data is sufficient to outperform the source model. 3. In almost all cases, the performance of domain adaption algorithms is considerably higher than source (SVMS) and target only (SVMT). This is most pronounced near the break-even point, e.g. 3% improvement for C. remanei and 14% for D. melanogaster. 4. Among the domain adaptation algorithms, the dual-task learning approach (SVMS,T) performed most often best (12/20 cases). Second most often best (5/20) performed the convex combination approach (SVMS +SVMT). From our observations we can conclude that the simple convex combination approach works surprisingly well. It is only outperformed by the dual-task learning algorithm which performs consistently well for all organisms and target training set sizes.

2 References

NA