A Structured-Outputs Method for Prediction of Protein Function

Artem Sokolov SOKOLOV@CS.COLOSTATE.EDU
Asa Ben-Hur ASA@CS.COLOSTATE.EDU
Colorado State University, Fort Collins, Colorado, 80523

Abstract

We apply the structured-output methodology to the problem of predicting the molecular function of proteins. Our results demonstrate that learning the structure of the output space yields better performance when compared to the traditional "transfer of annotation" method.

1. Introduction

We address the problem of automatic annotation of protein function using structured output methods. The function of a protein is defined by a set of keywords that specify its molecular function, its role in the biological process and its localization to a cellular component. The Gene Ontology (GO) imposes a hierarchy over the keywords and is considered the current standard for annotating gene products and proteins (Gene Ontology Consortium, 2000).

Computational methods for annotating protein function have been predominantly following the "transfer of annotation" paradigm where GO keywords are transferred from one protein to another based on the sequence similarity between the two. This is generally done by employing a sequence alignment tool such as BLAST (Altschul et al., 1990) to find annotated proteins that have a high level of sequence similarity to the un-annotated query protein. Such variations on the nearest-neighbor methodology suffer from serious limitations in that they fail to exploit the inherent structure of the annotation space. Furthermore, annotation transfer of multiple GO keywords between proteins is not always appropriate, e.g. in the case of multi-domain proteins (Galperin & Koonin, 1998).

Since proteins can have multiple functions, and those functions are described by a hierarchy of keywords, we formulate prediction of protein function as a hierarchical multi-label classification problem and apply structured output prediction methods to it. This work focuses on the structured-perceptron which we use as an alternative to the BLAST nearest-neighbor methodology. Empirical results demonstrate that learning the structure of the output space yields improved performance over transfer of annotation. In our experiments we use BLAST to define the input space features as well as to limit the output space during inference. We demonstrate that failure to limit the output space can be detrimental to the prediction accuracy. In future work we will explore the use of more sophisticated methods of structured output prediction, such as maximum margin classifiers (Tsochantaridis et al., 2005; Rousu et al., 2006).

2. Methods

Prediction of protein function can be formulated as a hierarchical multi-label classification problem as follows. Each protein is annotated with a macro-label $\mathbf{y} = (y_1, y_2, ..., y_k) \in \{0,1\}^k$, where each micro-label y_i corresponds to one of the k nodes that belong to the hierarchy defined by the Gene Ontology. The micro-labels take on the value of 1 when the protein performs the function defined by the corresponding node. Whenever a protein is associated with a particular micro-label, we also associate it with all its ancestors in the hierarchy, i.e. given a specific term, we associate with it all terms that generalize it. Note that the Gene Ontology consists of three distinct hierarchies: molecular function, biological process and cellular component. In this work we focus on the molecular function hierarchy.

We train a linear classifier to predict the molecular function of proteins. Given a protein characterized by $\mathbf{x}$ in the input feature space $\mathcal{X}$, we make inference for the most likely label according to:

$$\hat{\mathbf{y}} = h(\mathbf{x}) = \arg\max_{\mathbf{y}\in\mathcal{Y}} f(\mathbf{x}, \mathbf{y}|\mathbf{w})$$

where $\mathcal{Y}$ is the set of possible macro-labels we are willing to consider. The function $f(\mathbf{x}, \mathbf{y}|\mathbf{w}) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ can be thought of as a compatibility measure between an input $\mathbf{x}$ and an output macro-label $\mathbf{y}$. We assume the function is linear in $\mathbf{w}$, i.e. $f(\mathbf{x}, \mathbf{y}|\mathbf{w}) = \mathbf{w}^T\phi(\mathbf{x}, \mathbf{y})$ in some space defined by the mapping ϕ.

We train the classifier using a variant of the perceptron algorithm generalized for structured outputs (Collins,

Algorithm 1 Perceptron for Structured Outputs

Input: training data $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$
Output: parameters $\alpha_{i,\mathbf{y}}$ for $i = 1, ..., n$ and $\mathbf{y} \in \mathcal{Y}$.
Initialize: $\alpha_{i,\mathbf{y}} = 0 \quad \forall i, \mathbf{y}$.
repeat
  for $i = 1$ **to** n **do**
    Compute the top two scoring labels:
    $\hat{\mathbf{y}} \leftarrow \arg\max_{\mathbf{y} \in \mathcal{Y}} f(\mathbf{x}_i, \mathbf{y}|\alpha)$
    $\bar{\mathbf{y}} \leftarrow \arg\max_{\mathbf{y} \in \mathcal{Y} \setminus \hat{\mathbf{y}}} f(\mathbf{x}_i, \mathbf{y}|\alpha)$
    if $\hat{\mathbf{y}} \neq \mathbf{y}_i$ **then**
      Handle misclassification:
      $\alpha_{i,\mathbf{y}_i} \leftarrow \alpha_{i,\mathbf{y}_i} + 1$
      $\alpha_{i,\hat{\mathbf{y}}} \leftarrow \alpha_{i,\hat{\mathbf{y}}} - 1$
    else if $f(\mathbf{x}_i, \mathbf{y}_i) - f(\mathbf{x}_i, \bar{\mathbf{y}}) < \gamma$ **then**
      Handle margin violation:
      $\alpha_{i,\mathbf{y}_i} \leftarrow \alpha_{i,\mathbf{y}_i} + 1$
      $\alpha_{i,\bar{\mathbf{y}}} \leftarrow \alpha_{i,\bar{\mathbf{y}}} - 1$
    end if
  end for
until a terminating criterion is met

2002). Given a set of n training examples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, the algorithm attempts to find the vector $\mathbf{w}$ such that the decision function values for the correct output and the best runner-up are separated by the user-defined margin γ:

$$\mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}_i) - \max_{\mathbf{y} \in \mathcal{Y} \setminus \mathbf{y}_i} \mathbf{w}^T \phi(\mathbf{x}_i, \mathbf{y}) > \gamma \quad \forall i.$$

To make use of kernels, we assume that the weight vector $\mathbf{w}$ can be expressed as a linear combination of the training examples:

$$\mathbf{w} = \sum_{j=1}^{n} \sum_{\mathbf{y}' \in \mathcal{Y}} \alpha_{j,\mathbf{y}'} \phi(\mathbf{x}_j, \mathbf{y}').$$

This leads to reparameterization of the decision function in terms of the α coefficients:

$$f(\mathbf{x}, \mathbf{y}|\alpha) = \sum_{j=1}^{n} \sum_{\mathbf{y}' \in \mathcal{Y}} \alpha_{j,\mathbf{y}'} K((\mathbf{x}_j, \mathbf{y}'), (\mathbf{x}, \mathbf{y}))$$

where $K : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \rightarrow \mathbb{R}$ is the joint kernel defined over the input-output space. In this work, we take the joint kernel to be the product of the input space and the output space kernels: $K((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) = K_{\mathcal{X}}(\mathbf{x}, \mathbf{x}') K_{\mathcal{Y}}(\mathbf{y}, \mathbf{y}')$. For the output-space kernel, $K_{\mathcal{Y}}$, we use a linear kernel; the input-space kernel is described below.

The general routine for learning the coefficients α is presented in Algorithm 1. In our application, the terminating criterion is taken to be a limit on the number of iterations.

3. Experimental Results

We propose a loss function we call the *kernel loss* and argue for its use in hierarchical classification problems since it generalizes F-measure used in information retrieval (van Rijsbergen, 1979). Details will be provided elsewhere.

$$\Delta(\mathbf{y}, \hat{\mathbf{y}}) = 1 - \frac{K_{\mathcal{Y}}(\mathbf{y}, \hat{\mathbf{y}})}{\sqrt{K_{\mathcal{Y}}(\mathbf{y}, \mathbf{y}) K_{\mathcal{Y}}(\hat{\mathbf{y}}, \hat{\mathbf{y}})}} = 1 - \frac{\mathbf{y}^T \hat{\mathbf{y}}}{\sqrt{\mathbf{y}^T \mathbf{y} \cdot \hat{\mathbf{y}}^T \hat{\mathbf{y}}}}$$

We used the data from the following four species: *C. elegans*, *D. melanogaster*, *S. cerevisiae* and *S. pombe*. Our experiments followed the leave-one-species-out paradigm, where we withheld one species for testing and trained the perceptron on the remaining data, rotating which species got withheld. This variant of cross-validation simulates the situation of annotating a newly-sequenced genome (Vinayagam et al., 2004).

Prior to making predictions, we ran the data through several steps of preprocessing. First, we removed all annotations that were discovered through computational means as these were generally inferred by sequence or structure similarity and would introduce bias into any classifier that used sequence similarity to make a prediction. Second, we expanded the set of annotations associated with a protein to include all ancestor nodes of the nodes it was annotated with; for simplicity we considered a subset of the GO hierarchy called GO-slims. We then ran BLAST for each of the proteins in our dataset against all four species, removing the hits where the protein was aligned to itself.

We employed the nearest neighbor BLAST methodology as our baseline. For every test protein, we transferred the annotations from the most significant hit against a protein from another species. Proteins with e-values above 10^{-6} were not considered in our experiments.

The structured-output perceptron is provided exactly the same data as the BLAST method. The input-space kernel is an empirical kernel map that uses the negative-log of the BLAST e-values that are below 50, where the features were normalized to have values less than 1.0 and the input vectors are normalized to be unit vectors.

The inference during training was limited to only those macro-labels that appear in the training dataset. We call this space $\mathcal{Y}_1$. For inference of test sample labels we considered three different output spaces, $\mathcal{Y}_1, \mathcal{Y}_2, \mathcal{Y}_3$, in order to examine the effect of the size of the search space on prediction accuracy. We define $\mathcal{Y}_3(\mathbf{x})$ to be the set of macro-labels that appear in the significant BLAST hits of protein $\mathbf{x}$ (e-values below 10^{-6}). Ad-

Test on Fold size	*C. elegans* 844	*D. melanogaster* 1804	*S. cerevisiae* 1853	*S. pombe* 898	Output Space
BLAST NN	0.390(0.258)	0.278(0.264)	0.221(0.252)	0.223(0.240)	-
Perceptron	0.403(0.254)	0.280(0.262)	0.221(0.242)	0.255(0.243)	$\mathcal{Y}_1$
Perceptron	0.404(0.260)	0.265(0.271)	0.204(0.244)	0.221(0.243)	$\mathcal{Y}_2$
Perceptron	0.398(0.264)	0.263(0.271)	0.199(0.242)	0.222(0.243)	$\mathcal{Y}_3$
Random	0.507(0.217)	0.527(0.208)	0.529(0.200)	0.490(0.217)	-

Table 1. Empirical results comparing the performance of the traditional transfer-of-annotation method to the structured outputs approach. Presented are mean kernel loss per protein with the standard deviation values in parentheses. For comparison, we also include the performance of a random classifier that transfers annotation from a training example chosen uniformly at random.

ditionally, we define $\mathcal{Y}_2(\mathbf{x})$ to be the set of all subsets of macro-labels that can be obtained from the micro-labels in $\mathcal{Y}_3(\mathbf{x})$, with the constraint that each macro-label represents three leaf nodes of the hierarchy at the most. These label spaces satisfy: $\mathcal{Y}_3(\mathbf{x}) \subseteq \mathcal{Y}_2(\mathbf{x}) \subseteq \mathcal{Y}_1$.

The results are presented in Table 1. When the output label space is limited to $\mathcal{Y}_2$ or $\mathcal{Y}_3$ during testing, the structured perceptron algorithm outperforms the BLAST nearest-neighbor classifier. The larger label-space $\mathcal{Y}_1$, results in the inference procedure considering annotations that are irrelevant to the actual function of the test protein, which reduces the prediction accuracy. However, even in this case, the perceptron maintains competitive performance compared to the BLAST nearest-neighbor method. The results support our hypothesis that learning the structure of the output space is superior to simple transfer of annotations.

Note that the classifiers performed poorly when testing proteins from *C. elegans*. This is due to the fact that a vast majority of proteins in this species are annotated as protein binders (GOID:0005515). Such annotations contain little information from a biological standpoint and result in a skewed set of output labels. However, removing the species or the micro-label from the analysis lowers prediction accuracy suggesting that there is relevant information in the input space features captured by the dataset.

We have shown here that a structured output method performs better than a nearest neighbor method when provided with the same information. Our structured output method can be enhanced in several ways to further boost its performance: Additional information can easily be provided in the form of additional kernels on the input space that use other forms of genomic information (e.g. protein-protein interactions); the structured-perceptron can be replaced with maximum margin classifiers; and furthermore, semi-supervised learning can be used to leverage the abundance of available sequence information. In future work we will also consider larger datasets that include a larger number of species.

References

Altschul, S., Gish, W., Miller, W., Myers, E., & Lipman, D. (1990). Basic local alignment search tool. *J. Mol. Biol*, *215*, 403–410.

Collins, M. (2002). Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, 1–8.

Galperin, M. Y., & Koonin, E. V. (1998). Sources of systematic error in functional annotation of genomes: Domain rearrangement, non-orthologous gene displacement and operon disruption. *In Silico Biology*, *1*, 55–67.

Gene Ontology Consortium (2000). Gene ontology: tool for the unification of biology. *Nat. Genet.*, *25*, 25–9.

Rousu, J., Saunders, C., Szedmak, S., & Shawe-Taylor, J. (2006). Kernel-Based Learning of Hierarchical Multilabel Classification Models. *The Journal of Machine Learning Research*, *7*, 1601–1626.

Tsochantaridis, I., Joachims, T., Hofmann, T., & Altun, Y. (2005). Large Margin Methods for Structured and Interdependent Output Variables. *The Journal of Machine Learning Research*, *6*, 1453–1484.

van Rijsbergen, C. (1979). *Information retrieval*. London: Butterworths.

Vinayagam, A., K onig, R., Moormann, J., Schubert, F., Eils, R., Glatting, K.-H., & Suhai, S. (2004). Applying support vector machines for gene ontology based gene function prediction. *BMC Bioinformatics*, *5*, 178.