Propositionalisation of Profile Hidden Markov Models for Biological Sequence Analysis

Stefan Mutter, Bernhard Pfahringer, Geoffrey Holmes

Department of Computer Science
The University of Waikato
Hamilton, New Zealand
{mutter,bernhard,geoff}@cs.waikato.ac.nz

Abstract. Hidden Markov Models are a widely used generative model for analysing sequence data. A variant, Profile Hidden Markov Models are a special case used in Bioinformatics to represent, for example, protein families. In this paper we introduce a simple propositionalisation method for Profile Hidden Markov Models. The method allows the use of PHMMs discriminatively in a classification task. Previously, kernel approaches have been proposed to generate a discriminative description for an HMM, but require the explicit definition of a similarity measure for HMMs. Propositionalisation does not need such a measure and allows the use of any propositional learner including kernel-based approaches. We show empirically that using propositionalisation leads to higher accuracies in comparison with PHMMs on benchmark datasets.

1 Introduction

Traditionally, research in machine learning has focussed on fixed length attribute-value representations of data. Hence classifiers for this kind of data have undergone an intense optimisation process. In many areas, however, it is easier to represent data as structured terms. Examples include areas such as relational data mining or bioinformatics. In the field of classification from structured data [1], there are three common ways of dealing with this kind of representation: relational rule learning [2], kernel approaches [3, 4] or propositionalisation [5–7].

In this paper we use propositionalisation to generate attribute-value representations of varying complexity for Multiple Sequence Alignment. The Multiple Sequence Alignment is represented using a Profile Hidden Markov Model (PHMM) [8, 9]. A PHMM is a generative, probabilistic, graphical model. Using propositionalisation allows the use of complex PHMMs in a simple, discriminative way. Kernel methods have also been successfully applied to obtain a discriminative description of Hidden Markov Models (HMMs) [3, 10]. However, they require the definition of an explicit similarity measure between different HMMs. Propositionalisation approaches do not need such a pre-defined similarity measure.

The remainder of this paper is organised in the following way. Section 2 discusses Hidden Markov Models in general and their use in sequence classification. Section 3 describes kernel methods for structured classification emphasising

kernel methods for biological sequence data, whereas Section 4 explains propositionalisation approaches. In Section 5 we present our experimental setup and results. Conclusions and future work are presented in the final section.

2 Hidden Markov Models

Hidden Markov Models (HMM) are widely used for sequence analysis, especially in natural language processing and Bioinformatics [11, 9, 12]. In Bioinformatics HMMs have been successfully applied to gene-finding, phylogenetic analysis and protein secondary structure prediction. An HMM represents, in general, a probability distribution over sequences. It is a generative, probabilistic, graphical model.

Profile HMMs (PHMMs) are a special kind of HMM for representing Multiple Sequence Alignments. They allow an alignment to be trained from unaligned sequences. Figure 1 shows three unaligned amino acid sequences which can be used as input for a PHMM. This is exactly the kind of input data we use in our experiments.

```
MMFFADDAAAE
MMFFARRNSSTNNRREDPFMLWE
MMFFAE
```

Fig. 1. Three short, unaligned (artificially created) amino acid sequences.

A Multiple Sequence Alignment consists of three or more either DNA, RNA or amino acid sequences. In this paper we align amino acid sequences globally. PHMMs allow the construction and representation of a Multiple Sequence Alignment and offer the advantage of being a probabilistic model. They were introduced by Krogh et al. [8] especially for protein modelling, and are widely used in bioinformatics [9] to represent protein families. Their underlying graphical structure suits the characteristics of an alignment. The structure for our PHMM closely follows the popular PHMM model of the HMMER software [13], and differs slightly from the original proposition of Krogh et al [8]. There are no transitions from an insert to a delete state and vice versa as these transitions are very unlikely in the biological domain. Each position in a PHMM consists of a match, insert and delete state, exceptions are the first position and the last position. In the first position there is only a match and an insert state, and in the last position there is only a match state. Figure 2 shows a PHMM with four match states.

As stated previously, a PHMM can represent a Multiple Sequence Alignment. Figure 3 shows such a Multiple Sequence Alignment for the unaligned sequences from Figure 1. Following convention in the biological literature, deletions are marked with a '-'. In addition matching parts of the sequence are indicated by upper case letters, whereas insertions are displayed with lower case letters. The

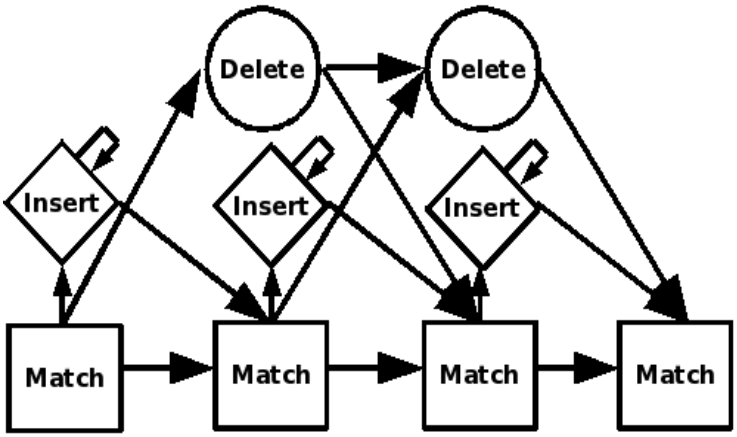

Fig. 2. The general structure of a Profile Hidden Markov Model (PHMM) of length 4.

graphical structure of a PHMM easily allows this representation and therefore to identify conserved regions, so that they can be displayed below one another. A Multiple Sequence Alignment, like the one presented in Figure 3, is part

```
MMFFA           DDAAA--E
MMFFArrnsstnnrrEDPFMLWE
MMFFA           -------E
```

Fig. 3. A global Multiple Sequence Alignment for our example sequences. This is one part of the output of a trained PHMM.

of a trained PHMM's output. However, because it is a probabilistic model, its representation of an alignment is much richer. We will make use of the alignment itself, as well as of the probabilistic features of a PHMM.

In this paper, PHMMs are trained from unaligned sequences using the Baum-Welch algorithm [9]. These trained models can be used directly for classification by calculating of the probability of a test sequence given the model. We use the performance of these models as a baseline comparison.

During training, a PHMM for each class is constructed separately. Therefore each PHMM is only trained on its positive instances. As Jaakkola et al. [3] point out, discriminative tasks such as classification might benefit from using the negative examples as well. They extract a kernel description of a HMM and use this for classification by a support vector machine. We achieve discriminative modelling by extracting features from the PHMM through propositionalisation and subsequently training a propositional learner. This approach is not restricted to kernel-based classifiers.

3 Kernel approaches

Kernel approaches are popular and extensively used in the machine learning community. Using kernel methods on structured data [4, 3, 10] has led to accurate classifiers [1]. Nevertheless, defining an appropriate kernel function is still a difficult problem [1].

For biological sequence analysis there are three different kernel approaches that are of special interest: general graph kernels used on a problem-specific graph for proteins [4], kernels defined over an arbitrary HMM [10] and Fisher kernels defined on an HMM representing a protein family [3].

Borgwardt et al. [4] define a random walk graph kernel over a specially defined graphical model for proteins. Each graph represents one protein and similarity is calculated by a graph kernel. Our approach differs in that respect, because we use a general model, a PHMM, to model the alignment of proteins. We align different proteins using one graphical model. In the PHMM the similarity is expressed by a probability. We propositionalise our structured model and therefore do not need to define a similarity measure.

Recently a probabilistic pair-wise kernel over HMMs has been used for spectral clustering of time series data [10]. This probability product kernel measures the distance between two HMMs each representing a time series. The clustering of the time series data is then based on this similarity measure. This semi-parametric approach allows the representation of the inherent structure of the time series data without being excessively strict with assumptions about the overall distribution. In the same way, we assume an HMM structure for all instances belonging to the same class, but make no assumption about the overall distribution. In our approach a PHMM is trained separately for each class.

Jaakkola et al. [3] define the Fisher score over a trained HMM for a protein family and use it as a kernel function in their support vector machine approach. They successfully use a trained generative model as an intermediate step to obtain a discriminative model by explicitly defining a similarity function. The HMMs used by Jaakkola et al. [3] are PHMMs. However, they allow transitions from insert to delete states and vice versa and a sequence alignment can begin and end with a match, insert or delete. In addition they align sequences locally, whereas we align sequences globally. Their PHMM is described by Karplus et al. [14]. In our approach we use the generative model directly to extract features for discriminative modelling. In addition, their approach is restricted to kernel based classifiers whereas we can use the extracted features as input for any propositional learner.

A drawback of all kernel approaches is the need to explicitly define a similarity measure with the kernel function. Propositionalisation does not have this requirement.

4 Propositionalisation

Propositionalisation transforms complex, structured representations of data into a fixed length representation involving attribute-value pairs [6]. Therefore, it de-

couples model from feature construction and introduces a new degree of flexibility and a wide range of possible features to be constructed from the more complex representation. Traditionally machine learning has focused on propositional learners, thus they are biased to this representation.

In this paper, propositionalisation means to take a probabilistic, graphical model and transform it into a fixed length attribute-value representation. To state it clearly, the suggested technique uses PHMMs as input and creates a fixed number of attributes and corresponding values from them. The major contribution of this paper is the examination of different ways to propositionalise a PHMM. The complexity of the different propositionalisations is varying. In addition, most of the propositionalisations can be used with HMMs in general as well.

By de-coupling model from feature construction we do not need to explicitly define a similarity measure for graphs like in kernel based methods. The representation resulting from propositionalisation can be used as input for a wide range of classifiers including but not limited to kernel-based classifiers. We propositionalise PHMMs and show extensions to propositionalise general HMMs. This propositionalisation and the subsequent propositional classification step require only a small overhead compared to the training of the underlying PHMM. As training time of the PHMM is a dominant factor, it is favourable to reduce it. Training a PHMM with smaller sequences and subsequently applying propositionalisation can achieve this. Using this approach the propositional learner might be able to cope with less information. We will investigate this potential later.

The key idea behind our propositionalisation approach is simply to take advantage of the canonical form of the underlying graph of a PHMM. The aim of the propositionalisation is to get a fixed length attribute-value representation of the datatset. This new representation should be as simple as possible in order not to include too many, potentially irrelevant features. On the other hand, a representation which is too simple might not capture all the information that is essential for an accurate classification. Therefore, we use propositionalisations of different complexities and compare the results.

In order to get a propositional representation for an input sequence, we calculate the most probable path for the sequence through the trained PHMM. This path is also known as the Viterbi path. On the Viterbi path each match state is visited except when it is skipped by a delete state. In the case of a match, it can be followed by a sequence of insertions. This directly tells us how to get a propositional representation of the Viterbi path by exploiting the fact that a PHMM allows the representation of the Viterbi path of any sequence in a fixed length way. The process proceeds as follows:

1. Create a nominal attribute for each match state. The values of the nominal attributes are the emitted symbols plus an extra symbol representing a deletion. In our PHMM the emission alphabet comprises the symbols defining amino acids.

2. Create a numerical attribute for each insert state. Set the value of the attribute according to the number of times this state is visited in the Viterbi path, i.e. the number of insertions at that point.

Thus every unit of a match, an insert and a delete state is converted into two attributes. Figure 4 shows the propositional representation for the simple example Multiple Sequence Alignment from Figure 3.

```
M,0,M,0,F,0,F,0,A, 0,D,0,D,0,A,0,A,0,A,0,-,0,-,0,E
M,0,M,0,F,0,F,0,A,10,E,0,D,0,P,0,F,0,M,0,L,0,W,0,E
M,0,M,0,F,0,F,0,A, 0,-,0,-,0,-,0,-,0,-,0,-,0,-,0,E
```

Fig. 4. The propositional representation of the example Multiple Sequence Alignment.

This representation is highly motivated by the graphical form of a PHMM and is simple. It can be used by any propositional learner that can handle nominal and numeric attributes. Hence we use a generative model for feature generation and convert it into a discriminative one via propositionalisation for the classification task.

The classification problems considered in this research are multi-class classification problems. For each class we train one PHMM using only the sequences belonging to this class. To generate a propositional representation of a sequence, we calculate the Viterbi paths through the trained PHMMs of all classes and combine the resulting propositional representations from each PHMM into one representation.

Using only the Viterbi path to propositionalise leads to a simple representation that is limited to PHMMs. In the remainder of this paper we will refer to this simple form of propositional representation as mode 1. It is preferable to have a simple way to propositionalise not only PHMMs, but HMMs in general. In order to achieve this goal, we have to make sure to generate fixed length descriptions independent of the length of the path. The following modes can all be applied to HMMs in general and are not limited to PHMMs. Thus, as long as we do not include the simple mode 1 into the propositional representation, our approach can be applied to HMMs in general. Mode 2 converts the whole HMM into a single numerical attribute. It represents the score of the Viterbi path for each sequence given the HMM. In the same way, we construct another numeric attribute in mode 3 representing the score of the sequence given the HMM. This is the sum of the scores of all possible paths, not only the score of the Viterbi path. These are two very simple ways to propositionalise general HMMs. However, there is much more information present in the underlying HMM. In mode 4 we create a numeric attribute for each state of the HMM representing the score of the state given the sequence. This representation has a fixed length for all input sequences. In the same way, the probability of each state given the sequence is used in mode 5. An advantage of a propositional approach is that we are now able to put together all different modes in all possible combinations.

Table 1. The propositional algorithms used in the experiments

Propositional Algorithm	Parameters
Naive Bayes	
k-Nearest Neighbours	uses best k between 1 and 100
Random Forest	with 100 trees and various numbers of features
SMO	with linear, polynomial and RBF kernel
Bagging	using 100 iterations with an unpruned J48
AdaBoost	using 100 iterations with a pruned J48

5 Experiments

For our experiments we set up the PHMM in the following way. In general, and in the absence of prior knowledge, the PHMM has as many match states as the average number of residues in the training sequences as suggested by Durbin et al. [9]. If the sequence length is artificially restricted for a training set, the number of match states equals the cutoff number for the sequence length. The emission probabilities in the insert and match states are initialised uniformly. We only train emissions in match states. All transitions are initialised uniformly as well. This is a very simple initialisation, because we do not assume any prior knowledge. Our purpose is to demonstrate the extent to which classification can be improved by adding a subsequent propositionalisation step over classification purely based on the generative PHMM.

The PHMM is trained from unaligned sequences using the Baum-Welch algorithm, a special case of the EM algorithm. It guarantees convergence to a (local) optimum. The convergence criterion is a sufficiently small change in the log-odds score relative to a random model. This score is normalized by the number of residues in a sequence[1]. Subsequently the trained PHMM is used for propositionalisation. In each classification problem, we train one PHMM per class label.

We test our propositional representations on various propositional learners. See Table 1 for an overview.

The datasets are taken from the Protein Classification Benchmark Collection [15]. They belong to the *3PGK_Protein_Kingdom_Phylum* set of sequences. It consists of 10 binary classification problems which map protein sequences into kingdoms of life based on phyla. The database provides training and test sets and reports benchmark results for the Area Under the ROC Curve (AUC) for these problems. We chose the two classification problems Eukaryota/Euglenozoa and Bacteria/Proteobacteria for our investigations. In the remainder of the paper we will refer to the former as 3pgk6. The latter will be called 3pgk4. The reported benchmark results in the data collection suggest that these two binary classification tasks are the most challenging ones in the *3PGK_Protein_Kingdom_Phylum* problem domain. For all datasets the best and worst AUC values are reported.

[1] The threshold used was proposed by A. Krogh in a personal e-mail communication.

Considering only the best AUC values for all ten classification problems in the *3PGK_Protein_Kingdom_Phylum* data, 3pgk4 best AUC result is 0.93 and the one for 3pgk6 is 0.92. No other dataset has a lower AUC value for its best performing classification model. In addition, all experimental results for 3pgk4 and 3pgk6 achieve at least an AUC value of 0.56. There is only one dataset with a lower overall worst AUC value.

The training set of 3pgk6 consists of 82 training instances whereas the test set has 49 instances. The average sequence length in the training set is 414 and 406 in the test set. The training set and the test set each have 44 positive instances. In the 3pgk4 training set there are 70 instances, 27 of them are positive. The test set consists of 61 instances. Of these, 31 instances belong to the positive class. In the training set the average sequence length is 413, whereas it is 408 in the test set. For both datasets sequences vary in general between 358 and 505 residues.

These datasets are used twice in the experiments. In a first setting we use the full sequence information, whereas in the second setting we artificially reduce the length of the sequence to the first 100 residues. All experiments are performed using the WEKA machine learning workbench [16].

We also present the result from a pure PHMM approach where the PHMM is used directly for classification. In this setting a PHMM for each class label is built as well. The log-odds score of the test sequence given the model is used as a basis for classification.

In all results we only show the propositional learner and its respective parameters which performed best. First, we present the results for the 3pgk6 dataset. In the first experiment we use the full sequence information. The results are shown in Table 2. All propositional representations outperform the pure PHMM

Table 2. The accuracies for the 3pgk6 dataset. The sequence length is not restricted. The propositional learner is a Support Vector Machine with RBF Kernel, γ is set to 1. We fit logistic models to the output. The baseline accuracy of the PHMM without propositionalisation is included as well.

modes' combinations	SVM accuracy	PHMM baseline accuracy
{1}, {1,2}, {1,3},{1,2,3},{4},{5}, {1,4},{1,5},{1,2,3,4},{1,2,3,5}	89.80%	81.63%

approach. In all settings a Support Vector Machine with an RBF Kernel works best. All representations, simple or complex achieve the same accuracy.

In a second experiment we restricted the sequence length artificially to the first 100 residues. We use the same combinations of modes as in the previous experiment. Table 3 summarises the results. In this case as well, the different combinations of modes lead to the same accuracy result. The best performing propositional learner is again a Support Vector Machine with an RBF Kernel.

Table 3. The accuracies for the 3pgk6 dataset. The sequence length is restricted to the first 100 residues. The propositional learner is a Support Vector Machine with RBF Kernel, γ is set to 1. We fit logistic models to the output. When mode 5 is used on its own, γ is set to 100

classifier	accuracy
PHMM	79.59%
all propositional learners	89.80%

Using propositionalisation in a setting were information is restricted does not lead to a decrease in accuracy, whereas for the pure PHMM approach it does.

Table 4 gives on overview of the results for the 3pgk4 dataset when the full sequence information is used. Again all propositional representations outperform

Table 4. The accuracies for the 3pgk4 dataset. The sequence length is not restricted. The baseline accuracy of the PHMM is 68.85%

combination of modes	propositional learner	accuracy
1	**Random Forest with 100 trees, 300 features**	**93.44%**
1,2	Random Forest with 100 trees, 200 features	90.16%
1,3	Random Forest with 100 trees, 200 features	90.16%
1,2,3	**Random Forest with 100 trees, 200 features**	**93.44%**
4	k-nearest neighbours	86.69%
5	Random Forest with 100 trees, 4 features	77.05%
1,4	k-nearest neighbours	86.69%
1,5	Random Forest with 100 trees, 15 features	73.77%
1,2,3,4	k-nearest neighbours	86.69%
1,2,3,5	Random Forest with 100 trees	77.05%

the pure PHMM based approach. For this dataset the simpler propositionalisations perform considerably better than the baseline classifier. In addition, they outperform more complex propositional representations. Best accuracy is achieved when we use the Viterbi path for propositionalisation or when we use the Viterbi path and include its score and the score of the sequence given the PHMM. However, when using the path information only, the Random Forest needs more features to achieve the same accuracy.

In the opposite case, when information is limited, the results shown in Table 5 indicate that additional information is helpful. The pure PHMM approach does not suffer from a lack of information. It achieves the same accuracy as before when the full sequence was used. The propositional approaches still outperform solely PHMM based classification. However, their accuracies are worse. In this setting the more complex propositionalisations outperform the simpler ones.

Table 5. The accuracies for the 3pgk4 dataset. The sequence length is restricted to 100 residues. The baseline accuracy of the PHMM is 68.85%

combination of modes	propositional learner	accuracy
1	Bagging	80.33%
1,2	Random Forest with 100 trees, 15 features	80.33%
1,3	k-nearest neighbours	78.69%
1,2,3	k-nearest neighbours	78.69%
4	k-nearest neighbours	81.97%
5	Random Forest with 100 trees, 4 features	81.97%
1,4	**Random Forest with 100 trees, 15 features**	**83.61%**
1,5	**linear SMO, complexity 0.001**	**83.61%**
1,2,3,4	**Random Forest with 100 trees**	**83.61%**
1,2,3,5	**linear SMO, complexity 0.001**	**83.61%**

However there are cases, when information is limited, where the pure PHMM classification results in a slightly better accuracy than the classifiers using propositionalised data as the following experiment on the 3pgk6 dataset shows. In this case, we restrict the sequence length to the first 150 residues. The baseline accuracy of the PHMM is 91.84%. The most accurate propositional learner is a Support Vector Machine with RBF Kernel, γ is set to 1. Additionally, it fits logistic models to the output. We used 4 different experimental setups. First, we propositionalise the dataset according to mode 1. In the second experiment, a combination of modes 1 and 2 is used for propositionalisation, whereas the third experiment employs modes 1 and 3. The last experiment is conducted with modes 1, 2 and 3. All these experiments lead to the same resulting accuracy of 89.80%. The pure PHMM approach outperforms the propositionalisation in this case. However, compared to the previous results, the difference in accuracies between the baseline and the propositional learners is not as large.

6 Conclusions and Future Work

In this paper we introduced a simple way to propositionalise a PHMM and extended it to ways of propositionalising general HMMs. We compared different propositional representations with each other and the classification performance of the PHMM as a baseline.

We showed that a simple propositionalisation of a complex, generative PHMM leads, most of the time, to better results on two benchmark datasets than a purely PHMM based classification. In cases where information is limited, more complex propositional representations perform better or equally as well as their simpler counterparts. These results indicate that propositionalisation of a PHMM is able to outperform a solely PHMM based classification approach. However, the findings on these benchmark datasets need to be verified on other datasets as well. Consequently, as a next step, we will test our approach on more and larger datasets.

In addition, we will have a closer look at the idea of reducing training time for a PHMM without losing overall discriminative power. Instead of limiting the number of residues to train smaller PHMMs, we anticipate training a PHMM ahead of full convergence of the Baum-Welch algorithm, and then using propositionalisation. This leads to a considerable speed-up of the training process. Propositionalisation might be able to compensate for a decrease in accuracy.

References

1. Karunaratne, T., Boström, H.: Learning to classify structured data by graph propositionalization. In: Proceedings of the 2nd IASTED International Conference on Computational Intelligence, IASTED/ACTA Press (2006) 283–288
2. Zaki, M.J., Aggarwal, C.C.: Xrules: An effective structural classifier for xml data. In: KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM (2003) 316–325
3. Jaakkola, T., Diekhans, M., Haussler, D.: Using the fisher kernel method to detect remote protein homologies. In: Proceedings of the 7h International Conference on Intelligent Systems for Molecular Biology, AAAI Press (1999) 149–158
4. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.: Protein function prediction via graph kernels. Bioinformatics **21**(1) (2005) 47–56
5. Pfahringer, B., Holmes, G.: Propositionalization through stochastic discrimination. In: Proceedings of the Work-in-Progress Track at the 13th International Conference on Inductive Logic Programming, Department of Informatics, University of Szeged, Hungary (2003) 60–68
6. Krogel, M., Rawles, S., Železný, F., Flach, P., Lavrač, N., Wrobel, S.: Comparative evaluation of approaches to propositionalization. In: Proceedings of the 13th International Conference on Inductive Logic Programming. (2003) 197–214
7. Lachiche, N.: Good and bad practices in propositionalisation. In: Advances in Artificial Intelligence, 9th Congress of the Italian Association for Artificial Intelligence (AI*IA). (2005) 50–61
8. Krogh, A., Brown, M., Mian, I., Sjölander, K., Haussler, D.: Hidden markov models in computational biology: Applications to protein modelling. Journal of Molecular Biology **235**(5) (1994) 1501–1531
9. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: Biological sequence analysis: probabilistic models of proteins and nucleic acids. Cambridge University Press (1998)
10. Jebara, T., Song, Y., Thadani, K.: Spectral clustering and embedding with hidden markov models. In: Proceedings of the 18th European Conference on Machine Learning (ECML). Volume 4701 of Lecture Notes in Computer Science., Springer (2007) 164–175
11. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE **77**(2) (1989) 257–286
12. Bystroff, C., Thorsson, V., Baker, D.: Hmmstr: A hidden markov model for local sequence-structure correlations in proteins. Journal of Molecular Biology **301** (2000) 173–190
13. Eddy, S.: Hmmer user's guide: biological sequence analysis using profile hidden markov models. http://hmmer.wustl.edu/ (1998)

14. Karplus, K., Barrett, C., Hughey, R.: Hidden markov models for detecting remote protein homologies. Bioinformatics **14**(10) (1998) 846–856
15. Sonego, P., Pacurar, M., Dhir, S., Kertész-Farkas, A., Kocsor, A., Gáspári, Z., Leunissen, J.A.M., Pongor, S.: A protein classification benchmark collection for machine learning. Nucleic Acids Research **35**(Database-Issue) (2007) 232–236
16. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. 2 edn. Morgan Kaufmann (2005)