V. Kůrková et al. (Eds.): ITAT 2014 with selected papers from Znalosti 2014, CEUR Workshop Proceedings Vol. 1214, pp. 72–78
http://ceur-ws.org/Vol-1214, Series ISSN 1613-0073, © 2014 A. Ridzik, B. Brejová

Neuropeptide Recognition by Machine Learning Methods

Andrej Ridzik[1], Broňa Brejová[2]

[1] Institute of Informatics of the Slovak Academy of Sciences,
Dubravská cesta 9, 845 07 Bratislava, Slovakia
[2] Faculty of Mathematics, Physics, and Informatics,
Comenius University, Mlynská dolina, 842 48 Bratislava, Slovakia

Abstract: Thanks to advances in DNA sequencing and bioinformatics methods, for many species we know their genomic sequence as well as sequences of proteins produced by their cells. However, the exact function of those proteins is often unknown. The goal of our work is to automatically recognize neuropeptides, special proteins involved in communication between neurons. Neuropeptides are created in a cell from longer protein precursors, and our goal is to determine, if a given protein is a likely precursor and if yes, which of its parts will serve as neuropeptides. Existing methods solve only partial aspects of this problem. Our more comprehensive system uses a combination of support vector machines and conditional random fields.

1 Introduction

In this paper, we study the problem of neuropeptide annotation. Recent advances in DNA sequencing allow us to obtain DNA sequences of many species. We can then use computational methods to find likely locations of genes encoding proteins produced by the organism in question (Yandell and Ence, 2012). The result is a list of putative proteins, each characterized by its sequence of amino acids, which can be represented in a computer as a string over 20-letter alphabet. However, the exact function of these proteins is often unknown.

The goal of our work is to automatically recognize neuropeptides, short proteins acting as neurotransmitters in communication between neurons and as hormones in endocrine regulation (Hook et al., 2008). Neuropeptides are created in cells from longer protein precursors, and our goal is to determine, if a given protein is a likely precursor and if yes, which of its parts will serve as neuropeptides.

Several similar methods were already developed, but each considers only partial aspects of the problem (Southey et al., 2006, 2008b; Ofer and Linial, 2014). We will review them, together with further biological background related to the problem, in the next section.

Our system aims to solve the full problem including recognition of precursors and identification of likely neuropeptides. For this purpose, we use semi-Markov conditional random fields (semi-CRF). This is a probabilistic model for annotating sequences, which generalizes hidden Markov models (HMMs). Both HMMs and semi-CRFs were previously used for several other problems in bioinformatics (Burge and Karlin, 1997; Krogh et al., 2001; DeCaprio et al., 2007).

2 Background and Related Work

As outlined above, a *protein* can be represented as a string over a 20-letter alphabet, where each letter represents one of the 20 amino acids commonly occurring in proteins produced by living cells. Shorter chains of amino acids are often called *peptides*, rather than proteins.

The process of neuropeptide synthesis in a cell is illustrated in Figure 1. First, a full-length precursor is synthesized by the standard gene expression machinery of the cell. This precursor starts with a special short sequence called a signal peptide, which targets the synthesized protein to the endoplasmic reticulum. In this organelle, the signal peptide is cleaved away, resulting in a shorter protein, which is transported to the Golgi apparatus. Proteins called convertases then cut the precursor to several shorter peptides. Some of the peptides resulting from this cleavage serve as neuropeptides, and are often further modified by removal of amino acids or attachment of other chemical groups. The remaining peptides resulting from the cleavage are destroyed by the cell; these are called propeptides. Convertases cleave the precursor at specific positions. For example, furin convertase cleaves after a peptide sequence of length 4 conforming to the consensus `RX[KR]R`, where R stands for arginine amino acid, K for lysine, [KR] for any these two, and X for any amino acid. However, not every occurrence of this consensus sequence is cleaved and the exact rule for recognizing cleavage sites is not known. More details of this process can be found for example in a review by Hook et al. (2008).

We will distinguish three problems related to finding neuropeptides among all proteins synthesized by the cells of a given species.

- *Neuropeptide precursor recognition.* This is a classification problem, which gets a protein sequence as an input and the goal is to decide if this protein will serve as a likely precursor for neuropeptides.

- *Prediction of precursor cleavage sites.* Here the input is a neuropeptide precursor, and the goal is to determine positions where the convertase will cleave the protein. It can also be formulated as a decision problem, where we want to decide for each position in

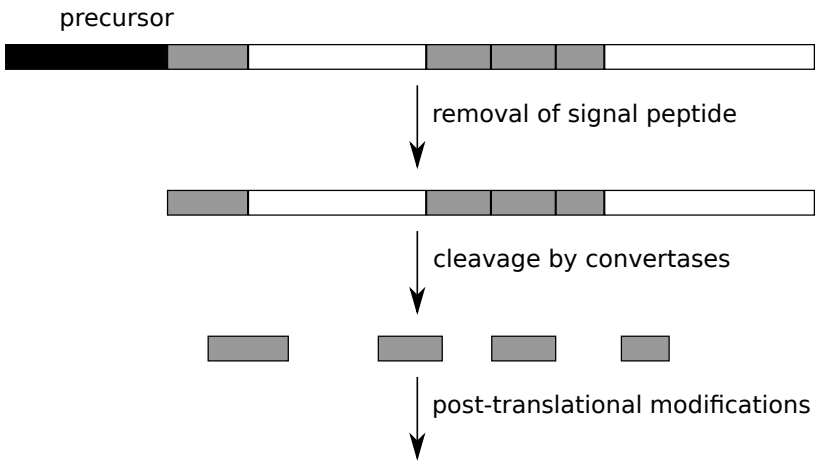

Figure 1: The synthesis and processing of neuropeptides in a cell. In the precursor, the black area represents the signal peptide, gray rectangles represent neuropeptides and white areas are propeptides discarded by the cell.

the precursor, if it is a likely cleavage site based on a sequence window surrounding this position.

- *Neuropeptide annotation.* This is the most general problem. The input might be a neuropeptide precursor or a different type of protein. The goal is to decide, if it is a likely neuropeptide precursor, and if yes, find which portions are likely to serve as individual neuropeptides. This problem encompasses both previous problems, and adds additional information, because neither of the two previous problems classifies individual products of cleavage as propeptides and neuropeptides.

Prediction of precursor cleavage sites is traditionally addressed by a web-based tool called NeuroPred (Southey et al., 2006, 2008a). NeuroPred uses both handcrafted consensus motifs as well as methods of machine learning. These methods (including logistic regression, neural networks, and the *k*-nearest neighbors algorithm) were trained on a set containing both positive and negative examples from well-studied proteins. Each classification method considers a short region of the sequence and decides if it is a likely cleavage site or not.

Authors of NeuroPred also developed a pipeline for identification of likely precursors (Southey et al., 2008b). However, this pipeline can only find precursors which have similar sequence as one of the already known precursors from related species, and thus it is not useful for finding completely overlooked precursors or for precursors from species that do not have a closely related species where neuropeptides were already identified.

Recently, Ofer and Linial (2014) have created a tool called NeuroPID which identifies likely neuropeptide precursors based solely on statistical information about the protein sequence, such as distribution of amino acids in short sequence windows, sequence entropy and others. They define a large number of such features and use known precursors as well as negative examples to train several well-known classification methods, such as support vector machines and random forests.

In our work, we focus on the general neuropeptide annotation problem, which was not addressed by existing approaches. We attempt to recognize neuropeptides and their precursors based solely on sequence features, without requiring sequence similarity to known neuropeptides.

3 Overview of Our System

The goal of our system is to assign to each amino acid of the input protein a label characterizing its role. We will use the following labels in neuropeptide precursors: `S` for signal peptides, `N` for neuropeptides, `P` for discarded propeptides, and `C` for cleavage positions. In negative examples (proteins, which are not neuropeptide precursors), we use label $\bar{\texttt{P}}$, but if these proteins also undergo cleavage into smaller peptides, we mark the cleavage sites by $\bar{\texttt{C}}$ and the resulting peptides by `X`. The result of annotation is thus a string over the alphabet of possible labels $\Sigma = \{\texttt{S}, \texttt{N}, \texttt{P}, \texttt{C}, \texttt{X}, \bar{\texttt{C}}, \bar{\texttt{X}}\}$ of the same length as the input protein, where each label describes a putative role of one amino acid. Neuropeptide precursors are marked by the presence of regions labeled `N`, and these regions are exactly the peptides predicted to serve as neuropeptides. Individual neuropeptides are separated by an amino acid marked `C`, or even by longer regions marked `P`. The output sequence of labels is called an *annotation* of the input protein.

The main component of our system is a semi-Markov conditional random field. It uses several features based on the input sequence, which are computed directly or using external programs. One of these external programs, SignalP, is an existing and frequently used software for recognizing signal peptides based on artificial neural networks (Bendtsen et al., 2004). The second external program is a classifier based on support vector machines (SVM) for recognizing cleavage sites, which we built ourselves. Its

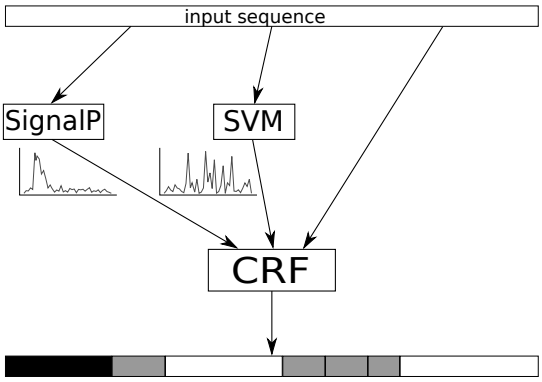

Figure 2: Architecture of our system. The input sequence is processed using SignalP and our SVM for cleavage sites recognition. The results are complemented by various simple statistics and form an input for the semi-CRF, which outputs the final annotation.

goal is to replace NeuroPred in our analysis, because NeuroPred is available only as a web-based service and as such is harder to integrate to a stand-alone tool. The overall architecture of our system is illustrated in Figure 2.

4 An SVM for Cleavage Site Prediction

To recognize cleavage sites in a protein sequence, we use one of the standard machine learning methods for classification, support vector machines (Cortes and Vapnik, 1995). We use a similar approach as NeuroPred (Southey et al., 2006). The input to the SVM is a binary vector representing a window of size $2k+1$ selected from the protein sequence. Each amino acid in the sequence window is encoded as a binary vector of length 20 in which one position is set to 1 and others to 0. The position of the value 1 within the vector indicates the identity of the amino acid. The whole window of length $2k+1$ thus requires $20(2k+1)$ binary variables. The goal of the SVM is to decide if the protein will be cleaved after the amino acid located in the middle of the window. We only consider windows that have amino acids lysine or arginine before the cleavage site, because cleavage sites seem to obey this minimal requirement.

To train the SVM on a set of examples extracted from proteins with known cleavage positions, we have used LibSVM library (Chang and Lin, 2011). We have used window of size 11 ($k = 5$). This window size performed well in NeuroPred as well as in our preliminary experiments on various sequences. As a kernel function, we have chosen the radial basis function. This kernel has a small number of parameters, good precision and fast convergence to optimum during training. The use of soft margin classification and radial basis function requires setting two hyper-parameters γ and C. We have set these parameters by cross-validation: we have split the training data into five groups. We have used four groups to train the model with different values of hyper-parameters and used the fifth group for evaluating the prediction accuracy of the result. This was repeated five times for each choice of the validation set, and the parameters were chosen based on averaged performance.

To evaluate the accuracy of our model, we have trained it on 16 known neuropeptide precursors from the honey bee *Apis mellifera* with total 70 cleavage sites used previously to train NeuroPred (Southey et al., 2008a). As testing data, we have used 21 precursors from the fruitfly *Drosophila melanogaster* with 87 cleavage sites. This data was described in the same paper. Our model classified correctly 91.1% of testing examples, whereas NeuroPred classified correctly 90.8%. Given this negligible difference, we conclude that our SVM classifier can be used instead of NeuroPred to predict cleavage sites in putative neuropeptide precursors.

5 Semi-Markov Support Vector Machine for Neuropeptide Annotation

In this section, we describe our system for neuropeptide annotation based on a semi-CRF.

Introduction to semi-CRFs. A conditional random field (CRF) is a probabilistic model for annotating sequences (Lafferty et al., 2001). Let $\mathbf{X} = (x_1, \ldots, x_n)$ be the input sequence and $\mathbf{Y} = (y_1, \ldots, y_n)$ its annotation. A CRF defines a conditional distribution over possible annotations $\mathbf{Y}$ given sequence $\mathbf{X}$, i.e., $\Pr(\mathbf{Y}|\mathbf{X})$. To define this distribution, it uses K local features $g_1, \ldots, g_K$ which can be tailored specifically for the studied problem.

We will use semi-CRF, which are a slight generalization of CRFs and differ in the exact definition of a local feature (Sarawagi and Cohen, 2005). In a semi-CRF, the annotation $\mathbf{Y}$ is expressed as a series of p segments $(s_1, \ldots, s_p)$, each segment using the same label. Segment s_i is given by a triple (b_i, e_i, a_i), where b_i is the start of the segment, e_i is its end and a_i is the annotation label. These values have to satisfy the following constraints:

$$b_1 = 1; \quad e_i \geq b_i; \quad e_{i-1} + 1 = b_i; \quad e_p = n; \quad a_{i-1} \neq a_i,$$

$$y_{b_i} = y_{b_i+1} = y_{b_i+2} = \cdots = y_{e_i} = a_i.$$

Each local feature g_k has five inputs: the whole input sequence $\mathbf{X}$, triple (b_i, e_i, a_i) describing a potential segment in the annotation, and the label a_{i-1} of the previous segment. The resulting number $g_k(a_{i-1}, b_i, e_i, a_i, \mathbf{X})$ represents a score how well the potential labels a_{i-1}, a_i fit the information from sequence X at positions specified by b_i and e_i. A global feature G_k for a given annotation $\mathbf{Y}$ can be obtained as a sum of local features for all segments in Y:

$$G_k(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{p} g_k(a_{i-1}, b_i, e_i, a_i, \mathbf{X}), \qquad (1)$$

The conditional probability of annotation $\mathbf{Y}$ in a semi-CRF depends on a weighted combination of global features

$$\Pr(\mathbf{Y}|\mathbf{X},\mathbf{w}) = \frac{1}{Z_{\mathbf{w}}(\mathbf{X})} \exp\left(\sum_{k=1}^{K} w_k G_k(\mathbf{X},\mathbf{Y})\right). \quad (2)$$

Here, $\mathbf{w}$ represents weights of global attributes and $Z_{\mathbf{w}}(\mathbf{X})$ is the normalization constant defined as

$$Z_{\mathbf{w}}(\mathbf{X}) = \sum_{\overline{\mathbf{Y}}} \exp\left(\sum_{k=1}^{K} w_k G_k(\mathbf{X},\overline{\mathbf{Y}})\right). \quad (3)$$

The summation runs through all possible annotations $\overline{\mathbf{Y}}$ of sequence $\mathbf{X}$. The values of individual feature functions can be arbitrary real numbers, because the normalization constant $Z_{\mathbf{w}}(\mathbf{X})$ ensures that we obtain a proper probability distribution over all annotations of a given sequence.

To create a semi-CRF, we manually choose a set of local features, and then use training data to find an optimal set of weights to maximize the conditional probability $\Pr(\mathbf{Y}|\mathbf{X},\mathbf{w})$. For inference, we are given the model, including the weights, and a sequence $\mathbf{X}$, and our goal is to find the optimal annotation $\arg\max_{\mathbf{Y}} \Pr(\mathbf{Y}|\mathbf{X},\mathbf{w})$. This can be done by a dynamic programming algorithm similar to the standard Viterbi algorithm for hidden Markov models (Sarawagi and Cohen, 2005). We have used an existing implementation of semi-CRFs (Sarawagi et al., 2014) for both training and inference in our model.

Topology of our model. Our model uses the set of annotation labels Σ described above. However, there are restrictions on possible sequences of labels in a permissible annotation, which are depicted in the state diagram shown in Figure 3. Every sequence starts with a signal peptide (we exclude proteins not containing this peptide from our analysis, as the presence of the peptide can be detected using the SignalP software). The model has then two branches, the bottom one corresponding to neuropeptide precursors and the top one to other proteins, which may or may not be cleaved. The cleavage sites C and $\overline{\text{C}}$ are always segments of length 1; other segments can be longer.

Feature functions of the model. Our model uses several groups of feature functions described below. While designing these feature functions we took into account a relatively small size of the available training sets as well as limitations of the used semi-CRF implementation.

- *Model topology.* Model topology described above is enforced only indirectly, by using the following set of feature functions. For each pair of labels u and v we create a feature function which indicates, if this pair occurs in two adjacent segments of a potential segmentation.

$$g_{edge_{(u,v)}}(a',b,e,a,\mathbf{X}) := \begin{cases} 1 & \text{if } a' = u \wedge a = v, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

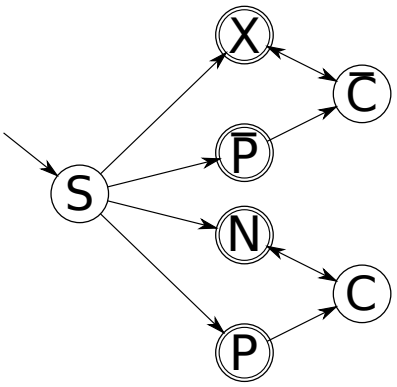

Figure 3: A state diagram describing admissible annotations in our model. Each state corresponds to one segment with a particular label. Transitions connect labels that can follow each other in an annotation. Each annotation starts with a segment labeled *S*. States corresponding to possible final segments are indicated by double circles.

Such feature function is created for all pairs of labels, even those that should not follow each other in our state diagram. However, such forbidden pairs are never used in our training data, and as a result, their weight should be negative (in theory $-\infty$). In contrast, pairs of labels that follow each other frequently in our training set should get a positive weight by the training procedure.

To control initial and final labels of a sequence, we also create the following two feature functions for each label v:

$$g_{begin_v}(a',b,e,a,\mathbf{X}) := \begin{cases} 1 & \text{if } a = v \wedge b = 1, \\ 0 & \text{otherwise,} \end{cases} \quad (5)$$

$$g_{end_v}(a',b,e,a,\mathbf{X}) := \begin{cases} 1 & \text{if } a = v \wedge e = |\mathbf{X}|, \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

- *Signal peptide information.* During preprocessing, we run SignalP on our input sequence, obtaining for each position a probability that this position is the end of the signal peptide. Let us denote this sequence of SignalP scores $\sigma_1,\ldots,\sigma_n$. Ideally, we would have a high value of this signal at the last position of the initial segment labeled *S*. This is captured by the following feature function.

$$g_{signal}(a',b,e,a,\mathbf{X}) := \begin{cases} \sigma_e & \text{if } b = 1 \wedge a = \text{S}, \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

- *Cleavage site prediction.* Similarly, we run our SVM model to obtain a probability at each position of the sequence, that this position will be cleaved. Let us denote the sequence of these scores $\alpha_1,\ldots\alpha_n$. Cleaved positions should be annotated by labels C or $\overline{\text{C}}$. To

capture this information, we use the following two feature functions:

$$g_{cl_v}(a',b,e,a,\mathbf{X}) := \begin{cases} \alpha_e & \text{if } a = v \wedge b = e, \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

where $v \in \{\mathtt{C}, \overline{\mathtt{C}}\}$. The weight of the feature assigned by training corresponds to the importance of the cleavage site prediction for predicting the correct annotation.

For every other label type, we want small values of the cleavage probability in the entire segment, because a high probability of cleavage may represent a cleavage site missed by the predicted annotation. Therefore for each label type $v \notin \{\mathtt{C}, \overline{\mathtt{C}}\}$ we add a feature function which considers the average squared value of a cleavage signal inside a segment with this label.

$$g_{\neg cl_v}(a',b,e,a,\mathbf{X}) := \begin{cases} \dfrac{1}{e-b+1}\displaystyle\sum_{i=b}^{e} \alpha_i^2 & \\ \quad \text{if } a = v, & \\ 0 \quad \text{otherwise.} & \end{cases} \quad (9)$$

Ideally, this feature should get a negative weight in training, because we want cleavage signals inside segments to be weak.

- *Segment length.* Feature functions in a semi-CRF allow us to score segment lengths. Ideally, we would use training data to estimate length distribution of different segment types and use probabilities from these distributions as feature values. Due to limited training data, we have decided to use simple features that classify each length as short, normal or long as follows.

$$g_{short_v}(a',b,e,a,\mathbf{X}) := \begin{cases} 1 & \text{if } a = v \wedge e-b+1 < min_v \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

$$g_{norm_v}(a',b,e,a,\mathbf{X}) := \begin{cases} 1 & \text{if } a = v \wedge \\ & min_v \le e-b+1 \le max_v \\ 0 & \text{otherwise,} \end{cases} \quad (11)$$

$$g_{long_v}(a',b,e,a,\mathbf{X}) := \begin{cases} 1 & \text{if } a = v \wedge e-b+1 > max_v \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

We have set thresholds min_v a max_v manually. According to Clynen et al. (2010), 98% of known neuropeptides have length between 3 and 60. We use a stricter thresholds $min_{\mathtt{N}} = 5$ a $max_{\mathtt{N}} = 60$. Shorter or longer neuropeptides are also permissible, but may incur a penalty, depending on weights assigned by training to $g_{short_{\mathtt{N}}}$ and $g_{long_{\mathtt{N}}}$ features. The same threshold values were also used for other segment types.

- *Statistical properties of the sequence.* Finally, we use a set of attributes describing typical frequencies of amino acids in different segment types. In particular, Ofer and Linial (2014) observe that neuropeptides are characterized by increased frequency of aromatic amino acids (F, W, and Y). We capture this information in the following feature function.

$$g_{aroma_v}(a',b,e,a,\mathbf{X}) := \begin{cases} \dfrac{1}{e-b+1}\displaystyle\sum_{i=b}^{e} [x_i \text{ is aromatic}] & \\ \quad \text{if } a = v, & \\ 0 \quad \text{otherwise.} & \end{cases} \quad (13)$$

These features turned out to be useful, because the weight of function g_{aroma_N} was always a relatively high positive number.

To consider frequencies of other amino acids and their groups, we have included also features of the following form:

$$g_{emit_v}(a',b,e,a,\mathbf{X}) := \begin{cases} \displaystyle\sum_{i=b}^{e} \log \Pr(x_i|v) & \text{if } a = v, \\ 0 & \text{otherwise.} \end{cases} \quad (14)$$

Here, $\Pr(x|v)$ is the frequency of amino acid x in features labeled as v in the training data. This feature function corresponds to a log-likelihood of a given segment in an i.i.d. model in which each amino acid is drawn from distribution $\Pr(x|v)$.

We also use a similar feature function, where we group together amino acids with similar biochemical properties (aromatic, basic, acidic etc.). This leads to the following attributes.

$$g_{emitg_v}(a',b,e,a,\mathbf{X}) := \begin{cases} \displaystyle\sum_{i=b}^{e} \log \Pr(gr(x_i)|v) & \\ \quad \text{if } a = v & \\ 0 \quad \text{otherwise,} & \end{cases} \quad (15)$$

where $gr(x)$ represents the biochemical group of amino acid x, and $\Pr(gr(x)|v)$ is the probability that one of the amino acids from this group appears in a segment annotated as v. These probabilities are also estimated from the training data.

6 Experiments

Training and testing data. To train and test our model, we have created a data set of neuropeptide precursors and other proteins representing negative examples. To do so, we have selected curated proteins from the UniProt database (UniProt Consortium, 2008; Uniprot, 2014). This database contains annotated protein sequences created either manually based on results from scientific literature or by automated methods. We have created two data sets. The first contains only sequences from *Arthropoda* and the

```
A: keyword:"Neuropeptide [KW-0527]"
   AND taxonomy:"Arthropoda [6656]"
   NOT keyword:"Receptor [KW-0675]"

B: keyword:"Neuropeptide [KW-0527]"
   AND taxonomy:"Metazoa [33208]"
   NOT keyword:"Receptor [KW-0675]"

C: NOT keyword:"Neuropeptide [KW-0527]"
   AND taxonomy:"Arthropoda [6656]"
   AND keyword:"Reference proteome [1185]"
   AND reviewed:yes

D: NOT keyword:"Neuropeptide [KW-0527]"
   AND taxonomy:"Metazoa [33208"
   AND keyword:"Reference proteome [1185]"
   AND reviewed:yes
```

Figure 4: Queries used to obtain sequences from the UniProt database. Queries A and B were used for positive examples, queries C and D for negative examples.

second from all animals (*Metazoa*). Proteins were selected using queries shown in Figure 4.

All obtained proteins were clustered based on sequence similarity using CD-HIT tool (Huang et al., 2010), where each cluster contains proteins with 90% identity. We have then selected a best annotated representative from each cluster. We have further filtered out sequences with incomplete or ambiguous annotations. This clustering and filtering process reduced the size of the *Arthropoda* data set from 1340 original proteins to 76 positive examples and for *Metazoa* from 2029 to 175 positive examples. A similar process for negative examples yielded only 35 sequences in *Arthropoda* and 255 in *Metazoa*.

To run our experiments, we have used five-fold cross validation. We have divided the set into five parts. One of these parts was used for testing and the remaining four for training. This process was repeated five times and the results were averaged. The SVM for cleavage site prediction was trained on the same training data as the CRF.

Annotation accuracy. Table 1 shows the annotation accuracy of different labels in four versions of our model. Two versions used only *Arthropoda* sequences for training and testing and two used a larger but more heterogeneous set of *Metazoa* proteins. For each set, we have also considered two models. One, denoted by subscript p, used only positive examples, and restricted the model to labels `S`, `C`, `N`, and `P`. The second one used the full model and both positive and negative examples.

To evaluate prediction accuracy, we consider each label separately and count the number amino acids that were correctly or incorrectly labeled by this label, obtaining the counts of true positives (TP), false positives (FP), and false negatives (FN). We then calculate the F_1 measure, which is a combination of precision and recall

$$F_1 = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

Here, precision (also known as positive predictive value) is the fraction of predicted occurrences of the label that are indeed correct, i.e. $\text{TP}/(\text{TP}+\text{FP})$. Recall (also known as sensitivity) is the fraction of real occurrences of the label that were correctly predicted, i.e. $\text{TP}/(\text{TP}+\text{FN})$.

In addition, we also compute the overall accuracy of the prediction as the fraction of correctly labeled amino acids, considering all labels together.

In most measures, the models restricted to *Arthropoda* are more accurate than the *Metazoa* models. This suggests that the statistical properties of neuropeptide precursors in vertebrates and invertebrates are sufficiently different to offset the advantage of using a larger combined training set.

We achieve a higher overall accuracy on the simpler problem without negative examples, which is expected, because it is an easier problem. In the general problem with negative examples, the prediction accuracy for label `C` is greater than for label $\bar{\texttt{C}}$. This might be caused by the fact that the cleavage site prediction model was trained only on neuropeptide precursors. Thus, we could potentially improve the prediction accuracy by training SVM models for neuropeptide precursors and other proteins separately.

To further quantify the effect of incorrect predictions by the SVM and SignalP on the accuracy, we have created a version of the *Arthropoda* model which replaces the scores from these two predictors by the correct values (value 1 for cleavage sites, value 0 for other sites). These values were then used for both training and testing. This modification increased the overall accuracy from 70.4% to 79.1%. This represents an upper bound of improvements achievable by changes in the SignalP and SVM predictors.

7 Conclusion

We have designed and implemented a system for annotation of neuropeptides. In future, we hope to use this system to seek novel neuropeptides in various organisms and to evaluate such predictions in collaboration with life scientists.

Acknowledgments. This research was supported by VEGA grants 1/1085/12 and 1/0719/14 and by the grant of the Slovak Academy of Sciences MVTS GAMMA.

References

Bendtsen, J. D., Nielsen, H., von Heijne, G., and Brunak, S. (2004). Improved prediction of signal peptides: SignalP 3.0. *Journal of Molecular Biology*, 340(4):783–785.

Model	F_1 measure for individual labels							%
	S	P	N	C	$\overline{\text{P}}$	X	$\overline{\text{C}}$	correct
Arthropoda$_p$	95.6%	78.6%	81.3%	88.6%	-	-	-	82.8%
Metazoa$_p$	96.0%	71.5%	72.0%	88.7%	-	-	-	75.9%
Arthropoda	98.1%	64.1%	76.0%	78.5%	34.7%	56.5%	53.7%	70.4%
Metazoa	95.9%	54.0%	65.3%	81.2%	72.4%	56.2%	48.6%	69.0%

Table 1: The accuracy of annotation for four different models described in the text. The last column contains the overall accuracy; the remaining columns show F_1 values for individual labels.

Burge, C. and Karlin, S. (1997). Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology*, 268(1):78–94.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27.

Clynen, E., Liu, F., Husson, S. J., Landuyt, B., Hayakawa, E., Baggerman, G., Wets, G., and Schoofs, L. (2010). Bioinformatic approaches to the identification of novel neuropeptide precursors. In *Peptidomics*, volume 615 of *Methods in Molecular Biology*, pages 357–364. Springer.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

DeCaprio, D., Vinson, J. P., Pearson, M. D., Montgomery, P., Doherty, M., and Galagan, J. E. (2007). Conrad: gene prediction using conditional random fields. *Genome Research*, 17(9):1389–1398.

Hook, V., Funkelstein, L., Lu, D., Bark, S., Wegrzyn, J., and Hwang, S.-R. (2008). Proteases for processing proneuropeptides into peptide neurotransmitters and hormones. *Annual Review of Pharmacology and Toxicology*, 48:393.

Huang, Y., Niu, B., Gao, Y., Fu, L., and Li, W. (2010). CD-HIT Suite: a web server for clustering and comparing biological sequences. *Bioinformatics*, 26(5):680–682.

Krogh, A., Larsson, B., von Heijne, G., and Sonnhammer, E. L. (2001). Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes. *Journal of Molecular Biology*, 305(3):567–570.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 282–289. Morgan Kaufmann.

Ofer, D. and Linial, M. (2014). NeuroPID: a predictor for identifying neuropeptide precursors from metazoan proteomes. *Bioinformatics*, 30(7):931–940.

Sarawagi, S. and Cohen, W. W. (2005). Semi-Markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems (NIPS 2004)*, pages 1185–1192.

Sarawagi, S. et al. (2014). CRF Project. http://crf.sourceforge.net/.

Southey, B. R., Amare, A., Zimmerman, T. A., Rodriguez-Zas, S. L., and Sweedler, J. V. (2006). NeuroPred: a tool to predict cleavage sites in neuropeptide precursors and provide the masses of the resulting peptides. *Nucleic Acids Research*, 34(W):W267–272.

Southey, B. R., Sweedler, J. V., and Rodriguez-Zas, S. L. (2008a). Prediction of neuropeptide cleavage sites in insects. *Bioinformatics*, 24(6):815–815.

Southey, B. R., Sweedler, J. V., and Rodriguez-Zas, S. L. (2008b). A Python analytical pipeline to identify prohormone precursors and predict prohormone cleavage sites. *Frontiers in Neuroinformatics*, 2:7.

Uniprot (2014). Uniprot release 2014-01. http://www.uniprot.org/.

UniProt Consortium (2008). The universal protein resource (UniProt). *Nucleic Acids Research*, 36(suppl 1):D190–D195.

Yandell, M. and Ence, D. (2012). A beginner's guide to eukaryotic genome annotation. *Nature Reviews Genetics*, 13(5):329–342.