LB2 Project and theory

International Bologna Master in Bioinformatics
A.A. 2024–2025

Contents

1 Glossary of Terms

- **Signal Peptide (SP)**: A short peptide sequence at the N-terminus of proteins targeted to the secretory pathway.
- **UniProtKB**: A comprehensive protein sequence database.
- **Cross-Validation**: A technique to evaluate model performance by partitioning data into training and validation sets.
- **Support Vector Machine (SVM)**: A supervised machine learning model used for classification.
- **Position-Specific Weight Matrix (PSWM)**: A matrix used to represent motifs in biological sequences.
- **Feature Extraction**: Process of converting raw data into numerical features.
- **Benchmarking Set**: A holdout dataset used to evaluate model generalization.
- **MMseqs2**: A tool for fast protein sequence clustering.
- **Deep Learning**: A subset of machine learning using neural networks with multiple layers.
- **Transformer**: A deep learning architecture based on self-attention mechanisms.

2 Theoretical Basis

2.1 Position-Specific Weight Matrix (PSWM)

Given a set S of N aligned sequences of length L, the PSPM M is computed as:

$$M_{k,j} = \frac{1}{N} \sum_{i=1}^{N} I(s_{i,j} = k)$$

Where: - $s_{i,j}$ is the residue at position j in sequence i - k is a specific amino acid - I is the indicator function

The PSWM W is then computed using log-odds ratios:

$$W_{k,j} = \log \frac{M_{k,j}}{b_k}$$

Where b_k is the background frequency of residue k.

With pseudocounts:

$$M_{k,j} = \frac{1}{N + 20} \left(1 + \sum_{i=1}^{N} I(s_{i,j} = k) \right)$$

2.1.1 Worked Example: PSWM

Suppose we align 3 sequences of length 4:

```
MKTG
MKTN
MRTG
```

At position 2: two K, one R. Without pseudocounts:

$$M_{K,2} = 2/3,\ M_{R,2} = 1/3$$

If $b_K = 0.05$, log-odds:

$$W_{K,2} = \log \frac{0.67}{0.05} \approx 2.6$$

2.2 Support Vector Machines (SVM)

The optimal hyperplane is found by solving:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i$$

Subject to:

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

The dual formulation:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

Subject to:

$$0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

2.2.1 Worked Example: Toy SVM

Points: $(x = 1, y = +1)$ and $(x = 3, y = -1)$. Optimal hyperplane is $x = 2$. Margin $= 2$. Weight $w = -2$, bias $b = 4$.

2.3 Evaluation Metrics

$$\begin{aligned}
\text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} \\
\text{Precision} &= \frac{TP}{TP + FP} \\
\text{Recall} &= \frac{TP}{TP + FN} \\
\text{F1-score} &= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \\
\text{MCC} &= \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}
\end{aligned}$$

2.3.1 Worked Example: Confusion Matrix

Confusion matrix:

	Predicted SP	Predicted non-SP
True SP	40	10
True non-SP	5	45

Metrics:

- Accuracy = (40+45)/100 = 0.85
- Precision = 40/(40+5) = 0.89
- Recall = 40/(40+10) = 0.80
- F1 = 0.84
- MCC $\approx$ 0.70

2.4 Machine Learning Models: A Comparative Overview

Machine learning encompasses a variety of algorithms, each with its own strengths, weaknesses, and ideal use cases. The following table provides a comparative overview of several key model types relevant to this project, highlighting their core characteristics.

Table 1: Comparison of Key Machine Learning Models

Aspect	**Hidden Markov Models (HMMs)**	**Neural Networks (NNs)**	**Support Vector Machines (SVMs)**	**Trees & Random Forests**
Type of use	Labelling (Sequence Classification)	Classification, Regression	Classification, Regression	Classification, Regression
Trainable parameters	Transition & Emission Probabilities	Weights and Biases	Support Vector coefficients (α)	Split points in each tree
# of params depends on	Number of hidden and observable states	Number of layers and neurons per layer	Number of support vectors	Number of trees and nodes per tree
Hyperparameters	Number of states, Topology	# of layers/neurons, Activation function	Kernel type, Regularization (C)	# of trees, Max depth
Input encoding	Sequential, Discrete	Fixed-length vectors	Fixed-length vectors	Fixed-length vectors
Training objective	Maximum Likelihood Estimation (MLE)	Minimize Loss Function	Maximize the margin between classes	Minimize impurity (e.g., Gini)
Training algorithm	Baum-Welch Algorithm (EM)	Backpropagation with Gradient Descent	Quadratic Programming (QP)	Greedy recursive partitioning
Pros	• Explicit probabilistic model. • Efficient for sequences.	• Universal approximators. • Excellent on complex data.	• Effective in high dimensions. • Memory efficient.	• Interpretable (single tree). • Handles non-linear data.
Cons	• Strong Markov assumption. • Discrete observations.	• "Black box" nature. • Data and compute heavy.	• Poor on large datasets. • Poor interpretability.	• Prone to overfitting (single tree). • Can be biased.

2.5 Deep Learning Fundamentals

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. While simple neural networks (shallow networks) have existed for decades, **deep learning** specifically refers to models with many layers—hence "deep" networks. This depth allows them to learn hierarchical representations of data, which is powerful for complex patterns like those in biological sequences.

2.5.1 Core Concepts

Artificial Neural Networks (ANNs): The building blocks are artificial neurons (or nodes) that receive input, apply a transformation (linear combination followed by a non-linear **activation function** like ReLU or Sigmoid), and produce an output. Neurons are organized into layers: an input layer, one or more **hidden layers**, and an output layer.

Depth and Hierarchy: Each layer learns to identify increasingly complex patterns. For example, in signal peptide prediction:

- Early layers might detect simple motifs or amino acid properties.
- Middle layers combine these to detect larger regions (n-region, h-region, c-region).
- Later layers integrate this information to predict the overall function (SP or not SP).

This automatic feature extraction is a key advantage over traditional ML, which often requires manual feature engineering.

Training Deep Networks:

- **Backpropagation**: The core algorithm for training. It calculates the gradient of the loss function with respect to each weight in the network by applying the chain rule of calculus, working backwards from the output layer to the input layer.
- **Gradient Descent**: An optimization algorithm used to minimize the loss function. It iteratively adjusts the weights in the direction that reduces the loss, as indicated by the gradients computed via backpropagation.

2.5.2 Common Deep Learning Architectures

Convolutional Neural Networks (CNNs): Specialized for processing grid-like data. For 1D sequences like proteins:

$$(f * g)[n] = \sum_{m=-M}^{M} f[m] \cdot g[n-m]$$

They use convolutional layers that apply filters to extract local features (e.g., k-mers), significantly reducing the number of parameters compared to fully-connected layers.

Recurrent Neural Networks (RNNs): Designed for sequential data. They have loops, allowing information to persist, making them suitable for modeling dependencies in biological sequences.

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks are RNN variants that solve the problem of learning long-range dependencies.

Transformers: A more recent architecture that has become dominant in Natural Language Processing (NLP) and is increasingly used for protein sequences. They rely on a **self-attention mechanism** to weigh the importance of different parts of the input sequence, enabling highly parallelized training and superior performance on tasks requiring context understanding.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

3 Linkage Knowledge

3.1 Biology-Informatics Integration

The project sits at the intersection of molecular biology, computer science, and statistics. Biological knowledge of the secretory pathway and signal peptide structure informs the feature extraction process (e.g., prioritizing n-terminal regions, calculating hydrophobicity). Computational models, from simple statistical matrices to complex deep learning networks, are then used to learn the mapping from these sequence features to functional labels. This integration allows for the development of predictive tools that can decipher biological instructions encoded in protein sequences.

3.2 Sequence-Structure-Function Relationship

The central dogma of molecular biology is reflected in computational approaches:

$$\text{Sequence} \rightarrow \text{Structure} \rightarrow \text{Function}$$

- **Sequence**: Primary data source (UniProtKB). Machine learning models use the raw amino acid sequence or derived features.
- **Structure**: Physical properties (hydrophobicity, charge, secondary structure propensity) serve as informative features that act as proxies for the physical structure of the peptide.
- **Function**: The ultimate goal is to predict the functional outcome: subcellular localization via the secretory pathway.

3.3 Machine Learning Paradigms

Method	Biological Interpretation	Computational Approach
von Heijne / PSWM	Cleavage site motif detection	Position-Specific Scoring Matrix
SVM	Feature-based discrimination	Maximum margin classification
Trees/RF	Rule-based decision process	Ensemble of hierarchical decisions
CNN	Local pattern & motif detection	Convolutional filters
RNN	Sequence context modeling	Recurrent connections
Transformer	Long-range dependencies	Self-attention mechanism

Table 2: Biological interpretation of computational methods for signal peptide prediction

3.4 Expanded Machine Learning Theory

The models in this project represent different philosophical approaches to learning from data.

- **Probabilistic Models (PSWM):** These models treat learning as a problem of estimating probability distributions from aligned sequences. They provide a framework for reasoning under uncertainty and are inherently interpretable as log-likelihood scores.
- **Geometric Models (SVMs):** SVMs approach classification as a problem of finding the optimal separating boundary (hyperplane) in a high-dimensional feature space. The kernel trick allows them to find complex, non-linear boundaries.
- **Ensemble Models (Random Forests):** This approach combines multiple weak learners (decision trees) to create a strong, robust model. By averaging predictions, it reduces variance and overfitting.
- **Connectionist Models (Neural Networks):** Inspired by biology, NNs learn by adjusting the strength of connections between neurons. Their power comes from distributing representations across many layers, allowing them to approximate complex functions and automatically learn relevant features from raw or minimally processed data.

3.5 Evolutionary Connections

- **Sequence conservation**: PSWM captures evolutionarily conserved motifs
- **Taxonomic distribution**: Kingdom-specific features reflect evolutionary divergence
- **Functional constraints**: SP characteristics constrained by secretory machinery

3.6 Multi-scale Integration

The project integrates knowledge across multiple scales:

Scale	Application
Molecular	Amino acid properties
Cellular	Subcellular localization
Organismal	Taxonomic classification
Computational	Machine learning models
Evolutionary	Sequence conservation

Table 3: Multi-scale integration in the project

4 Background Knowledge

4.1 Protein Subcellular Localization

Proteins function in specific cellular compartments. Knowledge of localization helps understand protein function, interactions, and potential drug targets.

4.2 Signal Peptides

SPs are N-terminal sequences that direct proteins to the secretory pathway. They are cleaved after translocation. SPs have three regions:

- Positively charged n-region
- Hydrophobic h-region
- Polar c-region with cleavage site

4.3 Machine Learning in Bioinformatics

ML techniques are used to predict protein features from sequence data. Common approaches include:

- Statistical methods (e.g., von Heijne)
- SVM-based classifiers
- Neural networks (e.g., CNNs, RNNs, Transformers)

5 Project Workflow

The project follows a standard machine learning pipeline:

1. Data Collection
2. Data Pre-processing
3. Data Analysis and Visualization
4. Feature Extraction
5. Model Implementation

6. Model Evaluation

7. Results Analysis and Reporting

6 Detailed Step-by-Step Explanation

6.1 Data Collection

Data is retrieved from UniProtKB using its REST API. Positive examples are eukaryotic proteins with experimentally verified SPs. Negative examples are proteins without SPs, localized in non-secretory compartments.

6.1.1 Output Formats

- TSV files with metadata (accession, organism, kingdom, length, cleavage site, etc.)
- FASTA files with protein sequences

6.2 Data Pre-processing

- Redundancy reduction using MMseqs2 (30% identity, 40% coverage)
- Split into training (80%) and benchmarking (20%) sets
- 5-fold cross-validation splits

6.3 Data Analysis and Visualization

Generate plots to understand data characteristics:

- Distribution of protein and SP lengths
- Amino acid composition comparisons
- Taxonomic distribution
- Sequence logos of cleavage sites

6.4 Feature Extraction

Features are extracted from the N-terminal region (first 22–40 residues):

- Amino acid composition
- Hydrophobicity (Kyte–Doolittle scale)
- Charge
- Secondary structure propensity
- Transmembrane tendency

6.5 Model Implementation

6.5.1 von Heijne Method

- PSWM built from cleavage-site contexts (–13 to +2)

- Pseudocounts = 1
- SwissProt background frequencies
- Scan N-terminal region (first 90 residues) to compute scores
- Threshold optimized via precision-recall curves

6.5.2 SVM-Based Approach

- Use scikit-learn's SVC with RBF kernel
- Feature combinations tested (AA, HP, CH, AH, TM)
- Hyperparameter tuning via grid search (C, gamma)

6.5.3 Optional: Deep Learning

- Implement CNNs, RNNs, or Transformers
- Use PyTorch/TensorFlow with Biopython for sequence encoding

6.6 Model Evaluation

- Cross-validation (5-fold)
- Blind test on benchmarking set
- Metrics: Accuracy, Precision, Recall, F1-score, MCC, AUC
- Compare with baseline (von Heijne) and state-of-the-art methods

6.7 Results Analysis

- Analyze false positives/negatives
- Examine misclassified transmembrane proteins
- Compare feature importance
- Visualize decision boundaries (if applicable)

6.8 Final Report

Write a manuscript in the style of a scientific paper, including:

- Abstract, Introduction, Methods, Results, Discussion
- Figures, tables, and references
- Code and data availability statement

7 Conclusion

The LB2 project provides hands-on experience in building a complete bioinformatics machine learning pipeline. Students learn to handle real-world data, implement classical and modern ML methods, and critically evaluate their results. The project also encourages exploration of advanced deep learning techniques for sequence analysis.

The theoretical foundations and linkage knowledge sections provide the necessary background to understand not only how the methods work but also why they are appropriate for biological sequence analysis, creating a bridge between computational techniques and biological principles.