Efficient adaptation of structure metrics in prototype-based classification

Bassam Mokbel, Benjamin Paassen, and Barbara Hammer [*]

CITEC centre of excellence, Bielefeld University, Germany

(This is a preprint of the publication [13], as provided by the authors.)

Abstract

More complex data formats and dedicated structure metrics have spurred the development of intuitive machine learning techniques which directly deal with dissimilarity data, such as relational learning vector quantization (RLVQ). The adjustment of metric parameters like relevance weights for basic structural elements constitutes a crucial issue therein, and first methods to automatically learn metric parameters from given data were proposed recently. In this contribution, we investigate a robust learning scheme to adapt metric parameters such as the scoring matrix in sequence alignment in conjunction with prototype learning, and we investigate the suitability of efficient approximations thereof.

1 Introduction

An ever increasing availability of problem-specific data formats and a rapidly growing data complexity raises the issue that data are often no longer vectorial, rather data structures such as sequences, trees, graphs, or similar have to be dealt with [3]. One prominent approach which enables machine learning for structures is based on a dissimilarity representation [14]: data are described by pairwise dissimilarities given by some problem-specific dissimilarity measure such as sequence alignment, structure alignment, graph or tree kernels. Then any machine learning technique which is capable of processing proximity data can be applied.

Facing such data, one particular problem with classical learning techniques is the inherent discrete nature of structured data, hence smooth model updates become difficult. For dissimilarity data we can rely on an implicit embedding of data in an underlying pseudo-Euclidean vector space or more general Krein space [14]. Mimicking the popular kernel trick, it is possible to extend many vectorial, distance-based methods to such an embedding, resulting in so-called

[*]Funding by the DFG priority programme 1527 *"autonomous learning"* under grant number HA 2719/6-1, and by the CITEC centre of excellence is gratefully acknowledged.

relational methods. Popular examples include unsupervised models such as *relational self-organising maps* and *relational generative topographic mapping*, or supervised counterparts such as *learning vector quantization* (LVQ) based schemes. [7, 4, 6]. Here we will exemplarily address prototype-based LVQ.

Specifically, we will consider *relational generalized LVQ* (RGLVQ) as an extension of *generalized LVQ* (GLVQ) to dissimilarity data [7]. GLVQ constitutes a popular and mathematically well-founded LVQ scheme with successful applications ranging from bioinformatics to robotics [15, 10, 16]. One interesting extension is its combination with metric learning, which not only enhances the representational power but also facilitates model interpretability [16]. While relational variants also yield a robust prototype-based model, their metric parameters are usually fixed, failing in situations where these parameters are unsuitable. We address the challenge to adapt metric parameters in RGLVQ, extending upon a well-proven concept of smooth metric learning in vectorial GLVQ [16].

More specifically, we consider symbolic sequences and sequence alignment as one relevant type of structured representation. Alignment heavily depends on the underlying scoring matrix which assigns scores to local symbolic comparisons and gaps. In bioinformatics, these can be inferred from evolutionary models, but in general their choice is based on a-priori assumptions about the domain or data space [17]. A few promising approaches how to infer scores from exemplary alignments have been proposed [5, 18, 1, 2]. One approach proposes to adapt scores for a discriminative classification task in conjunction with RGLVQ training, leading to first promising results [12]. The goal of this contribution is to extend this approach to a more robust adaptation suitable for realistic problems, and to test how it can be approximated to increase computational efficiency.

2 Learning vector quantization for sequence alignments

LVQ models represent vectorial data $\vec{a}^i$ by prototypes $\vec{w}^j$ with labels $c(\vec{w}^j)$ [16]. Classification uses a winner-takes-all rule: a data point is classified according to its closest prototype. Given labeled data $(\vec{a}^i, c(\vec{a}^i))$, GLVQ minimizes the cost

$$\sum_{i=1}^{N} \Phi\left(\frac{d^+(\vec{a}^i) - d^-(\vec{a}^i)}{d^+(\vec{a}^i) + d^-(\vec{a}^i)}\right)$$

where Φ is a monotonic function, d^+ is the distance of $\vec{a}^i$ to the closest prototype with a matching label, d^- refers to a non-matching label [16].

For dissimilarity data described by a symmetric matrix D with entries d_{ij}, an extension to *relational LVQ* is possible [14, 7]: a pseudo-Euclidean space and vectors $\vec{a}^i$ exist which induce d_{ij} [14]. Prototypes are given as convex combinations $\vec{w}^j = \sum_i \alpha_i^j \vec{a}^i$ with $\sum_i \alpha_i^j = 1$ inducing $d(\vec{a}^i, \vec{w}^j) = \sum_l \alpha_l^j d_{il} - 0.5 \sum_{ll'} \alpha_l^j \alpha_{l'}^j d_{ll'}$. This can be computed based on the coefficients $\vec{\alpha}^j$ and dissimilarities D only, without explicitly referring to vectors $\vec{a}^i$ or their counterparts of actual data entities (e.g. string sequences), see [7]. We can adapt the coefficients $\vec{\alpha}^+$ or $\vec{\alpha}^-$ of the closest correct or incorrect prototype, by a stochastic gradient descent.

For sequence data, we can choose dissimilarities D according to pairwise alignments. We denote sequences over an alphabet Σ as $\bar{a} = (a_1, \dots, a_I, \dots, a_{|\bar{a}|})$ with $a_i \in \Sigma$ and length $|\bar{a}|$. Assume a symmetric dissimilarity measure $d_\lambda(a_i, a_j) = \lambda_{ij}$ is fixed on $(\Sigma \cup \{-\})^2$ with $d_\lambda(a_i, a_i) = 0$ and $d_\lambda(a_i, b_j) \geq 0$ for $a_i \neq b_j \in \Sigma \cup \{-\}$. A (global) *alignment* of sequences $\bar{a}$ and $\bar{b}$ consists of extensions $(\bar{a}^*, \bar{b}^*) \in ((\Sigma \cup \{-\})^*)^2$ by gaps such that $|\bar{a}^*| = |\bar{b}^*|$. *Alignment costs* are

$$d^*(\bar{a}, \bar{b}) = \min \left\{ \sum_{i=1}^{|\bar{a}^*|} d_\lambda(a_i^*, b_i^*) \;\middle|\; (\bar{a}^*, \bar{b}^*) \text{ is alignment of } (\bar{a}, \bar{b}) \right\} .$$

Setting $\bar{a}(I) = (a_1, \dots, a_I)$ and $\bar{b}(J) = (b_1, \dots, b_J)$, alignment costs can be computed by dynamic programming (DP) using the recursion

$$\begin{aligned}
d^*(\bar{a}(0), \bar{b}(0)) = 0, \quad & d^*(\bar{a}(0), \bar{b}(J)) = \textstyle\sum_{j \leq J} d_\lambda(-, b_j), \\
& d^*(\bar{a}(I), \bar{b}(0)) = \textstyle\sum_{i \leq I} d_\lambda(a_i, -), \\
d^*(\bar{a}(I+1), \bar{b}(J+1)) = \min\{ \quad & A_1 := d^*(\bar{a}(I), \bar{b}(J)) + d_\lambda(a_{I+1}, b_{J+1}), \\
& A_2 := d^*(\bar{a}(I+1), \bar{b}(J)) + d_\lambda(-, b_{J+1}), \\
& A_3 := d^*(\bar{a}(I), \bar{b}(J+1)) + d_\lambda(a_{I+1}, -) \ \} .
\end{aligned}$$

3 Adaptive scoring for alignments

Sequence alignment crucially depends on the scores λ of d_λ. Similar to [12], we propose an adaptation of λ based on the RGLVQ costs. Derivatives of the summand corresponding to a sequence $\bar{a}^i$ with respect to λ_{km} yield

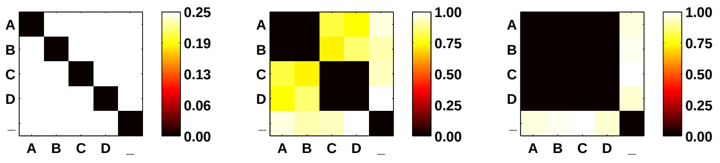
$$\Phi' \cdot \frac{2d^-(\bar{a}^i)}{(d^+(\bar{a}^i) + d^-(\bar{a}^i))^2} \cdot \frac{\partial d^+(\bar{a}^i)}{\partial \lambda_{km}} - \Phi' \cdot \frac{2d^+(\bar{a}^i)}{(d^+(\bar{a}^i) + d^-(\bar{a}^i))^2} \cdot \frac{\partial d^-(\bar{a}^i)}{\partial \lambda_{km}}$$

with $\partial d(\bar{a}^i, \bar{w}^j)/\partial \lambda_{km} = \sum_l \alpha_l^j \partial d_{il}^* / \partial \lambda_{km} - 0.5 \sum_{ll'} \alpha_l^j \alpha_{l'}^j \partial d_{ll'}^* / \partial \lambda_{km}$ where d_{il}^* refers to the alignment of sequences i and l. Alignment $d^*(\bar{a}, \bar{b})$ is not differentiable, but an approximation is, substituting min by
$\text{softmin}(x_1, \dots, x_n) = \sum_i x_i \cdot \exp(-\beta x_i) / \sum_j \exp(-\beta x_j)$ with the derivative
$\text{softmin}'(x_i) = (1 - \beta \cdot (x_i - \text{softmin})) \cdot \exp(-\beta x_i) / \sum_j \exp(-\beta x_j)$. The derivative $\partial d^*(\bar{a}, \bar{b}) / \partial \lambda_{km}$ can be computed in a DP scheme analog to the alignment:

$$\begin{aligned}
\tfrac{\partial d^*(\bar{a}(I+1), \bar{b}(J+1))}{\partial \lambda_{km}} &= \text{softmin}'(A_1) \cdot \left(\tfrac{\partial d^*(\bar{a}(I), \bar{b}(J))}{\partial \lambda_{km}} + \delta_k(a_{I+1}) \delta_m(b_{J+1}) \right) \\
&+ \text{softmin}'(A_2) \cdot \left(\tfrac{\partial d^*(\bar{a}(I+1), \bar{b}(J))}{\partial \lambda_{km}} + \delta_k(-) \delta_m(b_{J+1}) \right) \\
&+ \text{softmin}'(A_3) \cdot \left(\tfrac{\partial d^*(\bar{a}(I), \bar{b}(J+1))}{\partial \lambda_{km}} + \delta_k(a_{I+1}) \delta_m(-) \right)
\end{aligned}$$

where $\delta_k(a_i)$ tests whether the symbol a_i is element k in Σ.

We will investigate the role of the parameter β and efficient approximations of the computation in the experiments. For $\beta \to \infty$ the derivative $\partial d^*(\bar{a}, \bar{b}) / \partial \lambda_{km}$ converges to the number of times symbols number k and m



(a) Standard/init. λ (b) *Replacement* data (c) *Gap* data

Figure 1: Visualizations of the scoring matrix λ, where color/intensity encodes the values. On the left is a standard choice of λ which also serves as the initial state for the training, the middle and right show the final state of λ after adaptation.

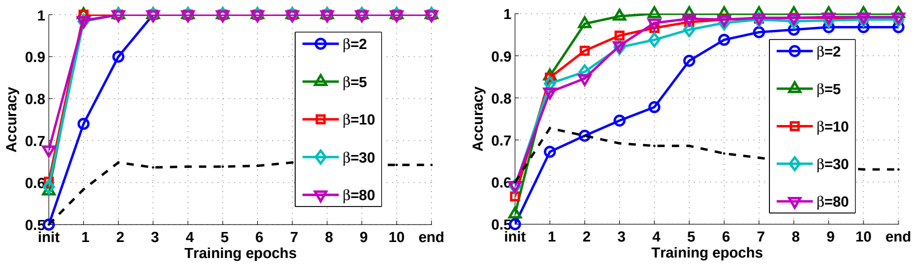
are paired in the alignment $(\bar{a}, \bar{b})$. For $\beta \to 0$, all three possible choices A_1, A_2, and A_3 are taken into account and no alignment paths stand out for the adaptation, resulting in homogeneous values λ_{km}. It is expected that optimal choices lie in between these extremes, corresponding to a good balance of exploitation of optimal alignment paths and exploration of competing alignment paths with similar quality. The latter is particularly relevant at the beginning of training and for large $|\Sigma|$. Because of the computational complexity, experiments have been reduced to the crisp case $\beta \to \infty$ only in [12]. Here, we will investigate different choices β, and will rely on the following two approximations for efficiency:

Approximation of prototypes by closest exemplars: $\partial d(\bar{a}^i, \bar{w}^j)/\partial\lambda_{km}$ refers to two sums with all coefficients α_l^j of $\bar{w}^j$. We use a `k`-approximation of the prototype which restricts to the closest `k` exemplars [7]. For the particularly interesting case `k` $= 1$, the derivative becomes $\partial d_{il}/\partial\lambda_{km}$, where $\bar{a}^l$ is the closest exemplar.

Dropping alignment paths with small contribution: In the limit $\beta \to \infty$, contributions restrict to the best alignment path, hence derivatives $\partial d^*(\bar{a}, \bar{b})/\partial\lambda_{km}$ for all λ_{km} can be computed in time $\mathcal{O}(|\bar{a}| + |\bar{b}|)$ based on the alignment matrix. In general, derivatives are weighted sums corresponding to alignments of the symbols m and l at some position (I, J) of the matrix. Weighting takes into account all possible paths which include this pair according to the path eligibility measured by $\text{softmin}'(A_i)$ for actions A_i on the path; the worst case complexity is $\mathcal{O}(|\bar{a}| \cdot |\bar{b}| \cdot |\Sigma|^2)$ using backtracing in the alignment matrix. We propose an approximation based on the observation that a small $\text{softmin}'(A_i)$ leads to a small weight of paths including A_i. Hence, we store the 3 terms A_1, A_2, A_3 together with the distances $\text{softmin}(A_1, A_2, A_3)$ in the matrix, and we cut all values $\text{softmin}'(A_i) < \theta$ for fixed $\theta \geq 0$. Backtracing depends on the nonzero values only, so that a speed-up to linear complexity is possible in the best case.

4 Experiments

We investigate the performance characteristics of RGLVQ using a fixed scoring matrix λ in comparison to adaptive scores λ based on the proposed approximations. First, we discuss the influence of the 'crispness' parameter β, and



(a) Different β for *Replacement* data (b) Different β for *Gap* data

Figure 2: The figures show how the crispness β affects the adaptation of λ and convergence of RGLVQ training. For different β, the mean test accuracy over all cross-validation runs is given in every learning epoch. The dashed black line represents RGLVQ training without adapting λ and serves as a baseline.

thereafter, the applicability and efficiency regarding real-world classification scenarios.

4.1 Artificial data

Replacement data: In this data set, all strings have 12 symbols randomly generated from the alphabet $\Sigma = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$ according to the regular expressions: $(\mathsf{A}|\mathsf{B})^5$ $(\mathsf{A}|\mathsf{B})$ $(\mathsf{C}|\mathsf{D})$ $(\mathsf{C}|\mathsf{D})^5$ for the first class, and $(\mathsf{A}|\mathsf{B})^5$ $(\mathsf{C}|\mathsf{D})$ $(\mathsf{A}|\mathsf{B})$ $(\mathsf{C}|\mathsf{D})^5$ for the second. Hence, replacements of A or B by C or D are discriminative, while replacements A with B, and C with D are not. After the training of λ, we expect high costs for discriminative replacements, while other replacement costs are close to zero. Also, we expect positive gap costs, since gaps could otherwise circumvent the alignment of the discriminative middle parts.

Gap data: The second data set focuses on gap scoring. Strings in the first class are random sequences $\bar{a}^i \in \Sigma^{10}$ of length 10, whereas strings $\bar{a}^l \in \Sigma^{12}$ in the second class are longer by 2 symbols. Therefore, replacements of letters are not discriminative, while the introduction of any gaps discriminates classes. Thus, gap costs should be high, while any symbol replacements should cost less.

Evaluation: For each data set, we created $N = 100$ sequences and evaluated the average classifier performance in a 5-fold cross-validation with 5 repeats. RGLVQ was trained using one prototype per class for 10 epochs. The learning rate for the adaptation of λ_{km} was set to $\eta = 1/N$ for replacement as well as gap scores. As initialization, we use a standard choice of $\lambda_{km} = 1/|\Sigma| \; \forall \, (k,m) \in (\Sigma \cup \{-\})^2, k \neq m$, and add small random noise to break ties in the initial alignments. All self-replacement scores remain fixed $\lambda_{kk} = 0$. During the adaptation, small or negative values $\lambda_{km} < \epsilon = 0.005$ are reset to ϵ in order to keep D non-negative.

The experimental results in Fig. 2 show the increased accuracy when adapting λ, e.g. for $\beta = 5$ a test accuracy of 100% (with 0 deviation) was achieved after the 4th epoch. Respectively, the adapted λ represent ideal scoring matrices for both data sets, which exactly fulfill our previously described expectations, see Fig. 1. In contrast, training RGLVQ with a fixed standard scoring λ remained close to a random guess throughout the learning epochs, see the baseline in Fig. 2.

We further evaluated how the 'crispness' parameter β influences the classifier and the training progress. In Fig. 2, we can see how a lower crispness (e.g. for $\beta = 2$) generally slows down the adaptation, while higher values seem to facilitate a faster convergence, sometimes at the expense of robustness (see $\beta = 80$ in Fig. 2b). Generally, we can observe that β directly affects the convergence characteristics, with an optimal value lying in a medium range.

4.2 Applicability and efficiency for real-world data sets

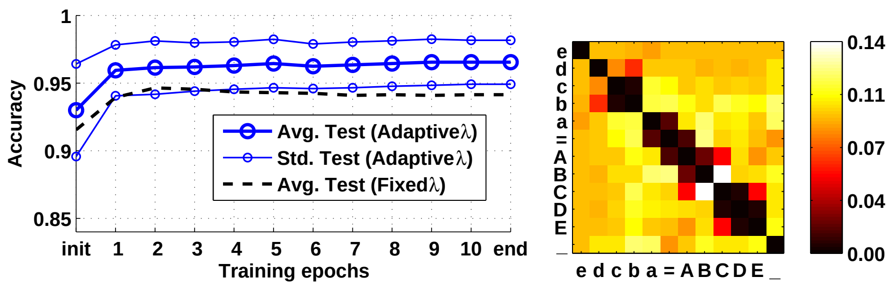
Chromosomes data: The sequences in this set represent band patterns from the Copenhagen Chromosomes database [11]. Every sequence encodes the differential succession of density levels observed in gray-scale images of a human chromosome. Since 7 levels of density are distinguished, a 13-letter alphabet $\Sigma = \{\mathsf{f}, \ldots, \mathsf{a}, =, \mathsf{A}, \ldots, \mathsf{F}\}$ represents a difference coding of successive positions, where upper and lower case letters mark positive and negative changes respectively, and "=" means no change[1]. From the database, we use a common benchmark set for binary classification, containing class 4 and 5, with 200 sequences each ($N = 400$). To handle the full 22-class data set, a local scoring matrix λ^j for every prototype $\vec{w}^j$ would be necessary, which is ongoing work, see Sec. 5.

The initial setup of λ was analog to the previous experiment, and one prototype per class was trained in a 5-fold cross-validation with 5 repeats. Crispness $\beta = 35$ was chosen, and $\eta_{\mathtt{Rep}} = 0.6 \cdot (1/N)$ was set for learning replacement costs λ_{km} and $\eta_{\mathtt{Gap}} = 0.4 \cdot (1/N)$ for gaps λ_{k-}. The results in Fig. 3 show an improvement of the average test accuracy by 3% after adaptation of λ. The ratio of mean intra-class distance to mean inter-class distance dropped from 0.94 to 0.91 in the adapted metric. Interestingly, λ shows a semantically meaningful pattern, with rather low values in the 1st and 2nd off-diagonals, which resembles the fact that density differences on neighboring scales are exchangeable within classes. (Note, that symbols f, F did not occur in the data and were thus not considered.)

[1]For details, see `http://algoval.essex.ac.uk/data/sequence/copchrom/`

Table 1: Runtimes (in minutes) to calculate the alignment derivative for all pairs of random strings $\bar{a}^i \in \Sigma^L$, $i \in \{1 \ldots 10\}$, using different thresholds θ and $\beta = 10$.

Sequence length L	100	150	200	250
Runtime ($\theta = 0$)	0.12	0.41	1.46	7.10
Runtime ($\theta = 0.15$)	0.07	0.22	0.53	1.64
Runtime ($\theta = 0.2$)	0.03	0.11	0.23	0.45
Runtime ($\theta = 0.25$)	0.02	0.06	0.12	0.22



(a) Classification accuracies (b) Adapted λ

Figure 3: Results for the *Chromosomes* data set, where the (semantically sound) adaptation of λ (right) yields an improvement of 3% in test accuracy (left).

Protein data: The sequences in this set are taken from a subset of the *SwissProt* database (release 37), which originally consists of 10,988 protein sequences in 32 classes. This subset was previously used in the context of RGLVQ classification, see [8]. For efficient testing of binary classification, we restrict here to the two classes with the lowest mean sequence lengths, using 617 sequences in total (class `4FE4S_FERREDOXIN` with 289 sequences, and `ADH_SHORT` with 323). RGLVQ training with 3 prototypes per class and distances based on the fixed standard scoring λ resulted in a test accuracy of 92%. Although the adaptation of λ lead to a slightly decreased 90% accuracy, it also produced a rather sparse scoring model: many values in λ are close to ϵ and only a smaller portion of the parameters influence the alignment with significant positive costs, namely $\left|\{(k,m)\,|\,\lambda_{km} > \epsilon + 0.01\}\right| = 488$ out of $650 = \left|(\Sigma \cup \{-\})^2\right|$ possible pairs. Looking at low-dimensional embeddings of the standard vs. the adapted distances (embedded by the t-SNE technique [19], see Fig. 4), we can observe that the main clusters of each class become more clearly distinguished by the adaptation. This is substantiated by the fact that the ratio of mean intra-class distance to mean inter-class distance decreases from 1.02 to 0.88. A detailed investigation about the semantic value of the sparse scoring model and increased class-separation is ongoing work.

Approximated alignment derivative for computational speedup: Since the calculation speed of derivatives $\partial d^*(\bar{a}, \bar{b})/\partial \lambda_{km}$ severely affects the overall runtime, we empirically evaluate the speedup by the approximation proposed in the end of Section 3. The threshold θ determines that values $\mathrm{softmin}'(A_i) < \theta$ are ignored in the backtracing of alignment paths. Since the impact of θ depends on the alphabet size and sequence length, it should be tuned according to good classification results for the given data set. Typical values are $\theta \in (0.01, 0.2)$. As a simple test scenario, we created several sets of random sequences, each consisting of 10 sequences $\bar{a}^i \in \Sigma^L$ with $\Sigma = \{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}\}$, for different choices of length L. For different thresholds θ, we tracked the runtime of calculating alignment derivatives for all 100 sequence pairs on a standard laptop computer with an *Intel Core i7* processor (4 cores, and calculations done in parallel). The results in Tab. 1 clearly show how increasing θ drastically reduces the computational effort, especially for longer sequences. At the same time, approximation to a certain extent does not reduce classifier performance: average test accuracy on the *Chromosomes* data remained at 97% for $\theta = 0.02$, decreasing the mean runtime by 7%.

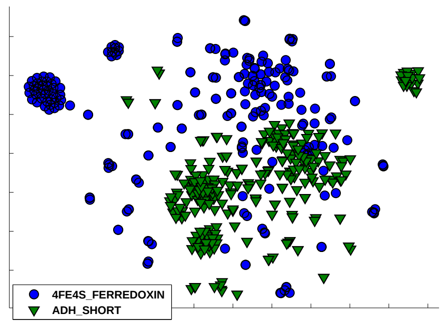
5 Discussion

We presented a technique to integrate the supervised adaptation of metric parameters (in this case the scoring pattern for sequence alignments) into a LVQ-based classifier framework. Specifically, we utilized RGLVQ as an overarching learning regime, which is able to process (dis)similarity data [7]. The goal is to facilitate class discrimination in the adapted dissimilarities, while the training of prototypes yields a sparse classification model for the data. Unlike in [9], we do not assume differentiability of the dissimilarity measure with respect to the data structures itself, but differentiability with respect to the metric parameters only. Therefore, our approach could serve as a generic foundation for metric adaptation schemes using dissimilarity measures for discrete structures such as sequences or, as a generalization, trees or graph structures. In addition to an improved class separation in adapted distances, the learned scoring could highlight the importance of structural replacement operations, and thus give further insight into the classification model. In the experiments, we demonstrated the viability of our method, and evaluated the influence of the crispness β, along with the computational speedup by an approximation technique. The adaptation of λ revealed semantically interesting symbolic scoring patterns, a more detailed analysis being the subject of future work. Since one parameter set λ affects the global metric in the data space, it could be beneficial to use class-specific scoring matrices λ^j, e.g. for every LVQ prototype, similar to local metric learning for vectorial data [16]. In terms of efficiency, a current limitation is the inherent dependency of RLVQ on the entire dissimilarity matrix D, which changes entirely if λ is adapted. Therefore, one could refer to low-rank techniques to approximate D based on a small number of landmark sequences.

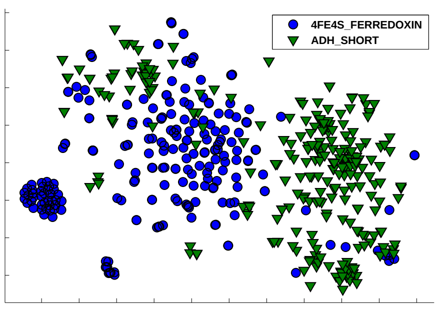
References

[1] M. Bernard, L. Boyer, A. Habrard, and M. Sebban. Learning probabilistic models of tree edit distance. *Pattern Recognition*, 41(8):2611–2629, 2008.

[2] L. Boyer, Y. Esposito, A. Habrard, J. Oncina, and M. Sebban. Sedil: Software for edit distance learning. *Lect. Notes Comput. Sci.*, 5212 LNAI(2):672–677, 2008.



(a) Distances with standard scoring



(b) Distances with adapted scoring

Figure 4: t-SNE visualizations of pairwise distances from the *Protein* data set: the adapted scoring (right) yields more class separation than standard scoring (left).

[3] T. Gärtner, G. Garriga, and T. Meinl, editors. *Proc. of The Workshop on Mining and Learning with Graphs*, 2006.

[4] A. Gisbrecht, B. Mokbel, and B. Hammer. Relational generative topographic mapping. *Neurocomputing*, 74(9):1359–1371, 2011.

[5] A. Habrard, J. Iñesta, D. Rizo, and M. Sebban. Melody recognition with learned edit distances. *Lect. Notes Comput. Sci.*, 5342 LNCS:86–96, 2008.

[6] B. Hammer and A. Hasenfuss. Topographic mapping of large dissimilarity data sets. *Neural Computation*, 22(9):2229–2284, 2010.

[7] B. Hammer, D. Hofmann, F.-M. Schleif, and X. Zhu. Learning vector quantization for (dis-)similarities. *Neurocomputing*, 2013. In Press.

[8] B. Hammer, B. Mokbel, F.-M. Schleif, and X. Zhu. White box classification of dissimilarity data. In *HAIS*, volume 7208 of *LNCS*, pages 309–321. Springer, 2012.

[9] M. Kästner, D. Nebel, M. Riedel, M. Biehl, and T. Villmann. Differentiable kernels in generalized matrix learning vector quantization. In *ICMLA, p. 132-137*, 2012.

[10] S. Kirstein, A. Denecke, S. Hasler, H. Wersing, H.-M. Gross, and E. Körner. A vision architecture for unconstrained and incremental learning of multiple categories. *Memetic Computing*, 1(4):291–304, 2009.

[11] C. Lundsteen, J. Phillip, and E. Granum. Quantitative analysis of 6985 digitized trypsin G-banded human metaphase chromosomes. *Clin. Genet.*, 18:355–370, 1980.

[12] B. Mokbel, B. Paassen, and B. Hammer. Adaptive distance measures for sequential data. In M. Verleysen, editor, *ESANN*, pages 265–270. i6doc.com, 2014.

[13] B. Mokbel, B. Paassen, and B. Hammer. Efficient adaptation of structure metrics in prototype-based classification. In S. Wermter, C. Weber, W. Duch, T. Honkela, P. D. Koprinkova-Hristova, S. Magg, G. Palm, and A. E. P. Villa, editors, *Artificial Neural Networks and Machine Learning - ICANN - 24th International Conference on Artificial Neural Networks, Hamburg, Germany, September 15-19, 2014*, volume 8681 of *Lecture Notes in Computer Science*, pages 571–578. Springer, 2014.

[14] E. Pekalska and B. Duin. *The Dissimilarity Representation for Pattern Recognition. Foundations and Applications.* World Scientific, 2005.

[15] F.-M. Schleif, B. Hammer, M. Kostrzewa, and T. Villmann. Exploration of mass-spectrometric data in clinical proteomics using learning vector quantization methods. *Briefings in Bioinformatics*, 9(2):129–143, 2008.

[16] P. Schneider, M. Biehl, and B. Hammer. Adaptive relevance matrices in learning vector quantization. *Neural Computation*, 21(12):3532–3561, 2009.

[17] V. Sperschneider. *Bioinformatics.* Springer, 2008.

[18] A. Takasu, D. Fukagawa, and T. Akutsu. Statistical learning algorithm for tree similarity. In *IEEE Int. Conf. on Data Mining, ICDM*, pages 667–672, 2007.

[19] L. van der Maaten and G. Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.