Maximum-Entropy Parameter Estimation for the k-nn Modified Value-Difference Kernel

Iris Hendrickx Antal van den Bosch

ILK / Computational Linguistics and AI, Tilburg University
P.O. Box 90153, NL-5000 LE Tilburg, The Netherlands

Abstract

We introduce an extension of the modified value-difference kernel of k-nn by replacing the kernel's default class distribution matrix with the matrix produced by the maximum-entropy learning algorithm. This hybrid algorithm is tested on fifteen machine learning benchmark tasks, comparing the hybrid to standard k-nn classification and maximum-entropy-based classification. Results show that the hybrid typically outperforms the lower-scoring of the two other algorithms, often significantly; in a majority of cases the hybrid yields the highest accuracy of the three algorithms. Error analysis indicates that the hybrid's errors overlap more with k-nn than with maximum entropy modeling.

1 Introduction

In this paper we introduce an extension to the Modified Value Difference kernel of the k-nn (k nearest neighbour) machine learning classifier [18, 5]. This kernel operates on a probability matrix, which by default is an unsmoothed class distribution matrix. We attempt to improve on this matrix by replacing it with the matrix as produced by maximum-entropy learning. The goal of this type of study is to gain more insight into the differences and commonalities of competing machine learning algorithms, and possibly gain from this exercise by producing a *bridge* algorithm that combines the best of both worlds.

Comparing the original two algorithms with a bridge algorithm offers a potential to exceed the scope of the typical comparative machine learning study in which existing algorithms are quantitatively compared, and in which the outcome can only demonstrate which of the algorithms is the "better" one in the given experimental setup. Also, by transposing a component of one algorithm into another, in this case the probability matrix of the maximum entropy classifier, we can study this component separately and test whether it is still valuable when pulled out of its intended context (for another example of a similar study with a bridge algorithm, cf. [19]).

The possibility to build a bridge algorithm between k-nn and maximum - entropy modeling by transplanting a matrix stems from the fact that k-nn can use probabilities, namely through probabilistic kernels such as the one based on the Modified Value Difference Metric (henceforth referred to as MVDM) [18, 5].

Maximum-entropy [10, 2] operates on a matrix that has the same two dimensions as the matrix central to the MVDM kernel, namely *feature values* and *class labels*. Naive Bayes classifiers [12] and linear threshold algorithms with one output unit per class, such as perceptrons [17] and Winnow networks [14] have this very same matrix at their core. Generally, the cells in these tables are filled with feature weights, which are used directly in classification. In contrast, a k-nn classifier with the MVDM kernel uses this value–class matrix to estimate numerical distances between pairs of symbolic feature values, to play a role in the general distance function at the heart of the k-nn classifier.

The raw probabilities used in this kernel can be unreliable when data is sparse. This problem can be tackled by smoothing techniques such as described by [4]. In this research, however, we have chosen a different approach as we build a bridge algorithm.

This paper is structured as follows. In Section 2 the MVDM kernel is described, as well as the transposition operation. We measure the effect of this transposition by performing cross-validation experiments on a set of machine learning benchmark tasks. Section 3 details the data used in the comparative experiments, as well as the experimental setup. The results of these experiments are given in Section 4. They are discussed in Section 5. We summarize our findings in Section 6.

2 The Modified Value Difference Metric kernel

The k-nn classifier [9, 6], re-introduced and revitalised in machine learning under the name IB1 [1] is a supervised inductive learning algorithm for learning classification tasks. The k-nn classifier treats a set of labeled training examples as points in a multi-dimensional feature space, and stores them as such in an *instance base* in memory (rather than performing some abstraction over them). Classification in memory-based learning is performed by the k-nn classification rule that searches for the k nearest-neighbor examples to the new instance to be classified, according to the distance function between two instances X and Y in Equation 1:

$$\Delta(X, Y) = \sum_{i=1}^{n} w_i \; \delta(x_i, y_i) \tag{1}$$

where n is the number of features, w_i is a global weight for feature i, and δ estimates the difference between the two instances' values at the ith feature. The classes of the k nearest neighbors then determine the class of the new case.

Some simple distance kernels for the k-nn classifier, such as the Overlap metric [1] in Equation 2:

$$\delta(x_i, y_i) = \begin{cases} abs(\frac{x_i - y_i}{max_i - min_i}) & \text{if numeric, else} \\ 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases} \tag{2}$$

fail to capture relative differences among matches between symbolic values; they treat all values of a feature as equally dissimilar – instances match on the same feature only if they have the same value at that feature (hence the "Overlap").

To allow relative differences between symbolic values to be acknowledged by the classifier, the (Modified) Value Difference metric (MVDM) was introduced in [18] and further refined by [5]. MVDM estimates the distance between two values v_1, v_2, as the difference of the conditional distribution of the classes C_i, as estimated from the training set:

$$\delta(v_1, v_2) = \sum_{i=1}^{k} |P(C_i|v_1) - P(C_i|v_2)| \tag{3}$$

A known weakness of the MVDM metric is its sensitivity to data sparseness. In many practical situations limited sets of examples are available, with values occuring only a few times or once in the whole data set. If two such values occur with the same class, MVDM will regard them as identical, and if they occur with two different classes their distance will be maximal (2.0 according to Equation 3). Re-estimations of the probabilities could enhance the MVDM kernel. One such re-estimation of the value–class matrix is the one produced by the maximum-entropy learning algorithm.

Maximum-entropy modeling [10, 2] is a supervised statistical learning algorithm that re-estimates a raw probability matrix based on empirical co-occurrence counts of classes and values (estimating $P(C, v)$ for all co-occurring C and v). Maximum-entropy modeling adapts the probabilities so that they maximize the entropy of the matrix, representing the highest level of uncertainty beyond its constraints, i.e. the evidence from the training data. Searching for the matrix with the maximum entropy is done in a iterative way using algorithms such as Generalized Iterative Scaling [8] and L-BFGS [15]. When produced, the matrix can be transplanted into the MVDM kernel, replacing all cells with raw estimations of $P(C|v)$ with maximum-entropy 's re-estimated $P(C, v)$ probabilities . We henceforth call the hybrid of k-nn with the MVDM kernel and maximum-entropy modeling probabilities MEMBL (for 'maximum-entropy memory-based learner'). Since maximum-entropy classification invests considerable effort in obtaining re-estimated, more useful and reliable probabilities, and k-nn simply uses unsmoothed probabilities derived from straight counts, the MVDM kernel might improve from this operation.

3 Method

3.1 Experimental setup

We tested the hybrid algorithm MEMBL, maximum-entropy modeling and k-nn and measured their performance on 15 different data sets. These data sets were randomized and split ten times in a 90% training part and 10% test part, so that 10-fold cross-validation experiments could be run. We calculated the mean and standard deviation of the accuracies on the ten splits. We also conducted paired t-tests between all pairs of algorithms to evaluate whether one is significantly better than the other on a certain task.

We selected 15 datasets from the UCI repository of machine learning databases [3]. These data sets have in common that they only contain symbolic features.

Task	Number of examples	features	classes	Class entropy
audiology	228	69	24	3.41
bridges	110	7	8	2.50
car evaluation	1,730	6	4	1.21
connect4	67,559	42	3	1.22
kr-vs-kp	3,197	36	2	1.00
lung-cancer	32	56	2	0.86
monks1	558	6	2	1.00
monks2	603	6	2	0.93
monks3	556	6	2	1.00
nursery	12,961	8	5	1.72
promoters	106	57	2	1.00
soybean-large	685	35	19	3.84
splice	3,192	60	3	1.48
tic-tac-toe	960	9	2	0.93
votes	437	16	2	0.96

Table 1: Basic statistics of the 15 UCI repository data sets.

Table 3.1 gives details on the numbers of features, classes, and examples available for these tasks.

The k-nn experiments were performed with the TiMBL software package [7], which can emulate IB1 with MVDM. The number of nearest neighbours, the feature weighting and distance weighting were optimized for each experiment individually by wrapped progressive sampling (WPS) [20]. WPS combines classifier wrapping [11] with progressive sampling of training material [16]. WPS starts with a large pool of experiments, each with one systematically generated recombination of tested algorithmic parameter settings. In the first step of WPS, each attempted setting is applied to a small amount of training material and tested on a small amount of held-out training data. Only the best settings are kept; all others are removed from the pool of competing settings. In subsequent iterations, this step is repeated, retaining only the best-performing settings, with an exponentially growing amount of training and held-out data – until all training data is used or one best setting is left. Selecting the best settings at each step is based on classification accuracy on the held-out data; a simple one-dimensional clustering on the ranked list of accuracies determines which group of settings is selected for the next iteration. The result of the WPS procedure, applied to every training partition of each 10-fold cross-validation experiment, yields one estimated-best algorithmic parameter setting which is then applied to the full 90% training set, and tested on the 10% test set.

For Maximum-entropy modeling we have used MAXENT, a Maximum Entropy toolkit of Zhang Le (version 20040315), with L-BFGS parameter estimation, 100 iterations and a Gaussian prior of 1.0. [13].

Experiments with MEMBL were done with the same algorithm and parameter

task	k-nn			MEMBL			MAXENT		
audiology	80.6	± 7.8		**81.0**	± 8.8		79.3	± 7.4	
bridges	51.8	± 10.8		54.5	± 8.1		**60.9**	± 10.8	>knn
car evaluation	**96.1**	± 1.3	>>max	**96.1**	± 1.1	>>max	90.3	± 1.8	
connect4	77.2	± 1.6	>max	**77.5**	± 1.5	>>max	75.7	± 0.4	
kr-vs-kp	96.9	± 0.6		**98.2**	± 1.0	>>max,>>knn	96.9	± 0.6	
lung-cancer	53.3	± 27.9		**56.7**	± 27.3		41.7	± 22.4	
monks1	**91.8**	± 10.4	>>max	91.6	± 9.8	>>max	74.5	± 5.0	
monks2	**79.4**	± 15.3	>max	79.1	± 15.6	>max	62.3	± 5.5	
monks3	**98.9**	± 0.9	>>max	**98.9**	± 0.9	>>max	96.6	± 1.7	
nursery	**98.8**	± 0.8	>>max	98.0	± 1.0	>>max	92.4	± 0.5	
promoters	89.6	± 7.7		90.5	± 8.8		**92.4**	± 5.9	
soybean-large	**94.6**	± 2.2	>>max	**94.6**	± 2.5	>max	92.3	± 2.7	
splice	94.5	± 1.5		94.1	± 1.7		**94.7**	± 1.3	
tic-tac-toe	92.7	± 3.2		**98.9**	± 1.8	>>knn	98.2	± 0.9	>>knn
votes	95.7	± 2.8		95.4	± 2.5		**95.9**	± 3.2	

Table 2: Generalization performances (average accuracies on test data, with standard deviations) on the 15 data sets by k-nn, MEMBL, and MAXENT. '>' indicates significance in a paired t-test with $p < 0.05$ and '>>' indicates an uncertainty of $p < 0.01$.

settings as used in the k-nn experiments, but with the value–class matrix produced by MAXENT replacing the standard matrix. We have used the model file as produced by MAXENT, converted this into a format suitable for TiMBL, and scaled the matrix values to numbers between 0 and 1.

4 Results

Table 2 displays the experimental results (average percentages of correctly classified test instances) of k-nn, MAXENT and MEMBL on the fifteen UCI benchmark tasks. The symbols >> and > indicate the cases where a significant improvement was found according to a paired t-test. For example, the abbreviation '>>max' stands for 'this particular classifier performed significantly better than MAXENT with an uncertainty of $p < 0.01$'.

In 8 out of 15 tasks, MEMBL performs significantly better than MAXENT; on two tasks it performs better than k-nn. The k-nn classifier significantly outperforms MAXENT 7 times, while MAXENT outperforms k-nn twice.

5 Discussion

The generalization performance results in Table 2 show several significant improvements of MEMBL over maximum-entropy modeling. Interestingly, neither MAXENT nor k-nn significantly outperforms MEMBL on any of the tasks, suggesting that MEMBL indeed combines the best of both worlds.

These results do not tell, however, whether transplanting the maximum-entropy matrix in the k-nn classifier makes MEMBL perform more similar to k-nn or to MAXENT. To get insight into the extent to which MEMBL deviates from k-nn and MAXENT, we counted the numbers of times each pair of algorithms assigned a different label to test instances. These counts, aggregated over all 10 test partitions for each task, are listed in Table 3, in columns 2, 3, and 4.

Task	k-nn – MAXENT	MEMBL – MAXENT	MEMBL – k-nn	% Error overlap MEMBL – k-nn	MEMBL – MAXENT
audiology	28	31	23	65.1	55.8
bridges	29	28	14	80.0	50.0
car evaluation	167	162	16	88.1	50.7
connect4	12488	11697	5828	78.2	59.4
kr-vs-kp	129	119	60	82.8	32.8
lung-cancer	12	12	3	85.7	71.4
monks1	128	127	21	76.6	66.0
monks2	195	191	16	92.9	64.3
monks3	13	13	0	100.0	100.0
nursery	963	996	219	38.3	47.3
promoters	9	12	7	70.0	30.0
soybean-large	40	37	14	81.1	67.6
splice	174	198	90	70.4	40.2
tic-tac-toe	85	18	77	18.2	45.5
votes	17	16	3	90.0	55.0

Table 3: The number of times a different label was assigned by the algorithms. The rightmost two columns show the percentage of overlap in mistakes between MEMBL and k-nn, and MEMBL and MAXENT.

The numbers indicate that the overlap between the labeling of k-nn and MEMBL is much higher than between MEMBL and MAXENT. In 12 out of 15 cases, the number of different labels assigned between MAXENT and MEMBL is less than half the number of different labels assigned between k-nn and MEMBL.

We also took the instances that were mislabeled by MEMBL, and counted the number of times when k-nn or MAXENT gave these instances the same incorrect label – in other words, we counted the number of cases in which both MEMBL and the other algorithm made the same error. The percentage of this number in the total number of errors made by MEMBL is shown in column 5 for k-nn, and in column 6 for MAXENT. Although MEMBL shares a considerable portion of its errors with MAXENT, it shares more errors with k-nn.

6 Conclusion

We introduced MEMBL, a hybrid of k-nn and maximum-entropy modeling. MEMBL is a k-nn classifier equipped with the MVDM kernel, in which the value–class matrix is the one produced by a maximum-entropy classifier on the same training material, rather than the standard matrix of conditional class distributions. In a series of experiments with 15 UCI benchmark datasets, MEMBL performed significantly better than k-nn on two tasks, and performed significantly better than maximum-entropy modeling in eight tasks. It never performed signficantly worse than either of the two other algorithms. These outcomes support a tentative conclusion that

MEMBL indeed combines the best of both worlds.

The differences between MEMBL and k-nn are smaller than the differences between MEMBL and maximum-entropy modeling; this can be concluded on the basis of the additional statistics we computed on overlapping disagreements in classifications and in overlapping errors. This observation can be explained by the fact that MEMBL is still a k-nn classifier, with biases (especially, global feature weighting and the nearest-neighbor classification step) that are alien to maximum-entropy classification.

Still, the results can be taken to imply that the value–class matrix of maximum-entropy modeling might be better used for computing distances between pairs of values in a distance-based classifier, than in a probabilistic classifier that uses the maximum-entropy probabilities directly.

Acknowledgements

This research is funded by the Netherlands Organisation for Scientific Research (NWO). The authors wish to thank Zhang Le for sharing information on the internals of his MAXENT implementation.

References

[1] D. W. Aha, D. Kibler, and M. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] A. Berger, S. Della Pietra, and V. Della Pietra. Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1), 1996.

[3] C. Blake and C.J. Merz. UCI repository of machine learning databases, 1998. `http://www.ics.uci.edu/ mlearn/MLRepository.html`.

[4] S. F. Chen, D. Beeferman, and R. Rosenfeld. Evaluation metrics for language models. In *DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

[5] S. Cost and S. Salzberg. A weighted nearest neighbour algorithm for learning with symbolic features. *Machine Learning*, 10:57–78, 1993.

[6] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, 13:21–27, 1967.

[7] W. Daelemans, J. Zavrel, K. Van der Sloot, and A. Van den Bosch. TiMBL: Tilburg memory based learner, version 5.0, reference guide. ILK Technical Report 03-10, Tilburg University, 2003.

[8] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

[9] E. Fix and J. L. Hodges. Disciminatory analysis—nonparametric discrimination; consistency properties. Technical Report Project 21-49-004, Report No. 4, USAF School of Aviation Medicine, 1951.

[10] S. Guiasu and A. Shenitzer. The principle of maximum entropy. *The Mathematical Intelligencer*, 7(1), 1985.

[11] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence Journal*, 97(1–2):273–324, 1997.

[12] P. Langley, W. Iba, and K. Thompson. An analysis of bayesian classifiers. In *Proceedings of the Tenth Annual Conference on Artificial Intelligence*, pages 223–228. AAAI Press and MIT Press, 1992.

[13] Zhang Le. *Maximum Entropy Modeling Toolkit for Python and C++*. Natural Language Processing Lab, Northeastern University, China, 2004. `http://www.nlplab.cn/zhangle/software/maxent/manual/`.

[14] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[15] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35:773–782, 1980.

[16] F. Provost, D. Jensen, and T. Oates. Efficient progressive sampling. In *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pages 23–32, 1999.

[17] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organisation in the brain. *Psychological Review*, 65:368–408, 1958.

[18] C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the* ACM, 29(12):1213–1228, December 1986.

[19] A. Van den Bosch. Feature transformation through rule induction: A case study with the k-nn classifier. In *Proceedings of the workshop Advances in Inductive Rule Learning*, 2004.

[20] A. Van den Bosch. Wrapped progressive sampling search for optimizing learning algorithm parameters. In *Proceedings of the 16th Belgian-Dutch Conference on Machine Learning*, 2004.