In: *Proceedings of CoNLL-2000 and LLL-2000*, pages 55–60, Lisbon, Portugal, 2000.

Learning Distributed Linguistic Classes

Stephan Raaijmakers
Netherlands Organisation for Applied Scientific Research (TNO)
Institute for Applied Physics
Delft
The Netherlands
raaijmakers@tpd.tno.nl

Abstract

Error-correcting output codes (ECOC) have emerged in machine learning as a successful implementation of the idea of distributed classes. Monadic class symbols are replaced by bit strings, which are learned by an ensemble of binary-valued classifiers (dichotomizers). In this study, the idea of ECOC is applied to memory-based language learning with local (k-nearest neighbor) classifiers. Regression analysis of the experimental results reveals that, in order for ECOC to be successful for language learning, the use of the Modified Value Difference Metric (MVDM) is an important factor, which is explained in terms of population density of the class hyperspace.

1 Introduction

Supervised learning methods applied to natural language classification tasks commonly operate on high-level symbolic representations, with linguistic classes that are usually monadic, without internal structure (Daelemans et al., 1996; Cardie et al., 1999; Roth, 1998). This contrasts with the distributed class encoding commonly found in neural networks (Schmid, 1994). Error-correcting output codes (ECOC) have been introduced to machine learning as a principled and successful approach to *distributed* class encoding (Dietterich and Bakiri, 1995; Ricci and Aha, 1997; Berger, 1999). With ECOC, monadic classes are replaced by *codewords*, i.e. binary-valued vectors. An ensemble of separate classifiers (dichotomizers) must be trained to learn the binary subclassifications for every instance in the training set. During classification, the bit predictions of the various dichotomizers are combined to produce a codeword prediction. The class codeword which has minimal Hamming distance to the predicted codeword determines the classification of the instance. Codewords are constructed such that their Hamming distance is maximal. Extra bits are added to allow for error recovery, allowing the correct class to be determinable even if some bits are wrong. An error-correcting output code for a k-class problem constitutes a matrix with k rows and $2^{k-1}-1$ columns. Rows are the codewords corresponding to classes, and columns are binary subclassifications or bit functions f_i such that, for an instance $\mathbf{e}$, and its codeword vector $\mathbf{c}$

$$f_i(\mathbf{e}) = \pi_i(\mathbf{c}) \tag{1}$$

($\pi_i(v)$ the i-th coordinate of vector v). If the minimum Hamming distance between every codeword is d, then the code has an error-correcting capability of $\lfloor \frac{d-1}{2} \rfloor$. Figure 1 shows the 5×15 ECOC matrix, for a 5-class problem. In this code, every codeword has a Hamming distance of at least 8 to the other codewords, so this code has an error-correcting capability of 3 bits. ECOC have two natural interpretations. From an information-theoretic perspective, classification with ECOC is like *channel coding* (Shannon, 1948): the class of a pattern to be classified is a datum sent over a noisy communication channel. The communication channel consists of the trained classifier. The noise consists of the bias (systematic error) and variance (training set-dependent error) of the classifier, which together make up for the overall error

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Figure 1: ECOC for a five-class problem.

of the classifier. The received message must be decoded before it can be interpreted as a classification. Adding redundancy to a signal before transmission is a well-known technique in digital communication to allow for the recovery of errors due to noise in the channel, and this is the key to the success of ECOC. From a machine learning perspective, an error-correcting output code uniquely partitions the instances in the training set into two disjoint subclasses, 0 or 1. This can be interpreted as learning a set of class boundaries. To illustrate this, consider the following binary code for a three-class problem. (This actually is a one-of-c code with no error-correcting capability (the minimal Hamming distance between the codewords is 1). As such it is an error-correcting code with lowest error correction, but it serves to illustrate the point.)

	f_1	f_2	f_3
C1	0	0	1
C2	0	1	0
C3	1	0	0

(2)

For every combination of classes (C1–C2, C1–C3, C2–C3), the Hamming distance between the codewords is 2. These horizontal relations have vertical repercussions as well: for every such pair, two bit functions disagree in the classes they select. For C1–C2, f_2 selects C2 and f_3 selects C1. For C1–C3, f_1 selects C3 and f_3 selects C1. Finally, for C2–C3, f_1 selects C3 and f_2 selects C2. So, every class is selected two times, and this implies that every class boundary associated with that class in the feature hyperspace is learned twice. In general (Kong and Dietterich, 1995), if the minimal Hamming distance between the codewords of an (error-correcting) code is d, then every class boundary is learned d times. For the error-correcting code from above this implies an error correction of zero: only two votes support a class boundary, and no vote can be favored in case of a conflict. The decoding of the predicted bit string to a class symbol appears to be a form of voting over class boundaries (Kong and Dietterich, 1995), and is able to reduce both bias and variance of the classifier.

2 Dichotomizer Ensembles

Dichotomizer ensembles must be diverse apart from accurate. Diversity is necessary in order to decorrelate the predictions of the various dichotomizers. This is a consequence of the voting mechanism underlying ECOC, where bit functions can only outvote other bit functions if they do not make similar predictions. Selecting different features per dichotomizer was proposed for this purpose (Ricci and Aha, 1997). Another possibility is to add limited non-locality to a local classifier, since classifiers that use global information such as class probabilities during classification, are much less vulnerable to correlated predictions. The following ideas were tested empirically on a suite of natural language learning tasks.

- A careful feature selection approach, where every dichotomizer is trained to select (possibly) different features.
- A careless feature selection approach, where every bit is predicted by a voting committee of dichotomizers, each of which randomly selects features (akin in spirit to the Multiple Feature Subsets approach for non-distributed classifiers (Bay, 1999).
- A careless feature selection approach, where blocks of two adjacent bits are predicted by a voting committee of *quadrotomizers*, each of which randomly selects features. Learning blocks of two bits allows for bit codes that are twice as long (larger error-correction), but with half as many classifiers. Assuming a normal distribution of errors and bit values in every 2 bits-block, there is a 25% chance that both bits in a 2-bit block are wrong. The other 75% chance of one bit wrong would produce performance equal to voting per bit. Formally, this implies a switch from N two-class problems to $N/2$ four-class problems, where separate regions of the class landscape are learned jointly.
- Adding non-locality to 1-3 in the form of larger values for k.
- The use of the Modified Value Difference Metric, which alters the distribution of instances over the hyperspace of features, yielding different class boundaries.

3 Memory-based learning

The memory-based learning paradigm views cognitive processing as reasoning by analogy. Cognitive classification tasks are carried out by matching data to be classified with classified data stored in a knowledge base. This latter data set is called the training data, and its elements are called *instances*. Every instance consists of a feature-value vector and a class label. Learning under the memory-based paradigm is *lazy*, and consists only of storing the training instances in a suitable data structure. The instance from the training set which resembles the most the item to be classified determines the classification of the latter. This instance is called the nearest neighbor, and models based on this approach to analogy are called *nearest neighbor models* (Duda and Hart, 1973). So-called k-nearest neighbor models select a winner from the k nearest neighbors, where k is a parameter and winner selection is usually based on class frequency. Resemblance between instances is measured using distance metrics, which come in many sorts. The simplest distance metric is the overlap metric:

$$\begin{aligned} \Delta(I_i, I_j) &= \sum_k \delta(\pi_k(I_i), \pi_k(I_j)) \\ \delta(v_i, v_j) &= 0 \text{ if } v_i = v_j \\ \delta(v_i, v_j) &= 1 \text{ if } v_i \neq v_j \end{aligned} \tag{3}$$

($\pi_i(I)$ is the i-th projection of the feature vector I.) Another distance metric is the Modified Value Difference Metric (MVDM) (Cost and Salzberg, 1993). The MVDM defines similarity between two feature values in terms of posterior probabilities:

$$\delta(v_i, v_j) = \sum_{c \in Classes} \mid P(c \mid v_i) - P(c \mid v_j) \mid \tag{4}$$

When two values share more classes, they are more similar, as δ decreases. Memory-based learning has fruitfully been applied to natural language processing, yielding state-of-the-art performance on all levels of linguistic analysis, including grapheme-to-phoneme conversion (van den Bosch and Daelemans, 1993), PoS-tagging (Daelemans et al., 1996), and shallow parsing (Cardie et al., 1999). In this study, the following memory-based models are used, all available from the TiMBL package (Daelemans et al., 1999). IB1-IG is a k-nearest distance classifier which employs a *weighted* overlap metric:

$$\Delta(I_i, I_j) = \sum_k w_k \delta(\pi_k(I_i), \pi_k(I_j)) \tag{5}$$

In stead of drawing winners from the k-nearest neighbors pool, IB1-IG selects from a pool of instances for k nearest distances. Features are separately weighted based on Quinlan's information gain ratio (Quinlan, 1993), which measures the informativity of features for predicting class labels. This can be computed by subtracting the entropy of the knowledge of the feature values from the general entropy of the class labels. The first quantity is normalized with the *a priori* probabilities of the various feature values of feature F:

$$H(C) - \textstyle\sum_{v \in Values(F)} P(v) \times H(C_{[F=v]}) \tag{6}$$

Here, $H(C)$ is the class entropy, defined as

$$H(C) = - \sum_{c \in Class} P(c) \log_2 P(c). \tag{7}$$

$H(C_{[F=v]})$ is the class entropy computed over the subset of instances that have v as value for F_i. Normalization for features with many values is obtained by dividing the information gain for a feature by the entropy of its value set (called the *split info* of feature F_i.

$$\begin{gathered} w_i = \frac{H(C) - \sum_{v \in Values(F_i)} P(v) \times H(C_{[F=v]})}{split{-}info(F_i)} \\ split - info(F_i) = - \sum_{v \in Values(F_i)} P(v) \log_2 P(v) \end{gathered} \tag{8}$$

IGTREE is a heuristic approximation of IB1-IG which has comparable accuracy, but is optimized for speed. It is insensitive to k-values larger than 1, and uses value-class cooccurrence information when exact matches fail.

4 Experiments

The effects of a distributed class representation on generalization accuracy were measured using an experimental matrix based on 5 linguistic datasets, and 8 experimental conditions, addressing feature selection-based ECOC vs. voting-based ECOC, MVDM, values of k larger than 1, and dichotomizer weighting. The following linguistic tasks were used. DIMIN is a Dutch diminutive formation task derived from the Celex lexical database for Dutch (Baayen et al., 1993). It predicts Dutch nominal diminutive suffixes from phonetic properties (phonemes and stress markers) of maximally the last three syllables of the noun. The STRESS task, also derived from the Dutch Celex lexical database, assigns primary stress on the basis of phonemic values. MORPH assigns morphological boundaries (a.o. root morpheme, stress-changing affix, inflectional morpheme), based on English CELEX data. The WSJ-NPVP task deals with NP-VP chunking of PoS-tagged Wall Street Journal material. GRAPHON, finally, is a grapheme-to-phoneme conversion task for English based on the English Celex lexical database. Numeric characteristics of the different tasks are listed in table 1. All tasks with the exception of GRAPHON happened to be five-class problems; for GRAPHON, a five-class subset was taken from the original training set, in order to keep computational demands manageable. The tasks were subjected to the

Data set	Features	Classes	Instances
DIMIN	12	5	3,000
STRESS	12	5	3,000
MORPH	9	5	300,000
NPVP	8	5	200,000
GRAPHON	7	5	73,525

Table 1: Data sets.

8 different experimental situations of table 2. For feature selection-based ECOC, backward sequential feature elimination was used (Raaijmakers, 1999), repeatedly eliminating features in turn and evaluating each elimination step with 10-fold cross-validation. For dichotomizer weighting, error information of the dichotomizers, determined from separate unweighted 10-fold cross-validation experiments on a separate training set, produced a weighted Hamming distance metric. Error-based weights were based on raising a small constant β in the interval $[0, 1)$ to the power of the number of errors made by the dichotomizer (Cesa-Bianchi et al., 1996). Random feature selection drawing features with replacement created feature sets of both different size and composition for every dichotomizer.

5 Results

Table 3 lists the generalization accuracies for the control groups, and table 4 for the ECOC algorithms. All accuracy results are based on 10-fold cross-validation, with $p < 0.05$ using paired t-tests. The results show that dis-

ALGORITHM	DESCRIPTION
$\mathcal{E}1$	ECOC, feature selection per bit (15), k=1, unweighted
$\mathcal{E}2$	ECOC, feature selection per bit (15), k=1, weighted
$\mathcal{E}3$	ECOC, feature selection per bit (15), MVDM, k=1, unweighted
$\mathcal{E}4$	ECOC, feature selection per bit (15), MVDM, k=1, weighted
$\mathcal{E}5$	ECOC, feature selection per bit (15), MVDM, k=3, unweighted
$\mathcal{E}6$	ECOC, feature selection per bit (15), MVDM, k=3, weighted
$\mathcal{E}7$	ECOC, voting (100) per bit (30), MVDM, k=3
$\mathcal{E}8$	ECOC, voting (100) per bit block (15), MVDM, k=3

Table 2: Algorithms

GROUP	I	II	III	IV
	IB1-IG k=1	IB1-IG k=3	IB1-IG k=1 MVDM	IB1-IG k=3 MVDM
DIMIN	98.1±0.5	95.8±0.5	97.7±0.7	98.1±0.5
STRESS	83.5±2.6	81.3±2.9	86.2±2.0	86.7±1.8
MORPH	92.5±1.4	92.0±1.4	92.5±1.4	92.5±1.4
NPVP	96.4±0.2	97.1±0.2	97.0±0.1	97.0±0.1
GRAPHON	97.1±2.4	97.2±2.3	97.7±0.7	97.7±0.8

Table 3: Generalization accuracies control groups.

tributed class representations can lead to statistically significant accuracy gains for a variety of linguistic tasks. The ECOC algorithm based on feature selection and weighted Hamming distance performs best. Voting-based ECOC performs poorly on DIMIN and STRESS with voting per bit, but significant accuracy gains are achieved by voting per block, putting it on a par with the best performing algorithm. Regression analysis was applied to investigate the effect of the Modified Value Difference Metric on ECOC accuracy. First, the accuracy gain of MVDM as a function of the information gain ratio of the features was computed. The results show a high correlation (0.82, significant at $p < 0.05$) between these variables, indicating a linear relation. This is in line with the idea underlying MVDM: whenever two feature values are very predictive of a shared class, they contribute to the similarity between the instances they belong to, which will lead to more accurate classifiers. Next, regression analysis was applied to determine the effect of MVDM on ECOC, by relating the accuracy gain of MVDM (k=3) compared to

Task	$\mathcal{E}1$ (I)	$\mathcal{E}2$(I)	$\mathcal{E}3$ (III)	$\mathcal{E}4$ (III)	$\mathcal{E}5$ (IV)	$\mathcal{E}6$ (IV)	$\mathcal{E}7$ (IV)	$\mathcal{E}8$ ($\mathcal{E}6$)
DIMIN	98.6±0.4$\surd$	98.5±0.4$\surd$	98.6±0.6$\surd$	98.7±0.6$\surd$	98.8±0.5$\surd$	98.9±0.4$\surd$	96.6±0.9×	98.4±0.4
STRESS	85.3±1.8$\surd$	86.3±2.0$\surd$	88.2±1.7$\surd$	88.8±1.7$\surd$	88.2±1.7$\surd$	89.3±1.9$\surd$	86.5±2.3×	88.8±1.7
MORPH	93.2±1.6$\surd$	93.2±1.5$\surd$	93.2±1.3$\surd$	93.2±1.3$\surd$	93.2±1.6$\surd$	93.2±1.5$\surd$	93.0±1.6$\surd$	93.4±1.5$\surd$
NPVP†	96.8±0.1$\surd$	96.9±0.2$\surd$	96.8±0.1	96.9±0.1	96.8±0.1	96.9±0.1	96.8±0.2×	96.8±0.2
GRAPHON	98.2±0.7	98.3±0.7	98.4±0.6$\surd$	98.3±0.5$\surd$	98.3±0.6$\surd$	98.5±0.5$\surd$	97.6±0.7×	97.6±0.8×

Table 4: Generalization accuracies for feature selection-based ECOC ($\surd$ indicates significant improvement over control group (in round brackets) , and × deterioration at $p < 0.05$ using paired t-tests). A † indicates 25 voters for performance reasons.

control group II to the accuracy gain of ECOC (algorithm $\mathcal{E}6$, compared to control group IV). The correlation between these two variables is very high (0.93, significant at $p < 0.05$), again indicative of a linear relation. From the perspective of learning class boundaries, the strong effect of MVDM on ECOC accuracy can be understood as follows. When the overlap metric is used, members of a training set belonging to the same class may be situated arbitrarily remote from each other in the feature hyperspace. For instance, consider the following two instances taken from DIMIN:

```
-,-,-,-,-,-,-,-,-,d,A,k,je
-,-,-,-,-,-,-,-,-,d,A,x,je
```

(Hyphens indicate absence of feature values.) These two instances encode the diminutive formation of Dutch `dakje` (*little roof*) from `dak` (*roof*), and `dagje` (lit. *little day*, proverbially used) from `dag` (*day*). Here, the values `k` and `x`, corresponding to the velar stop 'k' and the velar fricative 'g', are minimally different from a phonetic perspective. Yet, these two instances have coordinates on the twelfth dimension of the feature hyperspace that have nothing to do with each other. The overlap treats the `k-x` value clash just like any other value clash. This phenomenon may lead to a situation where inhabitants of the same class are scattered over the feature hyperspace. In contrast, a value difference metric like MVDM which attempts to group feature values on the basis of class cooccurrence information, might group `k` and `x` together if they share enough classes. The effect of MVDM on the density of the feature hyperspace can be compared with the density obtained with the overlap metric as follows. First, plot a random numerical transform of a feature space. For expository reasons, it is adequate to restrict attention to a low-dimensional (e.g. two-dimensional) subset of the feature space, for a specific class C. Then, plot an MVDM transform of this feature space, where every coordinate (a, b) is transformed into $(P(C \mid a), P(C \mid b))$. This idea is applied to a subset of DIMIN, consisting of all instances classified as `je` (one of the five diminutive suffixes for Dutch). The features for this subset were limited to the last two, consisting of the rhyme and coda of the last syllable of the word, clearly the most informative features for this task. Figure 2 displays the two scatter plots. As can be seen, instances are widely scattered over the feature space for the numerical transform, whereas the MVDM-based transform forms many clusters and produces much higher density. In a condensed fea-

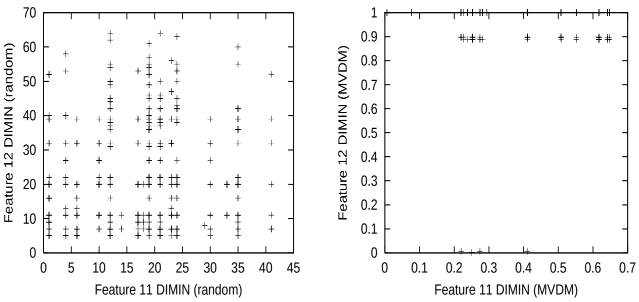

Figure 2: Random numerical transform of feature values based on the overlap metric (left) vs. numerical transform of feature values based on MVDM (right), for a two-features–one-class subset of DIMIN.

ture hyperspace the number of class boundaries to be learned per bit function reduces. For instance, figures 3 displays the class boundaries for a relatively condensed feature hyperspace, where classes form localized populations, and a scattered feature hyperspace, with classes distributed over non-adjacent regions. The number of class boundaries in the scattered feature space is much higher, and this will put an additional burden on the learning problems constituted by the various bit functions.

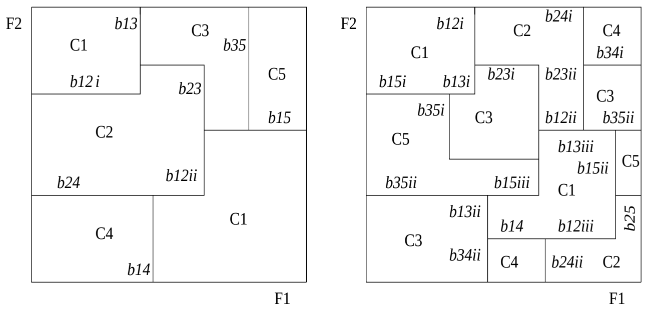

Figure 3: Condensed feature space (left) vs. scattered feature space (right).

6 Conclusions

The use of error-correcting output codes (ECOC) for representing natural language classes has been empirically validated for a suite of linguistic tasks. Results indicate that ECOC can be useful for datasets with features with high class predictivity. These sets typically tend to benefit from the Modified Value Difference Metric, which creates a condensed hyperspace of features. This in turn leads to a lower number of class boundaries to be learned per bit function, which simplifies the binary subclassification tasks. A voting algorithm for learning blocks of bits proves as accurate as an expensive feature-selecting algorithm. Future research will address further mechanisms of learning complex regions of the class boundary landscape, as well as alternative error-correcting approaches to classification.

Acknowledgements

Thanks go to Francesco Ricci for assistance in generating the error-correcting codes used in this paper. David Aha and the members of the Induction of Linguistic Knowledge (ILK) Group of Tilburg University and Antwerp University are thanked for helpful comments and criticism.

References

H. Baayen, R. Piepenbrock, and H. van Rijn. 1993. *The CELEX database on CD-ROM.* Linguistic Data Consortium. Philadelpha, PA.

S. Bay. 1999. Nearest neighbor classification from multiple feature subsets. *Intelligent Data Analysis, 3(3):191–209.*

A. Berger. 1999. Error-correcting output coding for text classification. *Proceedings of IJCAI'99: Workshop on machine learning for information filtering.*

C. Cardie, S. Mardis, and D. Pierce. 1999. Combining error-driven pruning and classification for partial parsing. *Proceedings of the Sixteenth International Conference on Machine Learning, pp. 87-96.*

N. Cesa-Bianchi, Y. Freund, D. Helmbold, and M. Warmuth. 1996. On-line prediction and conversion strategies. *Machine Learning 27:71–110.*

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

W. Daelemans, J. Zavrel, P. Berck, and S. Gillis. 1996. Mbt: A memory-based part of speech tagger generator. *Proceedings of the Fourth Workshop on Very Large Corpora, ACL SIGDAT.*

W. Daelemans, J. Zavrel, K. Van der Sloot, and A. Van den Bosch. 1999. Timbl: Tilburg memory based learner, version 2.0, reference guide. *ILK Technical Report – ILK 99-01. Tilburg.*

T. Dietterich and G. Bakiri. 1995. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research, 2:263–286.*

R. Duda and P. Hart. 1973. *Pattern classification and scene analysis.* Wiley Press.

E. Kong and T. Dietterich. 1995. Error-correcting output coding corrects bias and variance. *Proceedings of the 12th International Conference on Machine Learning.*

J.R. Quinlan. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, Ca.

S. Raaijmakers. 1999. Finding representations for memory-based language learning. *Proceedings of CoNLL-1999.*

F. Ricci and D. Aha. 1997. Extending local learners with error-correcting output codes. *Proceedings of the 14th Conference on Machine Learning.*

D. Roth. 1998. A learning approach to shallow parsing. *Proceedings EMNLP-WVLC'99.*

H. Schmid. 1994. Part-of-speech tagging with neural networks. *Proceedings COLING-94.*

C. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal,27:7, pp. 379-423, 27:10, pp. 623-656.*

A. van den Bosch and W. Daelemans. 1993. Data-oriented methods for grapheme-to-phoneme conversion. *Proceedings of the 6th Conference of the EACL.*