An Experimental Study on Rotation Forest Ensembles

Ludmila I. Kuncheva[1] and Juan J. Rodríguez[2]

[1] School of Electronics and Computer Science, University of Wales, Bangor, UK
l.i.kuncheva@bangor.ac.uk
[2] Departamento de Ingeniería Civil, Universidad de Burgos, 09006 Burgos, Spain
jjrodriguez@ubu.es

Abstract. Rotation Forest is a recently proposed method for building classifier ensembles using independently trained decision trees. It was found to be more accurate than bagging, AdaBoost and Random Forest ensembles across a collection of benchmark data sets. This paper carries out a lesion study on Rotation Forest in order to find out which of the parameters and the randomization heuristics are responsible for the good performance. Contrary to common intuition, the features extracted through PCA gave the best results compared to those extracted through non-parametric discriminant analysis (NDA) or random projections. The only ensemble method whose accuracy was statistically indistinguishable from that of Rotation Forest was LogitBoost although it gave slightly inferior results on 20 out of the 32 benchmark data sets. It appeared that the main factor for the success of Rotation Forest is that the transformation matrix employed to calculate the (linear) extracted features is sparse.

Keywords: Pattern recognition, Classifier ensembles, Rotation Forest, Feature extraction.

1 Introduction

Classifier ensembles usually demonstrate superior accuracy compared to that of single classifiers. Within the classifier ensemble models, *AdaBoost* has been declared to be the best off-the-shelf classifier [4]. A close rival to AdaBoost is *bagging* where the classifiers in the ensemble are built independently of one another, using some randomisation heuristic [3,5]. Bagging has been found to outperform AdaBoost on noisy data [1] but is generally perceived as the less accurate of the two methods. To encourage diversity in bagging, further randomisation was introduced in the Random Forest model [5]. A Random Forest ensemble consists of decision trees trained on bootstrap samples from the data set. Additional diversity is introduced by randomising the feature choice at each node. During tree construction, the best feature at each node is selected among M randomly chosen features, where M is a parameter of the algorithm.

Rotation Forest [15] draws upon the Random Forest idea. The base classifiers are also independently built decision trees, but in Rotation Forest each tree

M. Haindl, J. Kittler, and F. Roli (Eds.): MCS 2007, LNCS 4472, pp. 459–468, 2007.

is trained on the whole data set in a rotated feature space. As the tree learning algorithm builds the classification regions using hyperplanes parallel to the feature axes, a small rotation of the axes may lead to a very different tree. A comparative experiment by Rodriguez et al. [15] favoured Rotation Forest to bagging, AdaBoost and Random Forest. Studying kappa-error diagrams it was discovered that Rotation Forest would often produce more accurate classifiers than AdaBoost which are also more diverse than those in bagging.

This paper explores the effect of the design choices and parameter values on the performance of Rotation Forest ensembles. The rest of the paper is organized as follows. Section 2 explains the Rotation Forest ensemble method and the design choices within. Sections 3 to 6 comprise our lesion study. Experimental results are reported also in Section 6. Section 7 offers our conclusions.

2 Rotation Forest

Rotation Forest is an ensemble method which trains L decision trees independently, using a different set of extracted features for each tree [15]. Let $\mathbf{x} = [x_1, \ldots, x_n]^T$ be an example described by n features (attributes) and let X be an $N \times n$ matrix containing the training examples. We assume that the true class labels of all training examples are also provided. Let $\mathcal{D} = \{D_1, \ldots, D_L\}$ be the ensemble of L classifiers and F be the feature set.

Rotation Forest aims at building accurate *and* diverse classifiers. Bootstrap samples are taken as the training set for the individual classifiers, as in bagging. The main heuristic is to apply feature extraction and to subsequently reconstruct a full feature set for each classifier in the ensemble. To do this, the feature set is split randomly into K subsets, principal component analysis (PCA) is run separately on each subset, and a new set of n linear extracted features is constructed by pooling all principal components. The data is transformed linearly into the new feature space. Classifier D_i is trained with this data set. Different splits of the feature set will lead to different extracted features, thereby contributing to the diversity introduced by the bootstrap sampling.

We chose decision trees as the base classifiers because they are sensitive to rotation of the feature axes and still can be very accurate. The effect of rotating the axes is that classification regions of high accuracy can be constructed with fewer trees than in bagging and AdaBoost. Our previous study [15] reported an experiment whose results were favourable to Rotation Forest compared to bagging, AdaBoost and Random Forest with the same number of base classifiers. The design choices and the parameter values of the Rotation Forest were picked in advance and not changed during the experiment. These were as follows

- Number of features in a subset: $M = 3$;
- Number of classifiers in the ensemble: $L = 10$;
- Extraction method: principal component analysis (PCA);
- Base classifier model: decision tree (hence the name "forest").

Thirty two data sets from UCI Machine Learning Repository [2], summarized in Table 1, were used in the experiment. The calculations and statistical

Table 1. Characteristics of the 32 data sets used in this study

data set	c	N	n_d	n_c	data set	c	N	n_d	n_c
anneal	6	898	32	6	labor	2	57	8	8
audiology	24	226	69	0	lymphography	4	148	15	3
autos	7	205	10	16	pendigits	10	10992	0	16
balance-scale	3	625	0	4	pima-diabetes	2	768	0	8
breast-cancer	2	286	10	0	primary-tumor	22	239	17	0
cleveland-14-heart	5	307	7	6	segment	7	2310	0	19
credit-rating	2	690	9	6	sonar	2	208	0	60
german-credit	2	1000	13	7	soybean	19	683	35	0
glass	7	214	0	9	splice	3	3190	60	0
heart-statlog	2	270	0	13	vehicle	4	846	0	18
hepatitis	2	155	13	6	vote	2	435	16	0
horse-colic	2	368	16	7	vowel-context	11	990	2	10
hungarian-14-heart	5	294	7	6	vowel-nocontext	11	990	0	10
hypothyroid	4	3772	22	7	waveform	3	5000	0	40
ionosphere	2	351	0	34	wisc-breast-cancer	2	699	0	9
iris	3	150	0	4	zoo	7	101	16	2

Notes: c is the number of classes, N is the number of objects, n_d is the number of discrete (categorical) features and n_c is the number of contiunous-valued features

comparisons were done using Weka [20]. Fifteen 10-fold cross-validations were used with each data set.

Statistical comparisons in Weka are done using a corrected estimate of the variance of the classification error [14].

The remaining sections of this paper address the following questions.

1. Is splitting the features set F essential for the success of Rotation Forest?
2. How does K (respectively M) affect the performance of Rotation Forest? Is there a preferable value of K or M?
3. How does Rotation Forest compare to bagging and AdaBoost for various ensemble sizes L?
4. Is PCA the best method to rotate the axes for the feature subsets? Since we are solving a classification problem, methods for linear feature extraction which use discriminatory information may be more appropriate.

3 Is Splitting Essential?

Splitting the feature set, F, into K subsets is directed towards creating diversity. Its effect is that each new extracted feature is a linear combination of $M = \lfloor n/K \rfloor$ original features. Then the "rotation matrix", R, used to transform a bootstrap sample T of the original training set into a new training set ($T' = TR$) is sparse. [1]

[1] Here we refer to random projections broadly as "rotations", which is not technically correct. For the random projections to be rotations, the rotation matrix must be orthonormal. In our case we only require this matrix to be non-degenerate. The choice of terminology was guided by the fact that random projections are a version of the rotation forest idea, the only difference being that PCA is replaced by a non-degenerate random linear transformation.

To find out how a sparse rotation matrix compares to a full rotation matrix we used two approaches.

We call the first approach Random Projections. L random (nondegenerate) transformation matrices of size $n \times n$ were generated, denoted $R_1, \ldots, R_L$, with entries sampled from a standard normal distribution $\sim N(0, 1)$. There is no loss of information in this transformation as R_i can be inverted and the original space restored. However, any such transformation may distort or enhance discriminatory information. In other words, a rotation may simplify or complicate the problem, leading to very different sizes of the decision trees in the two cases.

In the second approach, which we call Sparse Random Projections, sparse transformation matrices were created so as to simulate the one in the Rotation Forest method. The non-zero elements of these matrices were again sampled randomly from a standard normal distribution. L such matrices were generated to form the ensemble.

For ensembles with both pruned and unpruned trees, Sparse Random Projections were better than Random Projections on 24 of the 32 data sets, where 7 of these differences were statistically significant. Of the remaining 8 cases where Random Projections were better than Sparse Random Projections, none of the differences were statistically significant.

Next we look at the reasons why Sparse Random Projections are better than Random Projections.

3.1 Accuracy of the Base Classifiers

One of the factors contributing to the accuracy of an ensemble is the accuracy of its base classifiers. Hence, one of the possible causes for the difference between the performances of Random Projections and Sparse Random Projections may be due to a difference in the accuracies of the base classifiers. We compared the results obtained using a single decision tree with the two projection methods. As in the main experiment, we ran 15 10-fold cross-validations on the 32 data sets. The results displayed in Table 2 show that

- Decision trees obtained with the projected data are worse than decision trees obtained with the original data.
- Decision trees obtained after a non-sparse projection are worse than decision trees obtained after a sparse projection.

One possible explanation of the observed relations is that projecting the data randomly is similar to introducing noise. The degree of non-sparseness of the projection matrix gauges the amount of noise. For instance, a diagonal projection matrix will only rescale the axes and the resultant decision tree will be equivalent to a decision tree built on the original data.

3.2 Diversity of the Ensemble

Better accuracy of the base classifiers alone is not sufficient to guarantee better ensemble accuracy. For the ensemble to be successful, accuracy has to be coupled

Table 2. Comparison of the results obtained with a single decision tree, using the original and the projected data. The entry $a_{i,j}$ shows the number of datasets for which the method of column j gave better results than the method of row i. The number in the parentheses shows in how many of these cases the difference has been statistically significant.

	With pruning			Without pruning		
	original	non-sparse	sparse	original	non-sparse	sparse
original	-	3 (0)	8 (0)	-	4 (0)	9 (0)
non-sparse	29 (17)	-	30 (13)	28 (15)	-	28 (11)
sparse	24 (8)	2 (0)	-	23 (7)	4 (0)	-

with diversity. To study further the effect due to splitting the feature set, we consider kappa-error diagrams with sparse and with non-sparse projections.

Kappa-error diagrams is the name for a scatterplot with $L(L-1)/2$ points, where L is the ensemble size. Each point corresponds to a pair of classifiers. On the x-axis is a measure of diversity between the pair, κ. On the y-axis is the averaged individual error of the classifiers in the pair, $E_{i,j} = \frac{E_i+E_j}{2}$. As small values of κ indicate better diversity and small values of $E_{i,j}$ indicate better accuracy; the most desirable pairs of classifiers will lie in the bottom left corner.

Figure 1 plots the kappa-error diagrams for two data sets, audiology and vowel-context, using Random Projections and Sparse Random Projections with pruned trees. The points come from a 10-fold cross-validation for ensembles of size 10. We chose these two data sets because they are typical examples of the two outcomes of the statistical comparisons of sparse and non-sparse random projections. For the audiology data set, sparse projections are substantially better while for vowel-context data the two methods are indistinguishable.

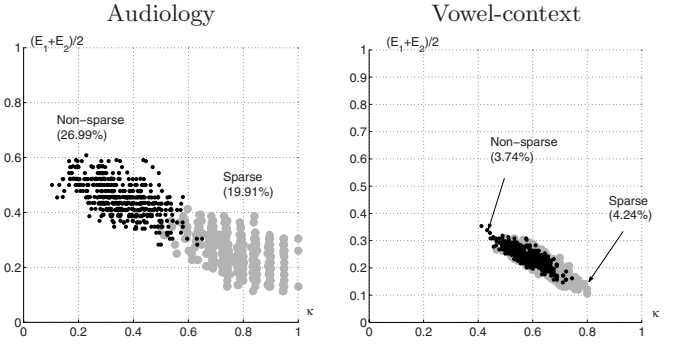

Fig. 1. Kappa-error diagrams for two data sets for sparse and non-sparse random projections. The ensemble errors are marked on the plots.

The figure shows that the success of the sparse projections compared to non-sparse projections is mainly due to maintaining high accuracy of the base classifiers. The additional diversity obtained through the "full" random projection is not useful for the ensemble.

The results presented in this section show that splitting the feature set is indeed essential for the success of Rotation Forest.

4 Number of Feature Subsets, K

No consistent relationship between the ensemble error and K was found. The patterns for different data sets vary from clear steady decrease of the error with K (audiology, autos, horse-colic, hypothyroid, segment, sonar, soybean, splice, vehicle and waveform datasets), through non-monotonic (almost horizontal) lines, to marked increase of the error with K (balance-scale, glass, vowel-nocontext and wisconsin-breast-cancer datasets). The only regularity was that for $K = 1$ and $K = n$ the errors were larger than these with values in-between. This is not unexpected. First, for $K = 1$, we have non-sparse projections, which were found in the previous section to give inferior accuracy to that when sparse projections were used. Splitting the feature set into $K = 2$ subsets immediately makes the rotation matrix 50% sparse, which is a prerequisite for accurate individual classifiers and ensembles thereof. On the other hand, if $K = n$, where n is the number of features, then any random projection reduces to rescaling of the features axes, and the decision trees are identical. Thus for $K = n$ we have a single tree rather than an ensemble.

As with K, there was no consistent pattern or regularity for M ranged between 1 and 10. In fact, the choice $M = 3$ in [15] has not been the best choice in terms of smallest cross-validation error. It has been the best choice for only 4 data sets out of the 32, and the worst choice in 8 data sets! Thus $M = 3$ has not been a serendipitous guess in our previous study. As Rotation Forest outperformed bagging, AdaBoost and Random Forest for this rather unfavourable value of M, we conclude that Rotation Forest is robust with respect to the choice of M (or K).

5 Ensemble Size, L

All our previous experiments were carried out with $L = 10$ ensemble members. Here AdaBoost.M1, Random Forest and Rotation Forest were run with L varying from 1 (single tree) to 100. Only unpruned trees were used because Random Forest only operates with those. Since the classification accuracies vary significantly from dataset to dataset, ranking methods are deemed to provide a more fair comparison [7]. In order to determine whether the ensemble methods were significantly different we ran Friedman's two-way ANOVA. The ensemble accuracies for a fixed ensemble size, L, were ranked for each dataset. Figure 2 plots the average ranks of the four methods versus ensemble size, L. All results shown are from a single 10-fold cross-validation. If the ensemble methods are equivalent, then their ranks would be close to random for the different data sets. According to Friedman's test, the ensemble methods are different at significance level 0.005 for all values of L. However, the differences are largely due to the rank of bagging being consistently the worst of the four methods. A further pairwise comparison was carried out. With significance level of 0.1, Rotation Forest was found to be significantly better than bagging for $L \in \{3, 6, 10 - 100\}$, better than AdaBoost

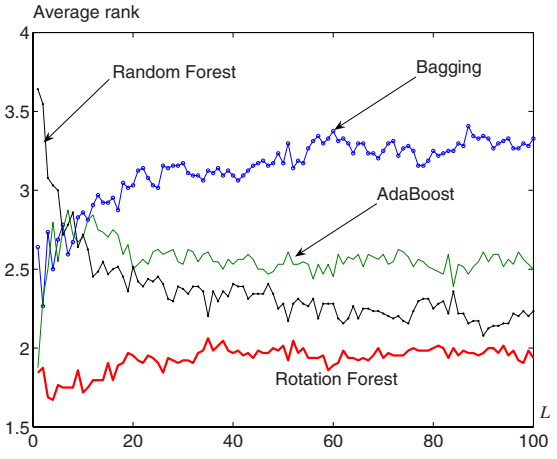

Fig. 2. The average ranks ($\bar{R}$) of the four ensemble methods as a function of the ensemble size (L)

for $L \in \{4, 7, 11, 12\}$, and better than Random Forest for $L \in \{1-5, 7, 8\}$. The differences are more prominent for small ensemble sizes. However, we note the consistency of Rotation Forest being the best ranking method across all values of L, as shown in Figure 2.

6 Suitability of PCA

6.1 Alternatives to PCA

Principal Component Analysis (PCA) has been extensively used in statistical pattern recognition for dimensionality reduction, sometimes called Karhunen-Loéve transformation. PCA has also been used to extract features for classifier ensembles [16, 17]. Unlike the approaches in the cited works, we do not employ PCA for dimensionality reduction but for rotation of the axes while keeping all the dimensions.

It is well documented in the literature since the 1970s that PCA is not particularly suitable for feature extraction in classification because it does not include discriminatory information in calculating the optimal rotation of the axes. Many alternative linear transformations have been suggested based on discrimination criteria [11,9,13,19,18]. Sometimes a simple random choice of the transformation matrix has led to classification accuracy superior to that with PCA [8].

To examine the impact of PCA on Rotation Forest, we substitute PCA by Sparse Random Projections and Nonparametric Discriminant Analysis (NDA) [12,10,6]. The Sparse Random Projections were all non-degenerate but were not orthogonal in contrast to PCA.

We compared Rotation Forest (with PCA, NDA, Random Projections and Sparse Random Projections), denoted respectively RF(PCA), RF(NDA), RF(R)

and RF(SR), with bagging, AdaBoost, and Random Forest. We also decided to include in the comparison two more ensemble methods, LogitBoost and Decorate, which have proved to be robust and accurate, and are available from Weka [20].

6.2 Win-Draw-Loss Analysis

Tables 3 and 4 give summaries of the results.

Table 3. Comparison of ensemble methods

	(2)	(3)	(4)	5)	(6)	(7)	(8)	(9)	(10)*
(1) Decision Tree	11/21/0	12/19/1	18/14/0	11/19/2	8/22/2	9/20/3	19/13/0	17/15/0	20/12/0
(2) Bagging		4/27/1	10/21/1	2/27/3	3/27/2	4/24/4	6/26/0	7/25/0	9/23/0
(3) AdaBoost			7/24/1	1/26/5	1/28/3	3/23/6	5/26/1	5/26/1	8/23/1
(4) LogitBoost				0/24/8	0/24/8	1/21/9	0/29/2	2/26/3	2/28/1
(5) Decorate					5/27/0	2/25/5	6/26/0	6/26/0	7/25/0
(6) Random Forest						2/26/4	6/26/0	8/24/0	9/23/0
(7) RF(R)							7/25/0	8/24/0	9/23/0
(8) RF(SR)								1/31/0	1/31/0
(9) RF(NDA)									0/32/0

*(10) RF(PCA)

Table 4. Ranking of the methods using the significant differences from all pairwise comparisons

Number	Method	Dominance	Win	Loss	Average rank
(10)	RF(PCA)	63	65	2	2.750
(4)	LogitBoost	59	66	7	3.656
(9)	RF(NDA)	50	54	4	3.938
(8)	RF(SR)	44	49	5	4.219
(3)	AdaBoost.M1	2	34	32	6.219
(6)	Random Forest	-19	21	40	6.281
(1)	Bagging	-23	22	45	6.500
(4)	Decorate	-25	19	44	6.844
(7)	RF(R)	-34	21	55	5.719
(1)	Decision Tree	-117	8	125	8.875

The three values in each entry of Table 3 refer to how many times the method of the column has been significantly-better/same/significantly-worse than the method of the row. The corrected estimate of the variance has been used in the test, with level of significance $\alpha = 0.05$.

Table 4 displays the overall results in terms of ranking. Each of the methods being compared receives a ranking in comparison with the other methods. The Dominance Rank of method 'X' is calculated as Wins–Losses, where Wins if the total number of times method 'X' has been significantly better than another method and Losses is the total number of times method 'X' has been significantly worse than another method.

To our surprise, PCA-based Rotation Forest scored better than both NDA and Random Projections (both sparse and non-sparse). It is interesting to note

that there is a large gap between the group containing Rotation Forest models and LogitBoost on the one hand and the group of all other ensemble methods on the other hand. The only representative of Rotation Forest in the bottom group is Random Projections, which is in fact a feature extraction ensemble. It does not share one of the most important characteristic of the rotation ensembles: sparseness of the projection. We note though that if the problem has $c > 2$ classes, LogitBoost builds c trees at each iteration, hence Lc trees altogether, while in Rotation Forest the number of trees is L.

The last column of the table shows the average ranks. The only anomaly in the table is the rank of RF(P) which places the method further up in the table, before Adaboost.M1. The reason for the discrepancy between Dominance and Average rank is that RF(P) has been consistently better than Adaboost.M1 and the methods below it but the differences in favour of RF(P) have not been statistically significant. On the other hand, there have been a larger number of statistically significant differences for the problems where Adaboost.M1, Random Forest, bagging and Decorate have been better than RF(R). PCA seems to be slightly but consistently better than the other feature extraction alternatives. Sparse random projections are the next best alternative being almost as good as NDA projections but substantially cheaper to run.

7 Conclusions

Here we summarize the answers to the four questions of this study

1. Is splitting the features set F into subsets essential for the success of Rotation Forest? Yes. We demonstrated this by comparing sparse with non-sparse random projections; the results were favourable to sparse random projections.
2. How does K (respectively M) affect the performance of Random Forest? No pattern of dependency was found between K (M) and the ensemble accuracy which prevent us from recommending a specific value. As $M = 3$ worked well in our experiments, we propose to use the same value in the future.
3. How does Rotation Forest compare to bagging, AdaBoost and Random Forest for various ensemble sizes? Rotation Forest was found to be better than the other ensembles, more so for smaller ensembles sizes (Figure 2).
4. Is PCA the best method to rotate the axes for the feature subsets? PCA was found to be the best method so far.

In conclusion, the results reported here support the heuristic choices made during the design of the Rotation Forest method. It appears that the key to its robust performance lies in the core idea of the method – to make the individual classifiers as diverse as possible (by rotating the feature space) and at the same time not compromising on the individual accuracy (by choosing sparse rotation, keeping all the principal components in the rotated space and using the whole training set for building each base classifier).

References

1. E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–142, 1999.
2. C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.
3. L. Breiman. Bagging predictors. *Machine Learning*, 26(2):123–140, 1996.
4. L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.
5. L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.
6. M. Bressan and J. Vitrià. Nonparametric discriminant analysis and nearest neighbor classification. *Pattern Recognition Letters*, 24:2743–2749, 2003.
7. J Demšar. Statistical comparison of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
8. X. Z. Fern and C. E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proc. 20th International Conference on Machine Learning, ICML*, pages 186–193, Washington,DC, 2003.
9. F.H. Foley and J.W. Sammon. An optimal set of discriminant vectors. *IEEE Transactions on Computers*, 24(3):281–289, 1975.
10. K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, Boston, MA, 2nd edition, 1990.
11. K. Fukunaga and W.L.G. Koontz. Application of the karhunen-loeve expansion to feature selection and ordering. *IEEE Transactions on Computers*, 19(4):311–318, April 1970.
12. K. Fukunaga and J. Mantock. Nonparametric discriminant analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(6):671–678, 1983.
13. J.V. Kittler and P.C. Young. A new approach to feature selection based on the karhunen-loeve expansion. *Pattern Recognition*, 5(4):335–352, December 1973.
14. C. Nadeau and Y. Bengio. Inference for the generalization error. *Machine Learning*, 62:239–281, 2003.
15. J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, Oct 2006.
16. M. Skurichina and R. P. W. Duin. Combining feature subsets in feature selection. In *Proc. 6th International Workshop on Multiple Classifier Systems, MCS'05*, volume LNCS 3541, pages 165–175, USA, 2005.
17. K. Tumer and N. C. Oza. Input decimated ensembles. *Pattern Analysis and Applications*, 6:65–77, 2003.
18. F. van der Heijden, R. P. W. Duin, D. de Ridder, and D. M. J. Tax. *Classification, Parameter Estimation and State Estimation*. Wiley, England, 2004.
19. A. Webb. *Statistical Pattern Recognition*. Arnold, London, England, 1999.
20. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2nd edition, 2005.