Deformation-Aware Log-Linear Models

Tobias Gass, Thomas Deselaers[1] and Hermann Ney

Human Language Technology and Pattern Recognition Group,
RWTH Aachen University, Aachen, Germany
<last name>@i6.informatik.rwth-aachen.de
[1] now with the Computer Vision Laboratory, ETH Zurich, Switzerland

Abstract. In this paper, we present a novel deformation-aware discriminative model for handwritten digit recognition. Unlike previous approaches our model directly considers image deformations and allows discriminative training of all parameters, including those accounting for non-linear transformations of the image. This is achieved by extending a log-linear framework to incorporate a latent deformation variable. The resulting model has an order of magnitude less parameters than competing approaches to handling image deformations. We tune and evaluate our approach on the USPS task and show its generalization capabilities by applying the tuned model to the MNIST task. We gain interesting insights and achieve highly competitive results on both tasks.

1 Introduction

One of the major problems in pattern recognition tasks is to model intra-class variability without washing away the inter-class differences. One typical application where many transformations have to be considered is the recognition of handwritten characters. In the past, many approaches towards modeling different writing styles have been proposed and investigated [1–5].

In this work, we propose a novel model that directly incorporates and trains deformation parameters in a log-linear classification framework.

The conventional approaches can be split into two groups: Approaches that directly incorporate certain invariances into their classification framework, e.g. incorporate the tangent distance into support vector machines [2], use kernel-jittering to obtain translated support vectors in a two-step training approach [1] and use transformation-invariant distance measures in nearest neighbor frameworks [4, 5]. Another approach is not to incorporate the deformation-invariance into the model but use a huge amount of synthetically deformed data during training of a convolutional neural network [3]. The first approach has the disadvantage that during testing a large amount of, potentially computationally expensive, image comparisons have to be performed whereas in the second approach the training procedure becomes potentially very expensive. None of the approaches presented above explicitly learns the parameters of the allowed deformations but the deformation model was hand-coded by the system developers.

In contrast to these approaches to transformation-invariant classification, Memisevic and Hinton [6] proposed an approach to learn image transformations from corresponding image pairs using conditional restricted Boltzmann machines.

In our approach, we aim at training a small (in the number of parameters) model that directly models deformations, automatically learns which deformations are allowed (and desired), is efficient to train and apply, and leads to good results. We build our approach around the image distortion model (IDM) [4], a zero-order, non-linear deformation model, which we shortly describe in the following section. The developed model can also be considered a grid-shaped hidden-conditional random field (HCRF) [7, 8] where the latent variables account for the deformations.

In section 3, we present our model which incorporates the IDM into log-linear models. In section 4, we present an experimental evaluation on the USPS and on the MNIST dataset and compare to several published state-of-the-art results as well as to an SVM with an IDM-distance kernel.

2 Image Distortion Model

The IDM has been proposed by several works independently under different names. For example, it has been described as "local pertubations" [9] and as "shift similarity" [10]. Here, we follow the formulation of [4]. The IDM is a zero order image deformation method that accounts for image transformations by pixel-wise aligning a test image to a prototype image without considering the alignments of its neighboring pixels, which allows for efficient calculation. An image alignment maps each pixel ij from an image A of size $I \times J$ to a pixel $(xy)_{ij}$ in the prototype image B. We denote an image alignment by $(xy)_{11}^{IJ} : (ij) \rightarrow (xy)_{ij}$. To restrict the possible alignments, commonly a maximal warp-range W, i.e. the maximal displacement between ij and $(xy)_{ij}$, is defined.

In [4], the IDM has been mainly used to obtain distances between pairs of images for nearest neighbor classification. It was noted that the use of local features extracted from small neighborhoods of the pixels, such as sub-windows of Sobel features, which are smoothed directed derivatives, lead to strongly improved alignments and directly to better classification results.

3 Integrating the IDM into Log-linear Models

Log-linear models are well-understood discriminative classifiers for which efficient training methods exist. Commonly, the class-posterior $p(c|X)$ for an observation X is directly modeled as

$$p(c|X) = \frac{p(c, X)}{\sum_{c'=1}^{C} p(c', X)} = \frac{\exp(g_\theta(X, c))}{\sum_{c'=1}^{C} \exp(g_\theta(X, c'))}, \tag{1}$$

where commonly $g_\theta(X, c) = \alpha_c + \lambda_c^T X$ is chosen.

To incorporate deformation-invariance into log-linear models, we treat the image alignment as a latent variable which is marginalized out:

$$p(c, X) = \sum_{(xy)_{11}^{IJ}} p(c, (xy)_{11}^{IJ}, X). \tag{2}$$

To account for image deformations in the discriminant function we extend $g_\theta(X, c)$ to

$$g_\theta(X, (xy)_{11}^{IJ}, c) = \alpha_c + \sum_{ij} \left(\alpha_{cij(xy)_{ij}} + \lambda_{c(xy)_{ij}}^T X_{ij} \right), \tag{3}$$

where $\theta = \{\alpha_c, \alpha_{cij(xy)_{ij}}, \lambda_{c(xy)_{ij}}\}$ and α_c is a class-bias, $\alpha_{cij(xy)_{ij}}$ corresponding to class-, position, and alignment depending deformation priors; $\lambda_{c(xy)_{ij}}$ is a class-dependent weight-vector. Note that each pixel ij of image X is represented by a D-dimensional vector to allow for additional features. Thus, the $\lambda_{c(xy)_{ij}}$ are of the same dimensionality.

3.1 Relationship to Gaussian Models

An interesting aspect of this model is that it can be rewritten as a discriminative Gaussian classifier analogously to [11]. We rewrite

$$p(c, (xy)_{11}^{IJ}|X) = \frac{p(c)\ p(X, (xy)_{11}^{IJ}|c)}{\sum_{c'} p(c')\ p(X, (xy)_{11}^{IJ}|c')} \tag{4}$$

and decompose

$$p(X, (xy)_{11}^{IJ}|c) = p((xy)_{11}^{IJ}|c)\ p(X|c, (xy)_{11}^{IJ}) \tag{5}$$

where $p((xy)_{11}^{IJ}|c)$ can be considered as a deformation prior and $p(X|c, (xy)_{11}^{IJ})$ is an emission probability for a given class and alignment.

Then, $p((xy)_{11}^{IJ}|c)$ can be rewritten as $p((xy)_{11}^{IJ}|c) = \prod_{ij} p((xy)_{ij}|ij, c)$ and

$$p(X|c, (xy)_{11}^{IJ}) = \frac{1}{\prod_{ij} \sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \sum\nolimits_{ij} \frac{(X_{ij} - \mu_{c(xy)_{ij}})^2}{\sigma^2} \right) \tag{6}$$

assuming a globally pooled diagonal covariance matrix.

The direct correspondence to the above model can be seen by setting

$$\alpha_c = \log\left(p(c)\right) - \tfrac{D}{2} \log\left(2\pi\sigma^2\right), \qquad \lambda_{c(xy)_{ij}} = \tfrac{1}{\sigma} \mu_{c(xy)_{ij}} \tag{7}$$

$$\alpha_{cij(xy)_{ij}} = \log\left(p((xy)_{ij}|ij, c)\right) - \tfrac{1}{2\sigma} \mu_{c(xy)_{ij}}^T \mu_{c(xy)_{ij}}. \tag{8}$$

This equivalence also shows that the $\alpha_{cij(xy)_{ij}}$ model deformation penalties. Furthermore, the transformations in eq. (7)(8) allow to start from a generative, deformation-aware model such as the one discussed in [4] to initialize our model.

3.2 Maximum Approximation

In order to avoid the evaluation of sums over latent variables, a common approach is to use the maximizing configuration of the latent variable, which allows us to rewrite eq. (2) as

$$p(c|X) \approx \frac{1}{Z(X)} \max_{(xy)_{11}^{IJ}} \left\{ p(c, (xy)_{11}^{IJ}, X) \right\} \tag{9}$$

with unchanged $Z(X)$.

In addition to applying the maximum approximation in the numerator, it is possible to also apply it in the denominator $Z(X) \approx \sum_{c'} \max_{(xy)_{11}^{IJ}} p(c', (xy)_{11}^{IJ}, X)$.

We performed experiments with the three different variants and found that the results differ only slightly. In particular, we found that the method with maximum approximation in numerator and denominator despite being the fastest has the tendency to perform best. Therefore, we perform the experiments in this paper using this method.

3.3 Training Method

The training of conventional log-linear models is a convex optimization problem and can be done efficiently. Here, we aim at maximizing the log-likelihood of the posteriors

$$F(\theta) = \sum_n \log p_\theta(c_n|X_n), \tag{10}$$

where θ are the parameters of the class posterior distribution (cp. Eq (3)). For our proposed model, the training problem is not convex anymore. However, given a fixed alignment, the training can be performed normally, and therefore, for the model with maximum approximation an algorithm that is guaranteed to converge (to a local optimum) exists. This can be seen by considering a class/alignment pair as a pseudo-class leading to a log-linear model with an enormous number of classes. For the two other variants (no maximum approximation/maximum approximation in numerator and denominator) this cannot be guaranteed, however, as we found in our experiments, the training converges well.

An extension of the GIS algorithm to allow for training of log-linear models with hidden variables has been presented in [12]. However the authors observed that although the algorithm is guaranteed to converge, convergence can be slow. Similarly to their approach, we also use an alternating optimization method:

Step 1: Train model parameters θ while keeping the alignment $(xy)_{11}^{IJ}$ fixed.
Step 2: Determine new alignments $(xy)_{11}^{IJ}$ with fixed model parameters θ.

These two steps are then repeated until convergence is reached. To train the parameters of the model with maximum approximation in numerator and denominator, the same procedure can be used, but here for each training observation, an alignment for each class has to be determined.

We train our model using the RProp-algorithm [13], which has the advantage that it is robust w.r.t. varying scales of the derivatives because it only takes into account the sign of the partial derivatives to determine the parameter updates.

3.4 Pooling of Deformation Priors

In our initial formulation, the $\alpha_{cij(xy)_{ij}}$ model the deformation priors separately for each class, pixel position and corresponding alignments leading to a large number of parameters partly sharing information. To reduce the number of parameters and allow for sharing deformation information we propose several pooling strategies over classes, positions, and deformations, respectively. An overview over these is given in table 1.

Table 1: **Deformation Priors.** The different variants of sharing α-parameters. We show the dependency of the deformation parameters $\alpha_{cij(xy)_{ij}}$ in functional form, where each of these functions is a table which is indexed by the parameters given in column 2. In the last column, we give the number of parameters in dependency of the number of classes C, the size of the image IJ, and the size of the allowed warp-range W.

Pooling method	$\alpha_{cij(xy)_{ij}}$	number of parameters
full alphas (no pooling)	$\alpha(c, i, j, i-x, j-y)$	$C(IJ)(2W+1)^2$
class pooling	$\alpha(i, j, i-x, y-j)$	$(IJ)(2W+1)^2$
deformation independent	$\alpha(c, i, j, \delta(x, i) \cdot \delta(y, j))$	$2C(IJ)$
position independent	$\alpha(c, i-x, j-y)$	$C(2W+1)^2$
position and deformation independent	$\alpha(c, \delta(x, i) \cdot \delta(y, j))$	$2C$

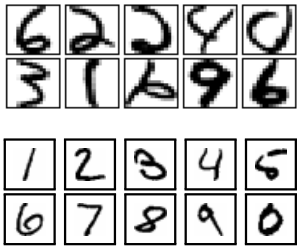

Fig. 1: Example images for the USPS (top) and the MNIST (bottom) tasks.

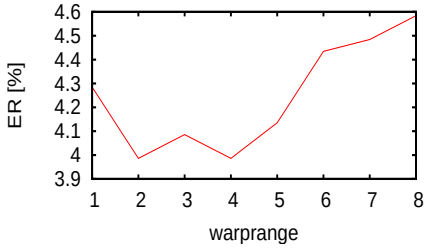

Fig. 2: The effect of different warp ranges on the error rate on the USPS test data.

4 Experimental Evaluation

We evaluate our methods on two datasets, the rather small, but well-known USPS dataset [14], which we use to evaluate and tune all parameters of our method, and on the MNIST dataset [15], on which we only repeat those experiments which performed best on the USPS dataset. In figure 1, we give an example image for each of the classes for these two datasets.

Both datasets consist of images from ten classes of handwritten digits, which are scaled between 0 and 1. The **USPS** dataset consists of 7291 training images and 2007 test images, and the **MNIST** dataset consists of 60 000 training images and 10 000 test images.

In the following, we first investigate the warp range and which features are best for finding the best alignments and for classification. Then we investigate the effect of the different deformation-sharing parameters. We also compare three different initializations and compare our results to the state-of-the-art and to an SVM with an IDM-distance kernel.

Warp range. One crucial parameter in the IDM is the warp range W, which controls the maximal horizontal and vertical displacement for each pixel. We

Table 2: **Features.** The impact of different local features and local context on the classification error [%]

local context used:	no		yes	
Features	train	test	train	test
gray values	2.63	7.62	0.47	7.57
Sobel	0.01	4.04	0.01	4.04
abs(Sobel)	0.78	4.88	0.18	4.78
Sobel + abs(Sobel)	0.01	3.84	0.14	3.64

Table 3: Error rates[%] obtained using the different initializations with and without alternating optimization.

		fixed align.		altern. opt.	
Init.	initial	train	test	train	test
Gaussian	6.52	0.69	4.63	0.69	4.63
log-linear	8.27	0.05	5.93	0.01	4.04
zero init	-	1.87	8.27	0.01	4.04

allow to map the pixel ij to every pixel $(xy)_{ij}$, where $i - W \leq x_i \leq i + W$ and $j - W \leq y_j \leq j + W$. In figure 2 the effect of different warp ranges on the error rate on the USPS dataset is shown. In these experiments we use simple Sobel features.

Features. It was already observed by Keysers et al. [4] that local context is essential to determine good alignments and they found that sub-windows of Sobel features performed best. Here, we investigate the impact of different local descriptors and sub-windows on the classification performance. The results of these experiments are shown in table 2.

We compare eight different setups: simple gray values, Sobel features, absolute values of Sobel features, and a combination of Sobel and absolute Sobel. Each feature setup is evaluated with and without 3×3 sub-windows. It can be observed that using Sobel features, scaled from -1 to 1, leads to a significant improvement over using just gray values and there is hardly a difference in the test error rate whether local context is used or not. Absolute Sobel values do not reach the performance of full Sobel features as they lose the direction of the edge information, although it can be observed that the model improves when combining the two. This is due to the fact that the feature combination contains both improved features for alignment as well as non-linear combinations of the original features which improve parameter estimation of the log-linear model. It can be observed that the use of sub-windows leads to a better performance when using the combined Sobel descriptors. Due to the minor improvements using the feature combination but nonetheless greatly increased training effort, we will use simple Sobel features for further investigations and re-combine the best approaches in section 4.1 for the MNIST dataset.

Alpha pooling. Table 4 shows the results obtained using the different strategies for deformation parameter sharing described in section 3.4.

It can be observed that, although the number of parameters is significantly reduced, the error rates on the test data are only slightly affected. This shows that it is not necessary to have position- and deformation-specific deformation priors but that most of the relevant deformation information can be stored in

Table 4: **Deformation prior sharing.** Error rates[%] on the training and test data of the USPS dataset using the different deformation prior sharing methods along with the number of deformation parameters and the number of parameters of the entire model.

Pooling method	train ER	test ER	def. param	total param
full alphas (no pooling)	0.01	4.04	64000	69130
class pooling	0.08	3.84	6400	11530
deformation independent	0.05	3.94	5120	10250
position independent	0.03	4.09	250	5380
position and deformation independent	0.51	3.89	20	5150
class pooling/pos. & deform. indep.	0.04	3.94	2	5132

the λ-parameters. The biggest difference is again observed on the training data, which makes us believe that the models with fewer parameters have better generalization capabilities.

Initialization and alternating optimization. As described in section 3.1, the presented model can be rewritten as a Gaussian model and can be initialized from a Gaussian model. Since we cannot guarantee convergence to the *global* optimum of the parameters, in this section, we consider three different ways to initialize the model: initialization from a non-deformation invariant log-linear model, initialization from a deformation-aware generative Gaussian model and initialization of all parameters with zeros.

For these alternatives, we compare the results using different training schemes. In the scheme "*fixed alignment*", we initialize the model, determine an alignment of the training data to the init-model and keep this alignment fixed until convergence. In the scheme "*alternating optimization*", we perform analogously to the previous experiments. That is, we initialize the model and alternate between re-aligning and parameter updates until convergence. The results of these experiments are given in table 3.

Interestingly, the final result is nearly independent of the initialization, which indicates that the alternating optimization is able to find a good set of parameters independent of the starting point. Only for the model initialised from the deformation aware Gaussian model, the alternating optimization has no effect. We believe that this model is stuck in a strong local optimum. However, if alternating optimization is not used, the other two models are clearly worse, which again highlights the importance of the alternating optimization. The training time for the different initialisations is similar where generally the model initialised with a log-linear model needs fewer iterations than the other two.

4.1 Transfer to MNIST & Comparison to the state-of-the-art

In table 5, we show how the model, with parameters (warprange, deformation sharing method, feature setup) tuned on the USPS dataset, performs on the

Table 5: Comparison of error rates[%], number of parameters and runtime of our deformation-aware log-linear model to state-of-the-art models.

	USPS		MNIST		
Model	# param.	ER	# param.	ER	run-time factor
log-lin. model+IDM using Sobel	69 130	4.04	211 690	1.63	50
+ abs(SobelHV)	74 250	3.84	227 370	1.32	100
+ deform. param. sharing	10 340	3.59	31 390	1.36	100
+ local context	92 190	3.69	282 270	1.50	900
log-linear model	2 570	8.2	7 850	7.4	1
+ abs(SobelHV)	5 130	5.5	15 690	3.0	2
single Gaussians	2 560	18.5	7 840	18.0	1
single Gaussians + IDM [4]	2 560	6.5	7 840	5.8	50
nearest neighbor [4]	1 866 496	5.6	47 040 000	3.1	729/6 000
nearest neighbor + IDM [4]	1 866 496	2.4	47 040 000	0.6	36 455/300 000
SVM	658 177	4.4	15 411 905	1.5	256/1 963
SVM + IDM [16]/[this work]	530 705	2.8	-	0.7	10 300/100 000
DBN [17]	640 610	-	1 665 010	1.3	210/ 220
conv. network [3]	-	-	180 580	0.4	-/25

MNIST dataset and compare the results for both datasets with several state-of-the-art results from the literature.

Additionally to the error rates, we give the total number of parameters that are necessary in the models to classify a test observation and the run-times of the different methods estimated from the number of basic mathematical operations relative to the fastest method. Note that sharing the deformation parameters has no noticeable impact on computation time, while each additional feature layer increases the run-time.

The results in the first block of table 5 are obtained using the deformation-invariant log-linear model. It can be seen that a combination of Sobel and absolute Sobel with *position and deformation independent α-pooling* improves the results. Additionally using local context does not lead to an improvement but rather to overfitting. All improvements using parameters optimized on the USPS dataset consistently transfer to improvements on the MNIST database, showing the good generalization capabilities of our model.

The first comparison result we give is that of a simple log-linear model, which due to the lack of deformation invariance performs significantly worse for both datasets, but is (along with the single Gaussian model) the fastest model. Both models only require to compare a test-observation to one prototype per class. The generative single Gaussian model with IDM, already performs much better but an IDM comparison is about 50 times as expensive as a simple component-wise comparison (due to the use of Sobel features and a deformation window of

5×5 pixels). For comparison, we additionally present results using a log-linear model using absolute Sobel features.

The nearest neighbor method needs as many comparison operations as there are training samples but the cost is independent of the number of classes, therefore the method is about 800 (resp. 6000) times slower than the simple log-linear model for the USPS dataset and for the MNIST dataset respectively. However, the nearest neighbor method with IDM obtains results among the best published results for both datasets.

The number of operations in the SVM depends on the number of support vectors. On both datasets, the number of support vectors is typically about 30% of all trainings samples and thus the method requires about a third of the run-time of the nearest neighbor classifier. For SVMs to include the IDM, we use radial basis kernel with a symmetric variant of the IDM-distance defined as $K_{\mathrm{IDM}}(X,V) = \exp\left(-\frac{\gamma}{2}(d_{\mathrm{idm}}(X,V) + d_{\mathrm{idm}}(V,X))\right)$, since non-symmetric kernels cause problems in training SVMs. Although this kernel is not necessarily positive definite, it was observed by Haasdonk [16] that in practice the training converges to a good result. We note that the symmetric IDM-distance is known to perform worse than the asymmetric one in nearest-neighbor experiments (3.4% instead of 2.4%). Nonetheless, the support vector machines obtain excellent results on both datasets, where the results on the MNIST database have been reported by [16] and the results on the USPS database have been obtained using our own implementation.

For further comparison we give two state-of-the-art results on the MNIST database using deep belief networks and convolutional neural networks. Both are based on neural networks where the deep belief network is proposed as a general learning technique [17] and no prior knowledge such as deformations is incorporated. The convolutional neural networks were designed with digit recognition in mind and are trained from a huge amount of automatically deformed training data [3]. The convolutional neural network obtains one of the best published results on the MNIST dataset despite its small size and efficient classification stage. However, the training phase for this network is computationally very expensive because the training data is automatically deformed and used for training several thousand times.

As an overview, it can be seen that our method compares favorably well to other methods. In particular in comparison with the other fast methods, only the convolutional neural networks, which are difficult to create and optimize, outperform our method with a comparable computation time. Furthermore, the small number of parameters in our model is a good indicator for its generalization performance which is underlined by the successful transfer of the parameters from the USPS dataset to the MNIST dataset.

5 Conclusion

We presented a new model that directly incorporates image deformations into a log-linear framework which achieves highly competitive results on well-known

handwritten character recognition tasks. It is possible to fine-tune the amount of deformation priors by sharing and it is shown that using fewer deformation prior parameters the model generalizes better. We also showed that the choice of the features is crucial to find good alignments and to obtain good results.

In the future we plan to investigate whether it is possible to extend the deformation-aware log-linear model toward log-linear mixture models analogously to the experiments reported in [12].

Acknowledgement. This work was partially funded by the DFG (Deutsche Forschungsgemeinschaft) under contract NE-572/6 and partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

References

1. DeCoste, D., Schölkopf, B.: Training invariant support vector machines. Machine Learning **46** (2002) 161–190
2. Haasdonk, B., Keysers, D.: Tangent distance kernels for support vector machines. In: ICPR, Quebec City, Canada (2002) 864–868
3. Simard, P.: Best practices for convolutional neural networks applied to visual document analysis. In: ICDAR, Edinburgh, Scotland (2003) 958–962
4. Keysers, D., Deselaers, T., Gollan, C., Ney, H.: Deformation models for image recognition. PAMI **29** (2007) 1422–1435
5. Keysers, D., Macherey, W., Ney, H., Dahmen, J.: Adaptation in statistical pattern recognition using tangent vectors. PAMI **26** (2004) 269–274
6. Memisevic, R., Hinton, G.: Unsupervised learning of image transformations. In: CVPR, Minneapolis, MN, USA (2007)
7. Lafferty, J., McCallum, A., Pereira., F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML. (2001)
8. Quattoni, A., Wang, S., Morency, L.P., Collins, M., Darrell, T.: Hidden conditional random fields. PAMI **29** (2007) 1848–1852
9. Uchida, S., Sakoe, H.: A survey of elastic matching techniques for handwritten character recognition. Trans. Information and Systems **E88-D** (2005) 1781–1790
10. Mori, S., Yamamoto, K., Yasuda, M.: Research on machine recognition of handprinted characters. PAMI **6** (1984) 386–405
11. Keysers, D., Och, F.J., Ney, H.: Maximum entropy and Gaussian models for image object recognition. In: DAGM Zürich, Switzerland (2002) 498–506
12. Heigold, G., Deselaers, T., Schlüter, R., Ney, H.: GIS-like estimation of log-linear models with hidden variables. In: ICASSP, Las Vegas, NV, USA (2008) 4045–4048
13. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: ICNN, San Francisco, CA, USA (1993)
14. ftp://ftp.kyb.tuebingen.mpg.de/pub/bs/data/
15. http://yann.lecun.com/exdb/mnist/
16. Haasdonk, B.: Transformation Knowledge in Pattern Analysis with Kernel Methods. PhD thesis, Albert-Ludwigs-Universität Freiburg (2005)
17. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation **18** (2006) 1527–1554