Linear Discriminant Analysis and Discriminative Log-linear Modeling*

Daniel Keysers and Hermann Ney
Lehrstuhl für Informatik VI – Computer Science Department
RWTH Aachen University – D-52056 Aachen, Germany
{keysers, ney}@informatik.rwth-aachen.de

Abstract

We discuss the relationship between the discriminative training of Gaussian models and the maximum entropy framework for log-linear models. Observing that linear transforms leave the distributions resulting from the log-linear model unchanged, we derive a discriminative linear feature reduction technique from the maximum entropy approach and compare it to the well-known linear discriminant analysis. From experiments on different corpora we observe that the new technique performs better than linear discriminant analysis if the dimensionality of the feature space is large with respect to the number of classes.

1 Introduction

Linear discriminant analysis (LDA) is a widely used tool in pattern recognition. It provides a linear transformation of the feature space that is generally combined with a feature reduction. The derivation of LDA can be based on the assumption that the class conditional distributions are Gaussians. We show that there exists a strong relation between discriminative log-linear models with the appropriate choice of features and discriminative training of Gaussian models. This connection leads to a counterpart of LDA in the context of log-linear models.

When using log-linear models for the class posterior in classification, it can be observed that the model distributions are not changed by any non-singular linear transformation of the feature space. Furthermore, no linear feature reduction can improve the log-likelihood of the posterior on the training data. As these models (also called maximum entropy models) are successfully used in a wide range of pattern recognition applications, the question is raised, how this framework can be used to estimate a discriminative linear feature reduction transformation.

This paper provides an answer to this question resulting in a maximum entropy linear discriminant analysis (MELDA). We show that the solution has two properties: 1. The transformation preserves exactly that linear subspace of the original feature space that is orthogonal to the class boundaries chosen by the maximum entropy training. 2. The solution follows from an appropriate choice for the degrees of freedom in the maximum entropy solution for the transformation matrix when considering the connection between maximum entropy training and Gaussian models.

We discuss experiments on different datasets comparing LDA and MELDA that suggest that LDA leads to better results for tasks with lower dimensionality of the feature space, whereas MELDA performs better on tasks with high dimensionality.

*This work was partially funded by the DFG (Deutsche Forschungsgemeinschaft) under contract NE-572/6.

2 Maximum entropy and Gaussian models

To classify an observation, we use Bayes' decision rule

$$r(\boldsymbol{x}) = \operatorname*{argmax}_k \{p(k|\boldsymbol{x})\} = \operatorname*{argmax}_k \{p(k) \cdot p(\boldsymbol{x}|k)\} .$$

Here, $p(k|\boldsymbol{x})$ is the class posterior probability of class $k \in \{1, \ldots, K\}$ given the observation $\boldsymbol{x} \in \mathbb{R}^D$, $p(k)$ is the a priori probability, $p(\boldsymbol{x}|k)$ is the class conditional probability for $\boldsymbol{x}$ given k and $r(\boldsymbol{x})$ is the decision of the classifier.

If we denote by Λ the set of free parameters of the distribution, the conventional maximum likelihood approach consists of choosing the parameters $\hat{\Lambda}$ maximizing the (log-) likelihood of the class conditional distribution on the training data $\{(\boldsymbol{x}_n, k_n)\}, n = 1, \ldots, N$. Alternatively, we can maximize the log-likelihood of the class posteriors,

$$\hat{\Lambda} = \operatorname*{argmax}_{\Lambda} \sum_n \log p_{\Lambda}(k_n|\boldsymbol{x}_n) \ , \qquad (1)$$

which is also called discriminative training, since the information of out-of-class data is used.

The principle of maximum entropy has origins in statistical thermodynamics, is related to information theory, and has been applied to pattern recognition tasks such as language modeling [1] and text classification. Applied to classification, the basic idea is the following: We are given information about a probability distribution by samples from that distribution (the training data). Now, we choose the distribution such that it fulfills the constraints implied by that information (more precisely: the observed marginal distributions of the chosen feature functions) but otherwise has the highest possible entropy. Consider a set of feature functions $\{f_i\}, i = 1, \ldots, I$ that are supposed to compute 'useful' information for classification:

$$f_i \ : \ \mathbb{R}^D \times \{1, \ldots, K\} \to \mathbb{R} \ : \ (\boldsymbol{x}, k) \mapsto f_i(\boldsymbol{x}, k)$$

It can be shown that the resulting distribution that maximizes the entropy has the following log-linear form:

$$p_\Lambda(k|\boldsymbol{x}) = \frac{\exp\left[\sum_i \lambda_i f_i(\boldsymbol{x},k)\right]}{\sum_{k'}\exp\left[\sum_i \lambda_i f_i(\boldsymbol{x},k')\right]}, \quad \Lambda = \{\lambda_i\}. \qquad (2)$$

This optimization problem is convex and has a unique global maximum, which is also the solution to the dual problem: Maximize the log probability (1) on the training data using the model (2). (Note that there may be more than one maximizing parameter set Λ, though.) Furthermore, effective algorithms are known that compute the global maximum of the log probability (1) given a training set. On the one hand, the algorithm known as generalized iterative scaling [2] and related algorithms can be proved to converge to the global maximum. On the other hand, we can also use general optimization strategies as e.g. conjugate gradient methods due to the convexity of the problem. The resulting model (2) is also known as logistic regression for $K=2$ or as multiclass logistic regression for $K>2$.

The crucial problem in maximum entropy modeling is the choice of the appropriate feature functions $\{f_i\}$. In general, these functions depend on the classification task considered. The straight forward way to define feature functions for classification purposes is to directly use the features provided for the specific task. Consider therefore the following first-order feature functions:

$$f_{k,i}(\boldsymbol{x},k') = \delta(k,k')\; x_i\;, \qquad f_k(\boldsymbol{x},k') = \delta(k,k')\;,$$

where $\delta(k,k') := 1$ if $k = k'$ and 0 otherwise denotes the Kronecker delta function. Here, the Kronecker delta is necessary to distinguish between the different classes. The functions f_k allow for a log-linear offset in the posterior probabilities. Now, using the properties of the Kronecker delta, the structure of the posterior probabilities becomes

$$p_\Lambda(k|\boldsymbol{x}) = \frac{\exp\left[\alpha_k + \sum \lambda_{k,i} x_i\right]}{\sum_{k'}\exp[\alpha_{k'} + \sum \lambda_{k',i} x_i]} = \frac{\exp[\alpha_k + \boldsymbol{\lambda}_k^T \boldsymbol{x}]}{\sum_{k'}\exp[\alpha_{k'} + \boldsymbol{\lambda}_{k'}^T \boldsymbol{x}]}, \qquad (3)$$

where $\Lambda = \{\lambda_{k,i}, \alpha_k\}$ and α_k denotes the coefficient for the feature function f_k.

To observe the connection to Gaussian models we use Bayes' rule and the definition of a Gaussian density and rewrite the class posterior probability of a Gaussian model for $p(\boldsymbol{x}|k)$ with pooled covariance matrix $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$ as [3]:

$$\begin{aligned} p(k|\boldsymbol{x}) &= \frac{p(k)\;\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma})}{\sum_{k'} p(k')\;\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_{k'}\boldsymbol{\Sigma})} \\ &= \frac{\exp\left[(\log p(k) - \frac{1}{2}\boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k) + (\boldsymbol{\mu}_k^T\boldsymbol{\Sigma}^{-1})\boldsymbol{x}\right]}{\sum_{k'}\exp\left[(\log p(k') - \frac{1}{2}\boldsymbol{\mu}_{k'}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_{k'}) + (\boldsymbol{\mu}_{k'}^T\boldsymbol{\Sigma}^{-1})\boldsymbol{x}\right]} \\ &= \frac{\exp[\alpha_k + \boldsymbol{\lambda}_k^T\boldsymbol{x}]}{\sum_{k'}\exp[\alpha_{k'} + \boldsymbol{\lambda}_{k'}^T\boldsymbol{x}]} \end{aligned} \qquad (4)$$

Note that the terms not depending on k cancel. As result, we see that for unknown class priors $p(k)$ the resulting model (4) is identical to the maximum entropy model (3). We can conclude that the discriminative training criterion (1) for the Gaussian model with pooled covariance matrices results in the same functional form as the maximum entropy model for first-order features. This allows us to use the well understood algorithms for maximum entropy estimation to estimate the parameters of a Gaussian model discriminatively.

If we repeat the same argument for the case of Gaussian densities without pooling of the covariance matrices, we find that we can again establish a correspondence to a maximum entropy model [3]. In the resulting model, the second-order terms depend on k and therefore do not cancel. The coefficients of these terms correspond to the entries in the negative inverse covariance matrix, meaning that these parameters can be estimated using a maximum entropy model with the additional second-order feature functions

$$f_{k,i,j}(\boldsymbol{x},k') = \delta(k,k')x_i x_j, \quad i \geq j.$$

One interesting consequence of using the corresponding maximum entropy model and estimation is that we implicitly relax the constraints on the covariance matrices to be positive (semi-) definite. Therefore, the resulting model is not exactly equivalent to a Gaussian model. Experiments have shown that this estimation procedure is very robust even for a large number of features [3]. It allows the estimation of the equivalent of full, class-specific covariance matrices with improving performance, in circumstances where this is not possible in a conventional maximum likelihood approach. Because of this robustness of the second-order features in the maximum entropy framework these were also used to increase the number of features for MELDA.

These results are different from the approach taken in [4], where the authors derive discriminative models for Gaussian densities based on priors of the parameters and the minimum relative entropy principle. Their solution results in discriminatively trained weights for the training data and therefore preserves the mentioned constraints.

3 Maximum entropy LDA

LDA is a well-known tool in pattern recognition. Using one of the possible formulations, LDA aims at minimizing intra-class variance with respect to inter-class variance using a linear transformation $\widetilde{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$, i.e. minimize

$$\min_{\boldsymbol{A}} \sum_n ||\boldsymbol{A}(\boldsymbol{x}_n - \boldsymbol{\mu}_{k_n})||^2, \qquad \sum_{n,k} ||\boldsymbol{A}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)||^2 \stackrel{!}{=} c \qquad (5)$$

One computational method to determine the LDA matrix $\boldsymbol{A}$ is to compute the within class scatter matrix $\boldsymbol{W}$ and the between class scatter matrix $\boldsymbol{B}$ and then solve the generalized eigenvalue problem $\boldsymbol{B}\boldsymbol{A}^T = \boldsymbol{\lambda}\boldsymbol{W}\boldsymbol{A}^T$. This requires the computation of the scatter matrices, which can be computationally inefficient if there are more features than training samples. In this case, we can reduce the size of the scatter matrices by applying a singular value decomposition to

the data, computing a projected representation of the data in span($\{\boldsymbol{x}_n\}$), computing the LDA in this vector space with lower dimensionality and then reversing the projection.

To derive the maximum entropy LDA, first note that in the log-linear model any non-singular linear transformation of the feature space leaves the maximum class posterior distribution unchanged. Consider a transformation $\widetilde{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$:

$$p_{\widetilde{\Lambda}}(k|\widetilde{\boldsymbol{x}}) = \frac{\exp\left[\widetilde{\alpha}_k + \widetilde{\boldsymbol{\lambda}}_k^T \boldsymbol{A}\,\boldsymbol{x}\right]}{\sum_{k'}\exp\left[\widetilde{\alpha}_{k'} + \widetilde{\boldsymbol{\lambda}}_{k'}^T \boldsymbol{A}\,\boldsymbol{x}\right]} = \frac{\exp\left[\alpha_k + \boldsymbol{\lambda}_k^T \boldsymbol{x}\right]}{\sum_{k'}\exp\left[\alpha_{k'} + \boldsymbol{\lambda}_{k'}^T \boldsymbol{x}\right]} = p_\Lambda(k|\boldsymbol{x}) \quad (6)$$

with ${\boldsymbol{\lambda}_k}^T = \widetilde{\boldsymbol{\lambda}}_k^T \boldsymbol{A}$ and $\alpha_k = \widetilde{\alpha}_k$. From the uniqueness of the maximum entropy distribution it follows immediately that the distribution is not changed. Now, even if $\boldsymbol{A}$ does not have full rank, all solutions $\widetilde{\Lambda}$ have at least one corresponding solution Λ, i.e. the criterion (1) can never be improved by applying a linear transformation to the feature space. This observation motivates the following approach: First compute the parameters in the original feature space, then choose the linear transformation accordingly.

Assume $D \geq K$. We want to estimate a transformation matrix $\widetilde{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ with $\boldsymbol{A} \in \mathbb{R}^{(K-1)\times D}$ maximizing the log-likelihood of the posterior of the log-linear model $p_{\widetilde{\Lambda}}(k|\widetilde{\boldsymbol{x}})$. The functional form (3) implies that in the computation of $p_\Lambda(k|\boldsymbol{x})$ those components of $\boldsymbol{x}$ that are orthogonal to all of the difference vectors $\{\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k'}\}$ do not change the result. This implies that only the projections of $\boldsymbol{x}$ onto the subspace

$$\text{span}\big(\{\boldsymbol{\lambda}_k - \boldsymbol{\lambda}_{k'}\}\big) = \text{span}\big(\{\boldsymbol{\lambda}_2 - \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_3 - \boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_K - \boldsymbol{\lambda}_1\}\big)$$

influence the posterior $p_\Lambda(k|\boldsymbol{x})$. Therefore, defining the transformation matrix as

$$\boldsymbol{A} = \big((\boldsymbol{\lambda}_2 - \boldsymbol{\lambda}_1), (\boldsymbol{\lambda}_3 - \boldsymbol{\lambda}_1), \ldots, (\boldsymbol{\lambda}_K - \boldsymbol{\lambda}_1)\big)^T \quad (7)$$

achieves the required result. The transformation retains those parts of the feature space orthogonal to the linear class boundaries chosen by maximum entropy training.

To observe the second property of the chosen transformation, we consider the relationship between the Gaussian model and the log-linear model again. From the equivalence in (4) we see that for the estimated parameters we have:

$$\boldsymbol{\lambda}_k^T = \widetilde{\boldsymbol{\lambda}}_k^T \boldsymbol{A} = \widetilde{\boldsymbol{\mu}}_k^T \widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{A} \quad (8)$$

This equation is under-determined for $\boldsymbol{A}$ if only the parameter vectors $\{\boldsymbol{\lambda}_k\}$ are known. After the transformation, we choose the mean vectors to be the null vector and the $K-1$ unit vectors, respectively, and the covariance matrix to be the identity matrix:

$$(\widetilde{\boldsymbol{\mu}}_1, \ldots, \widetilde{\boldsymbol{\mu}}_K) = \left(\begin{array}{c|c} 0 & \\ \vdots & \boldsymbol{I}_{K-1} \\ 0 & \end{array}\right), \qquad \widetilde{\boldsymbol{\Sigma}} = \boldsymbol{I}_{K-1}$$

This restriction leaves (8) satisfiable, as we have additional degrees of freedom in the log-linear model and we can always transform the maximum entropy solution by setting $\boldsymbol{\lambda}_k \leftarrow \boldsymbol{\lambda}_k - \boldsymbol{\lambda}_1$, forcing $\boldsymbol{\lambda}_1 = \boldsymbol{0}$. Thus, we obtain the same solution for the transformation matrix $\boldsymbol{A}$ as given in (7).

A further connection between LDA and MELDA can be observed if we derive an expression similar to (5) for MELDA. Consider again a Gaussian model as in (4) and (6) where we now neglect the class priors $p(k)$ and assume $\boldsymbol{\Sigma} = \boldsymbol{I}$. We obtain the expression:

$$\operatorname*{argmax}_{\boldsymbol{A}} \sum_n \log p_{\widetilde{\Lambda}}(k_n|\widetilde{\boldsymbol{x}}_n) =$$

$$\operatorname*{argmin}_{\boldsymbol{A}} \sum_n \tfrac{1}{2}||\boldsymbol{A}(\boldsymbol{x}_n - \boldsymbol{\mu}_{k_n})||^2 + \log \sum_k \exp\left[-\tfrac{1}{2}||\boldsymbol{A}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)||^2\right]$$

In this formulation we do not need an additional constraint and we observe that in the sum over all classes the distances to the closer competing class are exponentially more important than classes with means far away. This is not the case for the conventional LDA and emphasizes the discrimination between directly competing classes.

Table 1. Corpus statistics for the databases.

name	K	D	$D/(K-1)$	N(train)	N(test)
MONK	2	17	17.0	124	432
MONK^2	2	153	153.0	124	432
DNA	3	180	90.0	2 000	1 186
DNA^2	3	16 290	8 145.0	2 000	1 186
LETTER^2	26	136	5.4	15 000	5 000
USPS	10	256	28.4	7 291	2 007
USPS^2	10	4 930	547.8	7 291	2 007

4 Experiments and results

Databases. The experiments were performed on three corpora from the UCI and STATLOG database, respectively [5, 6] and the USPS handwritten digits task. The corpora were chosen to cover different properties with respect to the number of classes and features and with respect to the size. The statistics of the corpora are summarized in Table 1, where: K is the number of classes, D is the number of features, $D/(K-1)$ is the factor of feature reduction for LDA and MELDA, and N is the number of samples. From each of the corpora we created a "squared" version (indicated by a superscript 2) by using all feature products $x_i x_j, i \geq j$ as additional features. This procedure was based on the finding that the performance of the log-linear classifier generally improves with larger number of features. The squared corpora have $D(D+1)/2$ features with the exception of the USPS corpus. Here, a subset of the product features was chosen based on pixel neighborhoods.

MONK is an artificial decision task with categorical features also known as the monk's problem. For the DNA task, the goal is to detect gene intron/exon and exon/intron boundaries given part of a DNA sequence. For the experiments, any categorical features were transformed into binary features. The LETTER corpus consists of printed

Table 2. Experimental results: error rates [%].

FR		M	M^2	D	D^2	L^2	U	U^2
NONE	SG	28.5	22.7	9.9	11.2	44.2	19.4	25.0
	NN	21.3	21.3	23.4	33.1	4.7	5.6	7.2
	ME	28.7	0.9	6.2	5.1	9.5	8.8	7.2
LDA	SG	26.6	30.8	6.7	52.4	18.4	11.5	22.2
	NN	27.5	28.5	6.2	52.2	4.5	10.9	22.9
	ME	26.6	30.8	4.4	53.3	13.9	11.0	22.9
MELDA	SG	25.0	0.9	9.1	7.5	42.0	32.4	27.4
	NN	26.2	0.9	6.0	5.3	5.9	14.7	12.0
	ME	28.7	0.9	6.2	5.1	9.5	8.8	7.2

characters that were preprocessed and a variety of different features was extracted. For this corpus, we only consider its squared version LETTER^2, since the original corpus has $D = 16 < 26 = K$. The USPS corpus consists of images of handwritten digits normalized to $16 \times 16 = 256$ pixels, where the pixel values are directly taken as features.

Results. The results of our experiments are summarized in Table 2. On each of the corpora (names abbreviated by first letter), we compare the error rates of three different classifiers for each of the feature reduction (FR) methods (including no feature reduction). The three classifiers are the single Gaussian (SG, using pooled diagonal covariance matrices), the nearest neighbor (NN, using Euclidean distance), and the maximum entropy (ME) classifier. Note that in the lines with no feature reduction, the number of features used is larger by a factor of up to 8145.

As expected, the LDA performs better for the Gaussian classifier and MELDA performs better for the maximum entropy classifier as both methods are especially suited for these cases. Nevertheless in both these cases the general tendency is preserved: the relative performance of MELDA with respect to LDA increases with the feature reduction factor $D/(K-1)$. To illustrate this effect, Figure 1 shows the relative improvement of MELDA over LDA for the nearest neighbor classifier $1\text{–err}_{\text{MELDA}}/\text{err}_{\text{LDA}}$ (values > 0 indicate that MELDA performs better than LDA) for LETTER^2, MONK, USPS, DNA, MONK^2, USPS^2, DNA^2 in the order of feature reduction factor $D/(K-1)$. One further result is that in all cases where the squared and the original corpus were used, MELDA performs better in the artificially enlarged features space, while LDA performs worse.

Note that most of the presented results are not competitive with the best error rates obtained on the used corpora. This is due to the fact that none of the classifiers or feature reduction methods were tuned to the specific tasks (e.g. the performance of LDA on the USPS corpus can be considerably improved by first clustering the corpus into 40 clusters and then computing an LDA matrix resulting in a 39-dimensional feature space). However, this does not weaken the obtained results as we are interested in the relative performance of MELDA and LDA, knowing that LDA is a widely used technique.

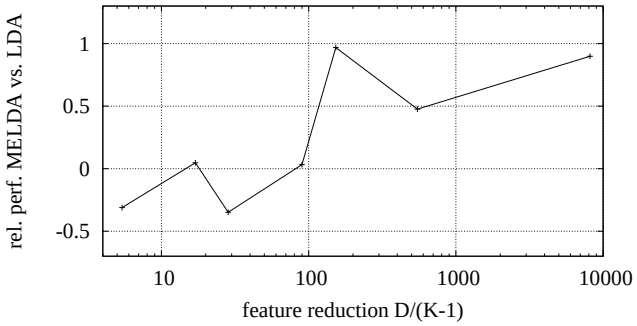

Figure 1. Relative performance MELDA / LDA.

5 Conclusion

We presented a linear feature reduction method based on the maximum entropy framework and log-linear models for the class posterior. We compared this MELDA to classical LDA both theoretically (also showing the connection between log-linear models and discriminative training of Gaussian models) and by experiments on several corpora. The main result is that MELDA performs better than LDA when the feature reduction factor $D/(K-1)$ is large. On some corpora using a very small number of features (even one or two) already produces very good results using MELDA (e.g. MONK^2, DNA^2).

Regarding LDA and log-linear models Hastie et al. write [7, p.105]: "It is generally felt that logistic regression is a safer, more robust bet than the LDA model, relying on fewer assumptions. It is our experience that the models give very similar results, [...]" The experiments reported in this paper suggest that the log-linear model gains in robustness and produces better results when the number of features with respect to the number of classes is large.

References

[1] A.L. Berger, S.A. Della Pietra, and V.J. Della Pietra. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–72, March 1996.

[2] J.N. Darroch and D. Ratcliff. Generalized Iterative Scaling for Log-Linear Models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

[3] D. Keysers, F.J. Och, and H. Ney. Maximum Entropy and Gaussian Models for Image Object Recognition. In *Pattern Recognition, 24th DAGM Symposium*, Zürich, Switzerland, LNCS 2449, pp. 498–506, September 2002.

[4] T. Jaakkola, M. Meila, and T. Jebara. Maximum Entropy Discrimination. In *Adv. in Neural Inf. Proc. Systems 12*, MIT Press, Cambridge, MA, pp. 470–476, 2000.

[5] C.J. Merz, P.M. Murphy, and D.W. Aha. UCI Repository of Machine Learning Databases. Univ. California, Irvine, 1997. http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[6] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, eds. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, 1994. Datasets: http://www.liacc.up.pt/ML/statlog/datasets.html.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, NY, 2001.