On a greedy learning algorithm for dPLRM with applications to phonetic feature detection

Tor André Myrvoll[*†] *and Tomoko Matsui*[†]

†The Institute of Statistical Mathematics, Tokyo, Japan
*Department of Electronics and Telecommunications, NTNU, Trondheim, Norway
myrvoll@iet.ntnu.no, tmatsui@ism.ac.jp

Abstract

In this work we investigate the use of a greedy training algorithm for the dual Penalized Logistic Regression Machine (dPLRM), and our target application is detection of broad class phonetic features. The use of a greedy training algorithm is meant to alleviate the infeasible memory and computational demands that arises during the learning phase when the amount of training data increases. We show that using only a subset of the training data, chosen in a greedy manner, we can achieve as good as or better performance as when using the full training set. We can also train dPLRMs using data sets that are significantly larger than what our current computational resources can accommodate when using non-greedy approaches.

Index Terms: machine learning, greedy algorithms, feature detection

1. Introduction

Machine learning techniques are again becoming increasingly more popular in the speech research community. Previous attempts of creating hybrid systems using neural networks are well documented (see [1]), but the performance never bested that of "traditional" hidden Markov Model (HMM) based systems.

The new generation of machine learning algorithms are best exemplified by the support vector machine (SVM), which is probably the best known example of what is often referred to as *kernel methods*. Without describing the SVM in detail we want to emphasize some of the strengths of this approach. The most attractive property of the SVM is the convexity of the objective function, which guarantees any existing optimum to be global. The SVM also takes care of generalization since its objective function is a weighted sum of empirical loss (training set performance) and a regularization term constraining the complexity of the classifier or regression function.

There are two main drawbacks to the original SVM. Presented with a test example in the form of a feature vector, $\mathbf{x}$, the SVM, represented here by a function $f(\cdot)$, will classify the example into one of two classes according to the sign of the evaluation $f(\mathbf{x})$. Although this works well for a simple classification task, the lack of a probabilistic formulation makes the SVM hard to integrate in a statistically based ASR system. Another drawback is the computational complexity of the training, as well as testing, when the amount of training data increases.

This work was done while T. A. Myrvoll was a visiting researcher at ISM, the Institute of Statistical Mathematics.

dPLRM solves the first problem by directly modeling the posterior class probabilities given the observations, while still yielding a convex optimization problem. On the other hand, the problem with the computational complexity is amplified as the dPLRM does not automatically yield sparse solutions as is the case for SVM. We address this problem by selecting which training observations to use in a greedy manner. The resulting algorithm will be demonstrated on the problem of classifying broad class phonetic features.

In the next sections we present the theory behind dPLRM and demonstrate how it can be formulated in terms of reproducing kernel Hilbert spaces. This is in turn used to formulate a greedy search algorithm used to select a sparse subset of the training data.

2. dual Penalized Logistic Regression Machines

The dPLRM was first proposed by Tanabe in [2, 3, 4], and has subsequently been utilized for speaker identification in [5, 6]. In this section we will give a brief presentation of dPLRM and the corresponding optimization techniques based on nonlinear conjugate gradient descent [7].

2.1. Modeling the conditional class probability

Let $\mathbf{x} \in \mathbb{R}^d$ be an observation vector and $c_i \in \mathcal{C}$ be one out of a finite number of classes. Our goal is to find an approximation to the posterior class probabilities $\{p(c_i|\mathbf{x})\}$. In the plug-in-MAP (PI-MAP) paradigm this is achieved through the use of Bayes theorem. The generative models $\{p(\mathbf{x}|c_i\}$ are estimated, and any subsequent decisions about which class c_i is the most likely is based on the relation

$$\hat{p}(c_i|\mathbf{x}) \propto p(\mathbf{x}|c_i)p(c_i), \tag{1}$$

where $p(c_i)$ is either a prior distribution of the classes or estimated from occurrence counts.

The dPLRM models the conditional class probability directly using a logistic function. Let us assume that we have a finite number of training examples,

$$\{z_n\} = \{(\mathbf{x}_n, y_n)\},\ n \in 1 \ldots N.$$

Again, $\mathbf{x}_n \in \mathbb{R}^d$ and $y_n \in \mathcal{C}$. We also define a general similarity measure between observations $\mathbf{x}$ as

$$k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}.$$

In general we will assume that $k(\cdot,\cdot)$ is a Mercer-kernel [8], but for now we can think of it as a simple inner product in $\mathbb{R}^d$,

$$k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle.$$

We can now define a class-dependent similarity measure between an observation $\mathbf{x}$ and the training observations $\{\mathbf{x}_n\}$ as

$$f(\mathbf{x}|c_i) = \sum_{n=1}^{N} \alpha_i(n) k(\mathbf{x}_n, \mathbf{x}). \quad (2)$$

When embedding these similarity measures in a logistic function we implicitly define set of functions parameterized by the parameters $\{\alpha_i(n)\}$, where $n \in 1 \dots N$ and $i \in 1 \dots |\mathcal{C}|$. By optimizing the parameters we can find the "closest" (in some for now undefined sense) approximation to the conditional class probabilities

$$p(c_i|\mathbf{x}) = \frac{e^{f(\mathbf{x}|c_i)}}{\sum_{c_j \in \mathcal{C}} e^{f(\mathbf{x}|c_j)}}. \quad (3)$$

To avoid overtraining a regularization term, $\frac{\delta}{2}\text{tr}\{\Gamma V K V^T\}$, is added to the negative log likelihood during the estimation phase. Here K is the Gram matrix containing all kernel products, V contains the parameters $\alpha_i(n)$ and Γ is a matrix with the observation count per class on the diagonal. Clearly, the term can be interpreted as a Gaussian prior on the model parameters.

The parameter optimization is usually done using conjugate gradient descent. Although this type of optimization technique is infeasible in most cases where the parameter space is large, in the case of the dPLRM the Hessian is very structured. This structure simplifies the computations significantly and yields a very effective search. For details on the derivation of the conjugate gradient descent algorithm for dPLRM, consult [2, 3, 4].

3. A greedy training scheme

The conjugate gradient descent referred to in the previous section lets us train dPLRMs efficiently provided that there is an adequate amount of system memory available. The problem is that the memory required increases quadratically with the number of training examples, as is obvious from the dependence on the kernel product matrix K. Intuitively, as the number of training examples increase, we should be able to find a subset of the total data set that is a good enough representation for our purposes. More formally, given a set of N training examples, pick out the $M \ll N$ examples that best represent the training set in the sense of minimizing the negative log posterior class probability of the total data.

Clearly a brute force approach is infeasible as it becomes a combinatorial problem. In what follows we reformulate the dPLRM to take advantage of the rich theory of reproducing kernel Hilbert spaces (RKHS). We then use the greedy search approach presented in [9] for use with clique selection in conditional random fields (CRF).

3.1. An RKHS formulation of dPLRM

As defined in section 2.1 the training data are pairs $z_n = (\mathbf{x}_n, y_n)$. We must first define some kernel,

$$\bar{k} : (\mathbb{R}^d \times \mathcal{C}) \times (\mathbb{R}^d \times \mathcal{C}) \to \mathbb{R}$$

on this pair, and the choice for this work is

$$\bar{k}(z, z') = \delta(y, y') k(\mathbf{x}, \mathbf{x}'), \quad (4)$$

where $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a Mercer kernel.

It is well known that for a Mercer kernel $k(\cdot, \cdot)$, functions of the form

$$f(\cdot) = \sum_{l=1}^{L} a_l k(x_l, \cdot) \quad (5)$$

form an inner product space induced by the following inner product definition,

$$\langle k(x, \cdot), k(x', \cdot) \rangle_K = k(x, x'), \quad (6)$$

and that its completion is a Hilbert space, $\mathcal{H}_K$. Let us write our dPLRM in terms of a function $f \in \mathcal{H}_K$,

$$p(c_i|\mathbf{x}) = \frac{e^{f(x, c_i)}}{\sum_{c_j \in \mathcal{C}} e^{f(x, c_j)}} \quad (7)$$

The object function we want to minimize, the negative log posterior probability of the training data plus a regularization term, can now be written in form of a functional on $\mathcal{H}_K$,

$$L(f) = -\sum_{n=1}^{N} f(\mathbf{x}_n, y_n) + \log \sum_{c_j \in \mathcal{C}} e^{f(\mathbf{x}_n, c_j)} + \Omega(\|f\|_K), \quad (8)$$

where $\Omega : \mathbb{R}^+ \to \mathbb{R}^+$ is a strictly increasing function.

Assume that we want to find the optimal function $f^* \in \mathcal{H}_K$, so that the functional in equation (8) is minimized. From [9] we have the Representer Theorem for CRFs, which states that the solution is of the form,

$$f^*(\cdot) = \sum_{n=1}^{N} \sum_{c_j \in \mathcal{C}} \alpha_n(c_j) \bar{k}((\mathbf{x}, c_j), \cdot). \quad (9)$$

Clearly, plugging (9) into (7) gives us an equivalent formulation of dPLRM.

It should be mentioned that the form of the dPLRM regularization term, $\text{trace}\{\Gamma V K V^T\}$ is inconsistent with $\Omega(\|f\|_K)$ (remember that V together with K is a representation of f). This is not a problem, however. Let $g = f^* + h$, where $h \perp f^*$ lies in a subspace orthogonal to the span of f^*. Then the norm of g is

$$\|g\|_K^2 = \|f^*\|_K^2 + \|h\|_K^2,$$

and so clearly

$$\Omega(\|f^*\|_K) \leq \Omega(\|f^* + h\|_K) \; \forall h \in \text{span}(f^*)^\perp.$$

The same holds true for the dPLRM regularization term, as the orthogonality of f^* and h implies that we can write

$$\begin{aligned} &\text{trace}\{\Gamma V K V^T\} \\ =&\text{trace}\{\Gamma_{f^*} V_{f^*} K_{f^*} V_{f^*}^T\} + \text{trace}\{\Gamma_h V_h K_h V_h^T\}, \end{aligned}$$

which again shows that the minimizer is indeed f^*.

3.2. Greedy search using the Gâteaux derivative

In this section we will use the Gâteaux derivative, or *functional gradient*, to pick kernels from the full training set. The Gâteaux derivative of a functional L at f in the direction h is defined as,

$$\frac{\partial}{\partial \epsilon} L(f + \epsilon h)\bigg|_{\epsilon=0}. \quad (10)$$

We use this definition to pick a subset of the training data in the following manner:

1. Start with an empty function $f = 0$.
2. Until some convergence criteria is met:
   (a) Calculate the Gâteaux derivative at f in the direction of every
   $$h_n(\cdot) = \bar{k}(z_n, \cdot) \tag{11}$$
   (b) Pick the h_n corresponding to the largest Gâteaux derivative and add it to f.
   (c) Optimize the kernel weights $\alpha_n(c_i)$.
   (d) Repeat

Let us now calculate the Gâteaux derivative at f in the general direction h for the functional in equation (8). We split our calculations into three parts – the terms corresponding to the numerator and denominator of the logistic function, and the regularization term. The numerator term yields the following,

$$\begin{aligned} &\frac{\partial}{\partial \epsilon} - \sum_{n=1}^{N} f(z_n) + \epsilon h(z_n) \Big|_{\epsilon=0} \\ &= - \sum_{n=1}^{N} h(z_n), \end{aligned} \tag{12}$$

which can be interpreted as the *empirical expectation* of h.

The Gâteaux derivative corresponding to the denominator term is,

$$\begin{aligned} &\frac{\partial}{\partial \epsilon} \sum_{n=1}^{N} \log \sum_{c_j \in \mathcal{C}} e^{f(\mathbf{x}_n, c_j) + \epsilon h(\mathbf{x}_n, c_j)} \Big|_{\epsilon=0} \\ &= \sum_{n=1}^{N} \frac{\sum_{c_j \in \mathcal{C}} h(\mathbf{x}_n, c_j) e^{f(\mathbf{x}_n, c_j) + \epsilon h(\mathbf{x}_n, c_j)}}{\sum_{c_j \in \mathcal{C}} e^{f(\mathbf{x}_n, c_j) + \epsilon h(\mathbf{x}_n, c_j)}} \Big|_{\epsilon=0} \\ &= \sum_{n=1}^{N} \sum_{c_j \in \mathcal{C}} p(c_j|\mathbf{x}) h(\mathbf{x}_n, c_j), \end{aligned} \tag{13}$$

which we can interpret as the probabilistic expectation with respect to the current model.

Finally we address the regularization term. We assume that f and h have no kernels in common and model the additional terms by augmenting the V and K matrices (Γ is unaffected by this augmentation),

$$\begin{aligned} \bar{V} &= [V \; \epsilon W] \\ \bar{K} &= \begin{bmatrix} K & K' \\ K' & K'' \end{bmatrix}. \end{aligned}$$

The Gâteaux derivative now becomes

$$\begin{aligned} &\frac{\partial}{\partial \epsilon} \frac{\delta}{2} \text{trace}\{\Gamma \bar{V} \bar{K} \bar{V}^T\} \Big|_{\epsilon=0} \\ &= \frac{\partial}{\partial \epsilon} \frac{\delta}{2} \text{trace}\{\Gamma (V K V^T + \epsilon V K' W^T \\ &\qquad + \epsilon W K' V^T + \epsilon^2 W K'' W^T)\} \Big|_{\epsilon=0} \\ &= \delta \text{trace}\{\Gamma V K' W^T\} \end{aligned} \tag{14}$$

Putting these three terms together the final Gâteaux derivative is,

$$\begin{aligned} &\frac{\partial}{\partial \epsilon} L(f + \epsilon h) \Big|_{\epsilon=0} \\ &= - \sum_{n=1}^{N} h(z_n) + \sum_{n=1}^{N} \sum_{c_j \in \mathcal{C}} p(c_j|\mathbf{x}) h(\mathbf{x}_n, c_j) \\ &\quad + \delta \text{trace}\{\Gamma V K' W^T\} \end{aligned} \tag{15}$$

We have now formulated an approach to select an "optimal" kernel in a greedy manner. The solution is still not entirely satisfying as we still have to calculate all the kernel products, $k(x_i, x_j)$, while searching through all of the observations in the training set. In the next section we present a suboptimal approach that utilizes a neighborhood graph to reduce the number of observations we need to investigate.

3.3. Neighborhood search

A very simple, but nevertheless effective approach to localized search is to create a graph from the training data using the k-nearest neighbor (kNN) algorithm. Here we calculate the distances in kernel space,

$$\|x - y\|_{\mathcal{H}} = k(x, x) - 2k(x, y) + k(y, y), \tag{16}$$

and use these distances to find the k closest observations to any x.

The idea is now the following, starting with an arbitrary observation, x:

1. Calculate the Gâteaux derivative with respect to x and every $x' \in \mathcal{N}(x)$, the neighborhood of x.
2. Choose the observation $\hat{x}$ that had the largest corresponding gradient. If $\hat{x} = x$ we terminate the search and add $\hat{x}$ to the dPLRM and update the parameters. Otherwise we chose $\hat{x}$ as our new node on the graph and returns to step 1.

Our hope is that observations whose corresponding kernels are close in $\mathcal{H}$ also gives rise to gradients that are close. As our problem is a continuous one the main prerequisite for this idea to work is that the observations are sufficiently close.

It should be mentioned that a kNN-based graph is not necessarily the best option. In general we would like a sparse graph representation of the Gram matrix K so that every point x can be reached by a path from any point x'. It is not clear however how this graph should be traversed, as the approach outlined earlier would certainly be trapped in local minima very quickly.

Another open question is how one should rearrange the graph as observations are removed. In this work we simply fully connect the neighbors of the observations being removed to avoid a disconnected graph. This is illustrated in figure 1.

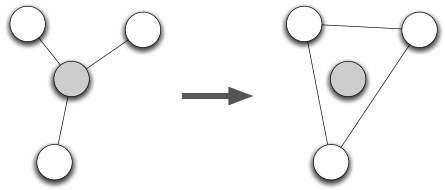

Figure 1: The neighbors of a observation that is removed from the graph is fully connected to avoid any breaking up of the graph.

4. Experiments

In this section we presents experimental results on a simple phonetic feature detection problem. Although this problem has some practical interest (see eg. [10]), our main purpose here is not to create a world class detector, but rather to illustrate the viability of the approach outlined in the previous sections.

We chose a subset of utterances from the TIMIT database [11], and re-labeled the phoneme segments according to table 1. We then extracted 25 millisecond (ms) segments every 10 ms and calculated the log spectrum of each segment. This yielded a 200 dimensional feature vector every 10 ms. The vectors were labeled according to which phoneme they belonged to.

Class	Phonemes
stop	b bcl d dcl g gcl p pcl t tcl k kcl q dx
fricative	jh ch s sh f z zh th v dh
nasal	m n ng en em eng nx
liquid	l r w y hh hv el
vowel	iy ih eh ey ae aa aw ay ah ao oy
	ow uh uw er axr ax-h ix ux ax
silence	h# pau epi

Table 1: Phonemes from the TIMIT database clustered according to broad class phonetic features.

For our experiment we used 53874 training examples and 19489 for testing. The examples where picked across all the speakers in the training and test subsets of TIMIT. We trained a baseline system using all the training examples and use this for comparison with the graph based algorithm

For the greedy training algorithm we added the training vectors one at the time. To add several kernels at each step would probably be a waste as the greedy algorithm most probably would chose a set of very similar kernels. We used a simple polynomial kernel of third order. Higher orders gave no significantly better results in our initial test. The graph search algorithm used a kNN with $k = 10$. We also made sure that the graph was fully connected. The results of our experiments are presented in figure 2

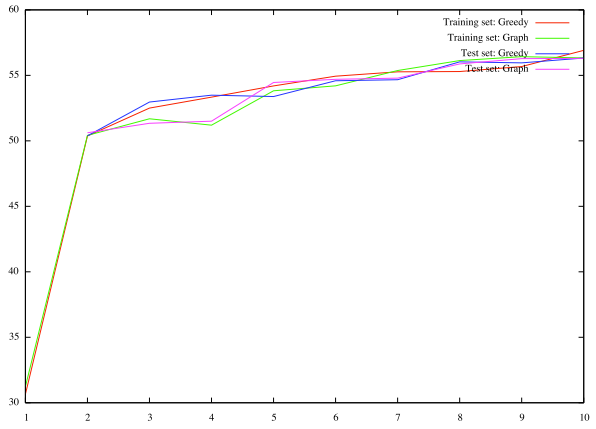

Figure 2: Comparison between searching the full training set and just a subset using a graph search algorithm for use with greedy training of a dPLRM classifier.

The results show that the graph selection algorithm yields a performance that is comparable to that of the greedy dPLRM formulation, while investigating only a small fraction of the available training data. This is very encouraging for the further development of this approach, as speech data usually consists of large amounts of data. The performance on the test data was almost the same as on the training data. This is of course not something one should expect in general, but in this case the simplicity of the classifier in terms of using only ten training vectors makes this result possible. Note that we are not even covering every speaker in the training set. No results could be presented on the regular dPLRM approach since the number of training vectors made this infeasible.

5. Conclusion

We have shown that using a graph search combined with a greedy learning algorithm for dPLRM training let us train a kernel classifier using a large number of training examples. We are now in a position to investigate different kernels, and combinations of those, in whatever setting we are interested in. It is not clear how large datasets we can handle with the approach outlined in this paper, but that is a topic of further research.

6. Acknowledgments

We want to thank prof. Kunio Tanabe of Waseda University, Tokyo, and prof. Kenji Fukumizu of the Institute of Statistical Mathematics, Tokyo, for valuable discussions and helpful input.

7. References

[1] H. Bourlard and M. Nelson, *Connectionist Speech Recognition, a Hybrid Approach*, Springer, 1993.

[2] K. Tanabe, "Penalized logistic regression machines: New methods for statistical prediction 1," *ISM Cooperative Research Report*, , no. 143, pp. 163–194, 2001.

[3] K. Tanabe, "Penalized logistic regression machines: New methods for statistical prediction 2," in *Proc. IBIS*, Tokyo, 2001, pp. 71–76.

[4] K. Tanabe, "Penalized logistic regression machines and related linear numerical algebra," *Kokyuroku*, , no. 1320, pp. 239–249, 2003.

[5] T. Matsui and K. Tanabe, "Probabilistic speaker identification with dual penalized logistic regression machine," in *Proc. ICSLP*, Jeju, S. Korea, October 2004, pp. 1797–1800.

[6] T. Matsui and K. Tanabe, "dplrm-based speaker identification with log power spectrum," in *Proc. Interspeech*, Lisboa, Portugal, Sep. 2005.

[7] David G. Luenberger, *Linear and Nonlinear Programming*, Addison-Wesley Publishing Company, 2 edition, 1989.

[8] R. Courant and D. Hilbert, *Methods of Mathematical Physics*, vol. 1, Wiley-Interscience, 1989.

[9] J. Lafferty, X. Zhu, and Y. Liu, "Kernel conditional random fields: Representation and clique selection," in *Proc. of ICML*, 2004.

[10] J. Li and C.-H. Lee, "On designing and evaluating speech event detectors," in *Proc. Interspeech*, Lisboa, Portugal, Sep. 2005.

[11] "Timit acoustic-phonetic continuous speech corpus," http://ldc.upenn.edu/.