A SURVEY OF LEARNING LINEAR THRESHOLD FUNCTIONS WITH NOISE

Sihan Liu, Zhichun Huang
University of Wisconsin-Madison
Madison, WI 53706, USA
{sliu556, zhuang294}@wisc.edu

1 INTRODUCTION

Linear Threshold Functions (LTFs) are a type of boolean functions $f(x) = sign(\langle w, x\rangle - \theta)$, parameterized by a weight vector $w \in \mathcal{R}^n$ and a threshold $\theta \in \mathcal{R}$. It represents a separation of the input space along the hyperplane defined by w, θ and have been extensively used in many classification tasks. Learning unknown LTFs has been one of the most studied area in machine learning since its birth, with the earliest known algorithm being the Perceptron Algorithm by Rosenblatt (1958).

A more practical variant of this problem is learning LTFs under some noise model, which is used to characterize uncertainties in the real world. Properties of specific noise models have significant implications on the learnability of the problem. On one hand, the simplest form of such noise model could be the Random Classification Noise (RCN), where in each example the label is flipped with probability $\eta < \frac{1}{2}$. On the other hand, if the labels are determined by some adversarial oracle, it becomes one of the most challenging noise model to learn from called adversarial noise whose computational intractability is known Guruswami & Raghavendra (2006).

Recently, Diakonikolas et al. (2019) shows that efficient weak LTF learner exists in the presence of Massart noise, a noise model that generalizes RCN and adversarial noise with noise rate at most $\eta < \frac{1}{2}$. In the following sections, we plan to give more backgrounds around the ideas in Diakonikolas et al. (2019) by examine existing approaches for learning LTFs under different margin and noise conditions.

2 PRELIMINARIES

2.1 MASSART NOISE

Let $c : R^n \to \{0, 1\}$ be the target hypothesis, D_x be an arbitrary distribution over R^n, and $0 < \eta < \frac{1}{2}$. A sample from a Massart noise model is defined as

$$EX^{Mas}(c, D_x, \eta) = \begin{cases} (x, c(x)) & \text{with probability } 1 - \eta(x) \\ (x, -c(x)) & \text{with probability } \eta(x) \end{cases}$$

where $\forall x \in R^n : \eta(x) \leq \eta$. An equivalent formulation is that the label is controlled by an adversary with probability η Sloan (1988). The definition of Massart noise lies between RCN and adversarial noise, and when $\forall x \in R^n : \eta(x) = \eta$ it is identical to RCN.

2.2 β-OUTLIER REMOVAL

When developing algorithms for learning LTFs, it is usually handy to preprocess the underlying distribution so that $\forall w \in R^n, \|w\|_2 = 1 : E_{x\sim D_x}[\langle w, x\rangle^2] = 1$. In other words, the expected squared distance between $x \sim D_x$ and any hyperplanes going through the origin is 1. This can be *approximately* accomplished by transforming the sampled set of inputs according to their eigenvectors and eigenvalues. Consider the matrix form X of the input set S, the matrix $A = E(xx^T)^{\frac{1}{2}} = (XX^T)^{\frac{1}{2}}$ can be obtained by diagonalizing the symmetric matrix XX^T. Then we can simply take $X = A^{-1}X$ to obtain the desired transformed input (we will refer to this as transforming to isotropic positions).

An issue with this approach, however, is that the eigenvalues along some principle directions of X can be skewed away from the actual variance in the presence of outliers. For example, it might be the case that some input points are extremely far away from the mean along some principle direction w^*, which causes almost every other points to have $\langle w^*, x\rangle^2 \approx 0$ after applying the transformation. Thus, it is important to detect such outliers and exclude them from the computation of A. A point x^* is defined as a β-outlier if

$$\exists w \in R^n, \|w\|_2 = 1 : \langle w, x^*\rangle^2 \geq \beta E_{x\sim D_x}[\langle w, x\rangle^2]$$

The primary criteria for removing outliers is that we want to remove a small portion of samples in S while the remaining set S' contains no β-outliers for $\beta = poly(n, \frac{1}{\epsilon})$. Blum et al. (1998) shows that, for $\beta = \mathcal{O}(n^7 b/\epsilon)$ (b is the bits of precision of the input space), there always exist such S' which is obtainable in polynomial time. The idea is to iteratively remove points outside a multitude $\gamma > \frac{\beta}{36n}$ of the inertial ellipsoid (the ellipsoid represented by the covariance) of the remain points until conditions of S' are satisfied. The algorithm is guaranteed to hault within $poly(n, b)$ iterations because $Vol(\{w \in R^n : E[\langle w, x\rangle^2] \leq 1] | x \in \cap_{i=0}^{j} S_i\})$ more than doubles at each iteration j. Dunagan & Vempala (2004) further proves that, for an arbitrary distribution, with sample complexity $m = \tilde{O}(\frac{n\gamma 2}{\delta^2})$, an ellipsoid T where

(i) $Pr_{x\sim D_x}[x \in (1+\delta)T] \geq 1 - \epsilon$
(ii) $(1+\delta)T$ contains no $(1+\delta)^{10}\gamma^2$ outliers

can be identified in $poly(n, b, \frac{1}{\epsilon}, \frac{1}{\delta})$ time with probability $1 - \delta$. Thus for any $\Gamma > 0$, we can efficiently obtain an ellipsoid that contains no Γ-outliers.

We refer the readers to Section 5 of Blum et al. (1998) and Theorem 3 of Dunagan & Vempala (2004) for further details.

3 Learning with Large margin assumption

The perceptron learning algorithm used in practice turns out to be very successful when there exists a non-trivial margin $\gamma > 0$ between points and the optimal classification hyperplane. The formal definition of margin is given below.

Definition 3.1 *Given a set of data points $S \subseteq R^n$, the margin over the samples are defined as $\rho = \max_{w:|w|_2=1} \min_{x\in S} |w \cdot x|$.*

We call S satisfies the large margin assumption if $\rho \geq \gamma$. In the following three sections, we will look into approaches to learn noisy LTFs under this assumption.

3.1 Noise free

When there is neither noise nor arbitrarily small margin, the perceptron algorithm returns a strong hypothesis within polynomial time [Minsky & Papert (1969)]. As this is not the main focus of the article, we will briefly summarize the result in the lemma below as it serves as the base case where the other more complicated scenarios is reduced to.

Lemma 3.2 *Given n labeled linearly separable points $(x_1, y_1) \cdots (x_n, y_n) \in \mathbb{R}^n \times \{-1, 1\}$, the classical perceptron algorithm returns a feasible point in at most $1/\rho^2$ iterations, where $\rho = \max_{w:|w|_2=1} \min_i |x_i \cdot w_i|$.*

3.2 Random Classification Noise

Just like many other applications where random classification noise is present, the strategy is to recast the sample-based noise-free algorithm into one under statistical query. Instead of computing the update based on a single example that may be polluted, we now use statistical query to estimate the mean vector over the mis-classified region, namely $E_{(x,y)\sim D^\eta}[x \cdot y | y(w \cdot x) < 0]$. The idea is used in Blum et al. (1998). However, as their primary focus is on the margin, we will postpone the formal statement to Section 4.2.

3.3 MASSART NOISE

The massart noise has been shown as the most challenging noise model to address. As stated in Diakonikolas et al. (2019), the updating rule for the perceptron learning algorithm can be viewed as performing gradient descent on a convex surrogate of the error function. However, under the massart noise model, it has been shown that the learning task is essentially non-convex as no convex surrogate can be used to achieve even a weak learner. As this result essentialy has a different flavor than designing learning algorithm for LTF, we will state the formal result here for reference but will not expand on the techniques used.

Theorem 3.3 *Consider the family of algorithms that produce a classifier* $\text{sign}(w \cdot x)$*, where* w^* *is the minimum of the function* $G(w) = E_{(x,y)\sim EX^{Mas}(c,D,\eta)}[\phi(y(w \cdot x))]$*. For any decreasing convex function* $\phi : \mathbb{R} \to \mathbb{R}$*, there exists a distribution D over* $\mathbb{B}^2 \times 1$ *with margin* $\leq \frac{\sqrt{3}-1}{4}$ *such that the classifier* $\text{sign}(w^* \cdot x)$ *mis-classifies at least* $\min(\frac{\eta}{8\gamma}, \frac{1}{2})$ *fraction of the points.*

Inspired by the work of Blum et al., the authors of Diakonikolas et al. (2019) realize even though a global convex proxy is not possible in this case, the space can be divided into region of "thick stripes" around the hyperplane, where a local convex proxy for the error function can be found on each of them. In particular, the local surrogate takes the form of $L(w) = E_{(x,y)\sim D}\text{LeakyRelu}_\lambda(-y(w \cdot x))$. Or equivalently, expressed in terms of the true error function as $L(w) = (\text{err}(w, x) - \lambda)|w \cdot x|$. Though the loss function does not behave well when there is a large variance in the quantity $|w \cdot x|$, the author manages to show the minimizer does a good job at least in classifying all points that have large value of $|w \cdot x|$. As a result, the algorithm can recursively divide the entire space into "thick stripes" and train a high accurate hypothesis on each of the region through the local convex surrogate. Informally, the algorithm can be viewed as a loop that runs in two phases while there is still a significant portion of unclassified area:

1. Obtain a minimizer of the convex surrogate function through iterative stochastic gradient update
2. Determine the local "stripe like" region around the current hypothesis hyperplane where the minimizer does well on. In particular, the region is parametrized by a single threshold T, namely $\{x : |w \cdot x| \geq T\}$.

The choice of the threshold T here is rather subtle as the algorithm needs to constantly balance two quantity: on one hand, it cannot filter out too many points as it will make the algorithm run in forever; on the other hand, it must make sure the mis-classified region is under control as the hypothesis will do increasingly poor as $|w \cdot x|$ gets smaller. As a result, the choice of the threshold of $w \cdot x$ becomes finding a feasible solution to the following inequalities

$$Pr_{(x,y)\sim D}[w \cdot x \neq y||w \cdot x| \geq T] \leq \eta + \epsilon \tag{1}$$

$$Pr_{(x,y)\sim D}[|w \cdot x| \geq T] \geq \gamma\epsilon \tag{2}$$

If we can indeed find such a T, we can then ensure each time the hypothesis created is both accurate enough and rules out a significant mass of samples. The author first gives proof of the feasibility of the task and then gives procedures to construct a valid solution.

The proof relies on two structural lemmas. The first one relates the mass of the large-margin area and the mis-classification region to the value of the proxy loss function. The second lemma gives an upper bound over the minimum value of the proxy function by plug in the optimal hyperplane w^* (not the minimizer of the proxy loss function but the underlying target threshold function). We state the formal versions of the two lemma for future reference

Lemma 3.4 *Given* η *as the upper bound of Massart Noise and* $\lambda = \eta + \epsilon$ *as the desired error rate for the learning algorithm, for any vector* w *such that* $L(w) < 0$*, the minimum* T *that satisfies* $Pr_{(x,y)\sim D}[h_w(x) \neq y|(w \cdot x) \geq T] \leq \lambda - \frac{|L(w)|}{2}$ *also satisfies* $Pr_{(x,y)\sim D}[|w \cdot x| > T] \geq \frac{|L(w)|}{2\lambda}$.

Lemma 3.5 *Given* η *as the upper bound of Massart Noise,* $\lambda = \eta + \epsilon$ *as the desired error rate for the learning algorithm and* γ *as the margin assumed, the minimum value of the proxy function is at least* $-\gamma(\lambda - \eta)$.

Combining the two lemmas, the feasibility then becomes evident.

The process of finding a feasible threshold exploits the Dvoretzky–Kiefer–Wolfowitz inequality, which bounds the Kolmogorov distance between the empirical distributions and true distribution. By requesting a sufficient number of samples, we can then just perform the search on the empirical distribution and the estimation will be good enough on the underlying real distribution as well.

4 LEARNING WITHOUT MARGIN ASSUMPTION

In most cases, we can not assume that the sample points satisfy the margin assumption. Without the margin assumption, adversarial samples can be constructed, leading to exponential convergence rate of the classic perceptron algorithm. For a specific construction, readers can refer to the result stated in Maass & Turán (1994) for more details. This section hence focus on a variety of techniques, such as pre-processing and boosting, that can be combined with the existing results for the "large margin" scenario to realize margin-independent polynomial learnability in the general case.

4.1 NOISE FREE

4.1.1 IMPROPER LEARNING

Blum et al. (1998) first presents a modified perceptron algorithm that runs in polynomial time independent of the sample set S. The algorithm has an extra parameter σ and halts when all $x \in S : h(x) \neq c(x)$ satisfies $|cos(w, x)| \leq \sigma$. At each iteration t, it either halts or updates w_t using $w_{t+1} \leftarrow w_t - (w_t \cdot \hat{x})\hat{x}$ for $x_t : cos(w, x_t) = max_{x \in S}|cos(w, x)|$ ($\hat{x}$ is an appropriate multiple of x^* that makes $w_{t+1} \perp x^*$). Since we always have $cos(w, x_t) > \sigma$ for each update, the algorithm converges in $O(\frac{1}{\sigma^2} \ln(n) \ln(\frac{1}{\delta}))$ steps and returns an algorithm that is consistent with $T = \{x \in R^n : cos(w, x) > \sigma\}$.

The modified Perception algorithm can be combined with with isotropic transformation and the outlier removal algorithm described in 2.2 to ensure that $cos(w, x) > \sigma$ with a high probability. After removing the β-outliers and transforming into isotropic positions, the resulting set S' satisfies $E_{x \in S'}[\langle w, x \rangle^2] = 1$ and $\max_{x \in S'} \langle w, x \rangle^2 \leq \beta E_{x \in S'}[\langle w, x \rangle^2] = \beta$ for any unit vector w. Thus, $E_{x \sim D_x}[cos(x, w)^2] \geq \frac{E_{x \sim D_x}[\langle w, x \rangle^2]}{max|x|^2} \geq \frac{1}{\beta n}$. Since $cos(x, w)^2 \in [0, 1]$, this implies that at least $\frac{1}{2\beta n}$ fraction of $x \in S'$ satisfies $cos(x, w)^2 \geq \frac{1}{2\beta n}$. Running the modified perceptron algorithm with $\sigma = \frac{1}{2\beta n}$ on S' therefore returns a hypothesis that is consistent with at least $\frac{1}{2\beta n}$ of D_x.

The algorithm above gives a weak learner that achieves $err(h) < \epsilon$ on $T \subset R^n$ with probability at least $1 - \delta$. To construct a strong PAC learner, we can iteratively restrict the distribution $D_x^{(t)}$ to the unknown region $\hat{S}_{t-1} = \{R^n \cap T_1^c \cap .. \cap T_{t-1}^c\}$ and apply the algorithm to obtain w_t and T_t. The final hypothesis is a decision list $\{(T_1, w_1), (T_2, w_2), ..., (T_t, w_t)\}$ with the rule "if $x \in T_i$ (in other words, $cos(w_i, x) > \sigma$), output $h(x) = \text{sign}(\langle w, x \rangle)$, otherwise repeat on (T_{i+1}, w_{i+1}). If none of the conditions satisfied, return randomly". Since $\forall t : Pr_{x \sim D_x^{(t)}}[x \in T_t] \geq \frac{1}{2\beta n}$, $Pr_{x \sim D_x}[x \in \hat{S}_t] \leq (1 - \frac{1}{2\beta n})^t$. Therefore, in the final hypothesis, the probability mass on the unknown region can be arbitrarily small ($= O(\delta)$), while the error on each known region T_i is small with high probability. Similar idea is also used in Diakonikolas et al. (2019) to construct a stronger hypothesis.

4.1.2 PROPER LEARNING

One may wonder whether proper learning is possible in the small margin noise-free case. The obvious approach is to model the learning task as a linear program. Then, standard polynomial algorithm like interior method or ellipsoid algorithm can be used to efficiently compute feasible solution. Dunagan & Vempala (2008) yet has proposed an alternative approach to solve linear programs of the form $Ax \geq 0, x \neq 0$ through a re-scaling procedure followed by a standard perceptron learning algorithm.

As mentioned in Lemma 3.2, the running time of the original algorithm crucially depend on the quantity $\rho(A) = \max_{x:|x|=1} \min_i \frac{1}{|a_i|}(a_i x)$, which the author denotes geometrically as the "radius" or "roundness" of the constrain matrix A in the corresponding linear program. The key intuition

behind the rescaling phase is then to iteratively increment ρ through linear transformation of the constrain matrix A.

Ideally, the transformation takes the form of $A(I + vx^*x^{*T})$, where $x^* = \text{argmax}_{x:|x|=1, Ax\geq 0} \min_i \frac{1}{|a_i|}(a_i x)$ and v is a scalar controlling the step size towards the ideal constrain matrix. By Setting $v \to \infty$, the result after the transformation will have its ρ value getting arbitrarily closed to 1. However, finding such a transformation is as hard as solving the original half-space learning algorithm as x^* itself also needs to satisfy the constrain $Ax \geq 0$. Thus, the author devises a routine, which they denote as the "Perceptron Improvement Phase", that aims at finding an approximate solution $\hat{x}$ which satisfies the following relaxed version of the constrains

1. $\hat{x} \cdot z \geq \frac{1}{\sqrt{n}}$, where $Az \geq 0$
2. $\min_i \frac{1}{|a_i|} a_i \hat{x} \geq \sigma$

The phase relies on random initialization and an iterative updating rule to obtain such $\hat{x}$. Through a simple probabilistic argument, it turns out for any vector $z \in \mathbb{R}^{\kappa}$, a random unit vector $\vec{x}_0$ actually satisfy the first property with constant probability. After that, much like the updating rule of the perceptron algorithm itself, the auxiliary routine iterative searches for a_i where the second property is violated and executes the update rule $x \leftarrow x - (\bar{a_i} \cdot x)\bar{a_i}$, where $\bar{a_i}$) is the normalized unit vector parallel to a_i. With some straightforward computation, the author shows the update rule will not influence the validity of the first property and will eventually halt at a $\hat{x}$ which satisfies the second property given the initialization x_0 is legit.

In Lemma 3.3, they argue that such approximation is good enough in a sense that the "radius" of the matrix transformed by $I + \hat{x}\hat{x}^T$ gets improved by at least a multiplicative factor of $1 + \frac{1}{3n}$ after the rescaling. Thus, they could just repeat the routine until the radius ρ becomes big enough. Then, they could just run the classic perceptron algorithm and the routine is guaranteed to terminate in $O(1/\rho^2)$ iterations.

Remark 4.1 *One may notice that the algorithm do not have a proper way for checking whether the estimation returned by the improvement phase indeed satisfies the first property as the algorithm does not have a valid z in hand. Indeed, it is possible the algorithm updates the constrain matrix A with an invalid $\hat{x}$, leading to decrease in ρ. Fortunately, the author manages to show that ρ will not decrease too much in this case. (In fact, it will not decrement more than a multiplicative factor of $1\frac{1}{32n} - \frac{1}{512n^2}$.) As a result, ρ will have an overall increasing trend since the increments outweight the downturn.*

4.2 Random Classification Noise

Similar to the large-margin case, the strategy is again to adapt the noise-free algorithms into their statistical query based version. It turns out improper learning is relatively easy in this case. As described in Blum et al. (1998), the outlier-removal routine is not affected by the classification noise at all since the computation does not involve any label but only deals with the distribution of input vectors. On the other hand, the modified perceptron algorithm does rely on the following two properties of the update in order to act properly

1. $\cos(w, x)l(x) \leq -\sigma/2$, where σ is the fixed threshold rule used
2. $\cos(w^*, x)l(x) \leq \frac{-\sigma^2}{16\sqrt{n \ln n}}$, where n is the dimension of the input point,

which is a relaxation of the original criteria

1. $\cos(w, x)l(x) \leq -\sigma$
2. $\cos(w^*, x)l(x) \leq 0$

Fortunately, statistical version of the update rule indeed satisfy the requirements. In the noise-free algorithm, the update rule explicitly searches for a mis-classified sample x which maximizes the angle $\cos(w, x)$, and then performs the update rule $w \leftarrow -(w \cdot x)x$. Alternatively, through statistical query, one can easily estimate the quantity $E_{(x,y)\sim D}[(w \cdot x)x|\cos(w, x)(w \cdot x) \leq -\sigma]$.

Since the conditional expectation itself satisfies the finer condition, we can make sure the result of statistical query will satisfy the relaxed version with high probability if we drive the estimation to high accuracy (polynomial in σ and n).

Thus, the entire algorithm can easily be adapted to work under the statistical query model, and hence under random classification noise.

4.3 MASSART NOISE

The algorithm combines the technique used in Sections 3.3 and 4.1.1. While algorithm in 3.3 ensures PAC-learnability in the presence of large margin, the outlier removal algorithm 4.1.1 can be used as a general tookit for restoring a "probabilistic" version of the margin assumption in the margin-free case. In particular, it makes sure the expectation of $(w \cdot x)^2$ over the distribution of x is lower bounded for any unit vector w. We will state the formal definitions of both the original margin and this probabilistic version for clarity

1. Original Margin $\max_{w:|w|_2=1} \min_{x\sim D}(w \cdot x)^2 \geq \gamma$
2. Probabilistic Margin $E_{x\sim D}[(w \cdot x)^2] \geq \Gamma$ for all unit vector w.

Remark 4.2 *There are many ways to interpret the second property. In Dunagan & Vempala (2004), the author interprets it as "no matter which hyperplane is chosen, a significant portion of points are always far from the hyperplane". While in Diakonikolas et al. (2019), it is alternatively interpreted as the "minimum eigenvalue of the covariance of the distribution". They are equivalent as the quantity can easily be rewritten as* $\hat{\rho} = \min_{w:|w|_2=1} E_{x\sim D}[\frac{1}{|x|_2^2}(w\cdot x)^2] \leq \min_{w:|w|_2=1} w^T E_{x\sim D}[xx^T]w$ *assuming the l2 norm of all points in the distribution is bounded by* 1. *The assumption on l2 norm can easily be realized through rescaling after removing all the outliers having large l2 value.*

The main lemmas the algorithm relies on are Lemmas 3.4 and 3.5. Lemma 3.4 is not really affected as it is not based on the margin assumption. Thus, the only remaining work is to adapt the proof of Lemma 3.5 so that it works even if we only have "probabilistic margin" over the distribution of sample points. The adaptation turns out to be seamless as the loss function is itself an expectation over the distribution.

Thus, combining the two techniques, the author obtain the first distributional-independent PAC learning algorithm for linear threshold functions under Massart noise model.

5 CONCLUSION

We have examined several existing approaches to learn LTFs under different noise models and margin conditions. Recent works show that even learning LTFs under Massart noise, which is considered one of the most challenging noise model to learn from, is efficiently feasible. A problem yet to be addressed is whether there exist efficient noise-tolerant proper learning algorithms for LTFs under Massart noise. Diakonikolas et al. (2019) suggested that proper learning might be achievable by improving upon Dunagan & Vempala (2008). Whether it is true remains an open question.

REFERENCES

Avrim Blum, Alan Frieze, Ravi Kannan, and Santosh Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1-2):35–52, 1998.

Ilias Diakonikolas, Themis Gouleakis, and Christos Tzamos. Distribution-independent PAC learning of halfspaces with massart noise. *CoRR*, abs/1906.10075, 2019. URL `http://arxiv.org/abs/1906.10075`.

John Dunagan and Santosh Vempala. Optimal outlier removal in high-dimensional spaces. *Journal of Computer and System Sciences*, 68(2):335 – 373, 2004. ISSN 0022-0000. doi: https://doi.org/10.1016/j.jcss.2003.07.013. URL `http://www.sciencedirect.com/science/article/pii/S0022000003001442`. Special Issue on STOC 2001.

John Dunagan and Santosh Vempala. Optimal outlier removal in high-dimensional spaces. *Journal of Computer and System Sciences*, 68(2):335–373, 2004.

John Dunagan and Santosh Vempala. A simple polynomial-time rescaling algorithm for solving linear programs. *Mathematical Programming*, 114(1):101–114, 2008.

V. Guruswami and P. Raghavendra. Hardness of learning halfspaces with noise. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pp. 543–552, 2006.

Wolfgang Maass and György Turán. How fast can a threshold gate learn? In *Proceedings of a workshop on Computational learning theory and natural learning systems (vol. 1): constraints and prospects: constraints and prospects*, pp. 381–414, 1994.

Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386-407, 1958.

Robert H. Sloan. Types of noise in data for concept learning. In *Proceedings of the First Annual Workshop on Computational Learning Theory, COLT '88*, pp. 91–96, 1988.