Automated Systems & Technologies
25-26 May 2015 • St. Petersburg, Russia

LEARNING FROM NOWHERE

Bruno Apolloni[*], Simone Bassis, Jacopo Rota,
Gian Luca Galliani, Matteo Gioia, Luca Ferrari

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
Via Comelico 39/41, 20135 Milano, Italy
`apolloni@di.unimi.it`

Abstract

We extend the FIS paradigm to the case where universe of the discourse is hidden to the learning algorithm. Hence the training set is constituted by a set of fuzzy attributes in whose correspondence some consequents are observed. The scenario is further complicated by the fact that the consequents are evaluated exactly in terms of the same fuzzy sets in a recursive way. The whole arose from everyday life problems faced by the European Project Social&Smart in the aim of optimally regulate household appliance runs. We afford it with a two phase procedure that is reminiscent of the Distal learning in neurocontrol. A web service is available where the reader my check the efficiency of the assessed procedure.

INTRODUCTION

Since their introduction by Zadeh [7], fuzzy sets have been intended as a rigorous way of dealing with non formalized knowledge falling in the sphere of the experience and intuition of the designers. Rather than *explaining why*, membership functions are a clear way of *describing the granularity* of the information owned by them. Fuzzy rule systems constitute a dynamical version of the fuzzy sets paradigm, hence a favorite tool for solving system control problems when fuzziness affects the description of their dynamics [3]. The general expression of this fuzzy rule system is the following one:

$$\begin{array}{rcl}
\textbf{if } x_1 \text{ is } A_{11} \text{ and} & \dots & \text{and } x_n \text{ is } A_{1n} \textbf{ then } y \text{ is } B_1 \\
\textbf{if } x_1 \text{ is } A_{21} \text{ and} & \dots & \text{and } x_n \text{ is } A_{2n} \textbf{ then } y \text{ is } B_2 \\
\vdots & \vdots & \vdots \\
\textbf{if } x_1 \text{ is } A_{k1} \text{ and} & \dots & \text{and } x_n \text{ is } A_{kn} \textbf{ then } y \text{ is } B_k
\end{array} \tag{1}$$

[*]This work has been supported by the European Project FP7 317947 Social&Smart

where each rule consists of an antecedent prefixed by **if** followed by a consequent prefixed by **then**; A_{ij} and B_i, for all $i = 1,\dots,k$ (k – the number of rules), $j = 1,\dots,n$ (n – the number of conditions), are fuzzy sets defined in the corresponding input and output spaces, while x_i and y are variables corresponding to respectively the i-th condition and conclusion.

Within the European project SandS (as a contraction of Social&Smart[1]) aimed at optimally and adaptively ruling in remote the microcontrollers of our household appliances, we faced peculiar fuzzy rules that concerning, for instance, a bread maker, sound as:

- **if** the loaf is less crusty and soggy **then** increase rising time
- **if** the loaf is very crusty and crisp **then** decrease rising time

These rules may be easily framed in the Sugeno reasoning model [4] where the consequents are crisp variables to be computed through a weighted mixture of functions depending directly on the input variables; in formula:

$$y = \sum_{i=1}^{k} \overline{w}_i f_i(x) = \frac{\sum_{i=1}^{k} w_i f_i(x)}{\sum_{i=1}^{k} w_i} \tag{2}$$

where f_i defines the activation function of the i-rule, whose shape and arguments depend on the chosen model, w_i denotes the satisfaction degree of its premises, and $x = (x_1,\dots,x_n)$.

Figure 1 gives a schematic idea of the system in relation to the two rules above. Their distinguishing features come from the fact that when the user says that “the loaf is very crusty”, s/he has no reference metric variable to set to a specific value, even though s/he distinguishes different kinds of crustiness that gathers in the same quantifier “very crusty”. This denotes that variables x_js exist; but they are hidden to the eahouker her/himself. Nevertheless, to identify a suitable value of y we must estimate their values – a problem not far from the one of identifying the state transition matrix in the Hidden Markov Models [5].

The second difficulty is connected to f_i in two respects. On the one hand we hazard functional forms such as linear and quadratic ones to interpret the consequent “increase/decrease” of the rising time. But this is a common plague of the Sugeno approach. On the other hand, we don’t know the relation between the operational parameter (the rising time) and its effect on the crustiness and humidity appreciation on the part of the taster. This entails a non trivial identification problem to be faced, in a way reminiscent of distal learning in neurocontrol [2] with a two-phases (identification and control) algorithm.

In this paper we afford this problem for the actual assessment of the SandS system on a bread maker. The paper is organized as follows. In the next section we formalize the training problem and specify the adopted options; then we discuss the numerical results, and finally draw some conclusions.

[1] `http://www.sands-project.eu`

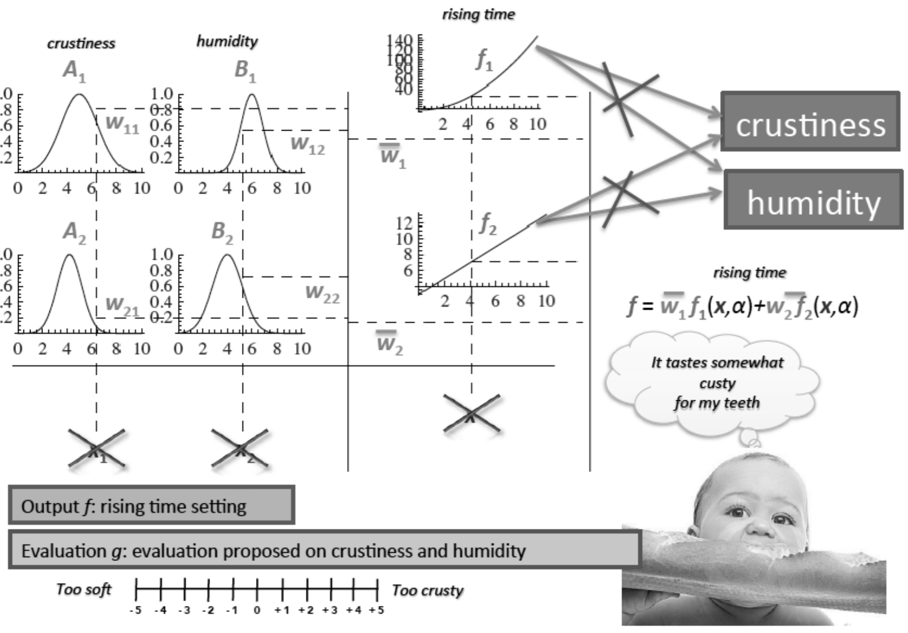


Figure 1: A learning-from-nowhere instance.

THE FORMAL FRAMEWORK

The proposed fuzzy rule system is characterized by:

- n_c crisp variables y_is, associated to the operational parameters;
- n_f fuzzy variables g_is, each described by r quantifiers, associated to the task execution evaluation;
- k rules, grouped in n_c clusters each containing n_f^r rules, where each cluster is responsible for the update of the single operational parameter, and each rule consists in principle of $n_f + 1$ antecedents (the n_f fuzzy variables plus the operational parameter the cluster refers to).

With reference to the standard ANFIS architecture [1]:

- **layer** 0, where we enter the $n = n_c + n_f$ crisp variables $x = (y, g)$, is complicated by the hidden features of those underlying the fuzzy sets (namely, the judgments g). This resolves in the addition of learned shift values to initial ones, where the latter may be either the nominal values of the fuzzy numbers used by the tester or completely random numbers (in a reasonable range) computed by the system;

- in the fuzzification **layer** 1, where the membership functions (m.f.s) are introduced, we focus only on triangular functions $\mu_{A_{\{a,b,c\}}}(x)$ and asymmetric Gaussian-like functions $\mu_{A_{\{v,\sigma_l,\sigma_r\}}}(x)$ which require only three parameters to be specified; namely:

$$\mu_{A_{\{a,b,c\}}}(x) = \begin{cases} \frac{x-a}{(b-a)} & \text{if } x \in [a,b] \\ \frac{c-x}{(c-b)} & \text{if } x \in (b,c] \\ 0 & \text{otherwise} \end{cases}; \quad \mu_{A_{\{v,\sigma_l,\sigma_r\}}}(x) = \begin{cases} \mathrm{e}^{-\frac{(x-v)^2}{\sigma_l^2}} & \text{if } x \leq v \\ \mathrm{e}^{-\frac{(x-v)^2}{\sigma_r^2}} & \text{if } x > v \end{cases} \tag{3}$$

- in **layer** 2, we compute the satisfaction degree w_j of the premise of the j-th rule (the conjunction of the antecedents) as the product of the membership degrees of the metric variables;

- in **layer** 3, the normalized satisfaction degree $\bar{w}$ is computed as usual by dividing a single degree by the sum of degrees in a same cluster of rules referring to the same consequent.

- in **layer** 4, the Sugeno functions are computed as a function of the layer 0 crisp variables; and

- in **layer** 5 their outputs are summed in each cluster with a weight equal to the normalized degree computed at layer 3.

We afford the entire training task in terms of a back-propagation algorithm, hence a long derivative chain as the following:

$$\frac{\partial E}{\partial \theta} = \frac{\partial E}{\partial g} \cdot \frac{\partial g}{\partial y} \cdot \frac{\partial y}{\partial \theta} \tag{4}$$

where the error E is the canonical mean square error between original signal τ (the target) and reconstructed signal y in the identification phase, whereas it is assumed to be the square of the task-execution evaluation – the judgment g – in the control phase (see Table 1). Note that, for the sake of clarity, we add a superscript to the involved variables to identify the timestamp the variable refers to only whenever necessary to solve ambiguities: so, for instance, both y and $y^{(t)}$ will refer to the signal y observed at time t. Moreover, in the identification phase g is a dummy variable, so that $\frac{\partial E}{\partial g} \cdot \frac{\partial g}{\partial y}$ contracts in $\frac{\partial E}{\partial y}$, with y as in (2), whereas in the control phase $\frac{\partial E}{\partial g}$ is g itself, modulo a constant. The last derivatives of y w.r.t. all the underlying parameters (identifying fuzzy sets and Sugeno functions) are computed as usual. Note that, since the fuzzy set supports are hidden, we add another parameter, namely the shifts to x values themselves, whose derivative computation is analogous to the one performed on fuzzy set parameters. In conclusion, our inference concerns the parameter vector θ, with

$$\theta = \left(m_{jk}, \sigma_{ljk}, \sigma_{rjk}, \delta_{ijk}, s_{ijk}\right) \tag{5}$$

Ident. phase	Control phase	
	Analytic	Numeric
$E = (y-\tau)^2$	$E = g^2$	
$\frac{\partial E}{\partial y} = 2(y-\tau)$	$\frac{\partial E}{\partial g} = 2g$	
	$\frac{\partial g}{\partial y} = \left(\frac{\partial y^{(t+1)}}{\partial g^{(t+1)}}\right)^{-1}$	$\approx \frac{g^{(t+1)}-g^{(t)}}{y^{(t+1)}-y^{(t)}}$
$\frac{\partial y}{\partial \theta} =$ canonical learning update rule		

Table 1: The analytics of the two-phase learning procedure.

with δ denoting the shifts we give to x, and s those to the Sugeno parameters, $\{i,j,k\}$ the indexes of a cluster, a rule inside the cluster and a fuzzy set inside the rule, respectively. In the identification phase their increments are ruled as follows:

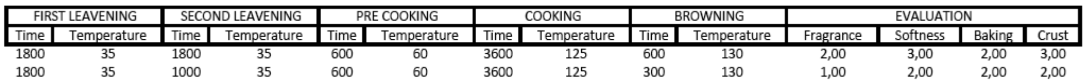
$$\Delta s_{ijk} = \eta(\tau_i - y_i)\overline{w}_j \cdot \varsigma_{ij}(x_k) \tag{6}$$

$$\Delta \nu_{jk} = \begin{cases} \eta \sum_i^n (\tau_i - y_i)\left(s_{ij} - y_i\right)\overline{w}_j\left(x_k - \nu_{jk}\right)/\sigma^2_{ljk} & \text{if } x_k \le \nu_{jk} \\ \\ \eta \sum_i^n (\tau_i - y_i)\left(s_{ij} - y_i\right)\overline{w}_j\left(x_k - \nu_{jk}\right)/\sigma^2_{rjk} & \text{if } x_k > \nu_{jk} \end{cases} \tag{7}$$

$$\Delta \sigma_{ljk} = \begin{cases} \eta \sum_i^n (\tau_i - y_i)\left(s_{ij} - y_i\right)\overline{w}_j\left(x_k - \nu_{jk}\right)^2/\sigma^2_{ljk} & \text{if } x_k \le \nu_{jk} \\ \\ 0 & \text{if } x_k > \nu_{jk} \end{cases} \tag{8}$$

$$\Delta \sigma_{rjk} = \begin{cases} 0 & \text{if } x_k \le \nu_{jk} \\ \\ \eta \sum_i^n (\tau_i - y_i)\left(s_{ij} - y_i\right)\overline{w}_j\left(x_k - \nu_{jk}\right)^2/\sigma^2_{rjk} & \text{if } x_k > \nu_{jk} \end{cases} \tag{9}$$

$$\Delta \delta_{ijk} = \begin{cases} \eta \sum_i^n (\tau_i - y_i)\left(\left(s_{ij} - y_i\right)\overline{w}_j\left(-\frac{x_k - \nu_{jk}}{\sigma^2_{ljk}}\right) + \overline{w}_j\frac{\partial s_{ij}}{\partial x}\right) & \text{if } x_k \le \nu_{jk} \\ \\ \eta \sum_i^n (\tau_i - y_i)\left(\left(s_{ij} - y_i\right)\overline{w}_j\left(-\frac{x_k - \nu_{jk}}{\sigma^2_{rjk}}\right) + \overline{w}_j\frac{\partial s_{ij}}{\partial x}\right) & \text{if } x_k > \nu_{jk} \end{cases} \tag{10}$$



FIRST LEAVENING		SECOND LEAVENING		PRE COOKING		COOKING		BROWNING		EVALUATION			
Time	Temperature	Time	Temperature	Time	Temperature	Time	Temperature	Time	Temperature	Fragrance	Softness	Baking	Crust
1800	35	1800	35	600	60	3600	125	600	130	2,00	3,00	2,00	3,00
1800	35	1000	35	600	60	3600	125	300	130	1,00	2,00	2,00	2,00

Figure 2: Parameters/evaluations pairs in the bread-making experiments.

Rather, in the control phase the error function reads:

$$E = \sum_{l}^{n_f} \left(g_i^{(t+1)} \right)^2 \tag{12}$$

where $g_i^{(t+1)}$ is the i-th evaluation provided by the user at time $t+1$. Here, the derivative $\frac{\partial g^{(t+1)}}{\partial y^{(t+1)}}$ is the most critical part of the chain rule (4), finding either: an analytical solution in the reciprocal of the derivative $\frac{\partial y^{(t+1)}}{\partial g^{(t)}}$, as it emerges from the identification of $y^{(t+1)}$, or a numerical one to bypass identification errors, giving rise, respectively, to:

$$\frac{\partial E}{\partial \theta} = \eta \sum_{i}^{n_c} \sum_{i'}^{n_f} 2 g_{i'}^{(t+1)} \left(\frac{\partial y_i^{(t+1)}}{\partial g_{i'}^{(t)}} \right)^{-1} \frac{\partial y_i^{(t+1)}}{\partial \theta} \tag{13}$$

$$\frac{\partial E}{\partial \theta} = \eta \sum_{i}^{n_c} \left(y_i^{(t+1)} - \tau_i \right)^2 \sum_{i'}^{n_f} 2 g_{i'}^{(t+1)} \frac{g_{i'}^{(t+1)} - g_{i'}^{(t)}}{y_i^{(t+1)} - y_i^{(t)}} \frac{\partial y_i^{(t+1)}}{\partial \theta} \tag{14}$$

The former indeed suffers from the typical plague of being identified in a domain that may be far from the one where y will be applied during the control phase – a drawback typically bypassed by alternating the two phases [6]. The second solution may deserve overfitting, and suffers from the cold start problem.

NUMERICAL RESULTS

Moving to our recipe generation task, we applied our two-phase procedure to the bread maker experiment dataset[2]. As it emerges from Figure 2, we have 10 parameters (time and temperature in the 5 baking phases) and 4 evaluations (fragrance, softness, baking, crustiness). As first training instance, we omitted training the FIS on parameter 6, because of its scarce variability, and 10, because temperature settings above 150 are dummy given a security threshold to 150 set in the microcontroller. Rather, the identification on the remainig parameters (after proper normalizaion) proved enough satisfactory (see Figure 3). However, many records in the training set come from a training phase on the part of the users as well, so that the suggestions on the new recipe on their part, which constitutes the FIS identification target, may prove definitely misleading.

[2]The database is available at `http://ns3366758.ip-37-187-78.eu/exp/excel`

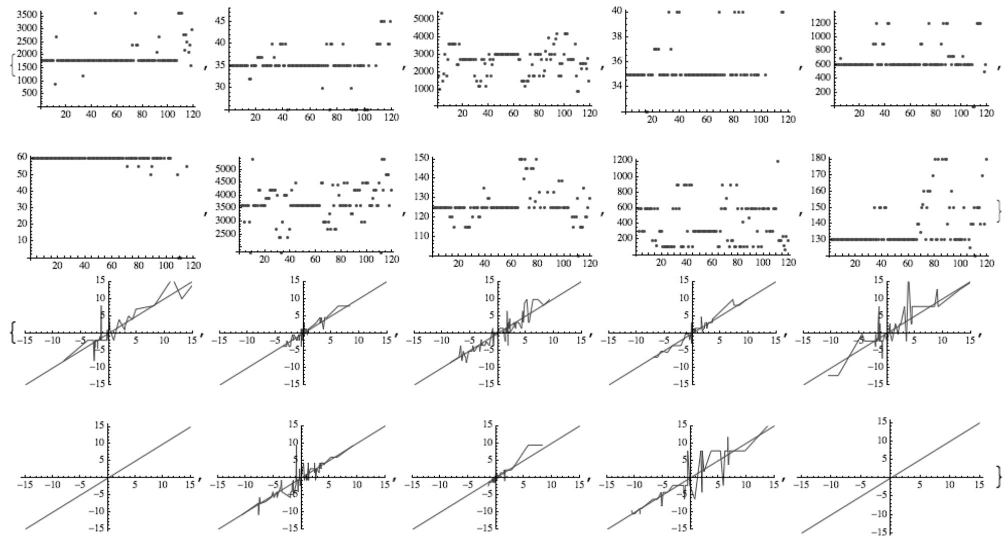


Figure 3: Plots of the tested parameter values and their recovering in the bread-making experiment.

Hence we decided doping the training set with a set of examples made up of the original instances as for the premises and optimal parameter settings (that we learned during the experimental campaign) as consequences. This expedient was profitable. On the one side the system learned well the new training set, as shown in Figure 4. Essentially the system is pushed toward the optimal parameters in the consequents, profiting of a well biased noise represented by the original examples. With this FIS configuration the system provides good suggestion on new inputs (hence in generalization). However, to render the system adaptive to the evaluation of the single users, we needed to train the FIS following the single user interactions, as done with the faces. Here the problem is a bit more tricky due to the mentioned slowness of the on-line process. Hence we devised the following expedient. After a good system identification (performed as before), we may assume that FIS computes the correct parameters in the doped part of the training set, so that the evaluation given in correspondence of these parameters properly applies to the computed parameters as well. Hence we may use this part of the training set to train the system on the evaluation as well (the control phase). Rather, we adopted an intermediate strategy by pairing the identification error with the control error through a mutual weighting based on a linearly convex composition. Namely, with reference to Table 1, the error becomes:

$$E = \xi \sum_{i}^{n_f} g_i^2 + (1-\xi) \sum_{i}^{n_i} (\tau_i - y_i)^2 \tag{15}$$

for proper (small enough) $\xi \in [0,1]$. Now that the FIS has been weaned, it is ready to work on-line. In this case, the error expression remains the same, where the second term acts as a regularization term, since the target τ now is the previous parameter value and

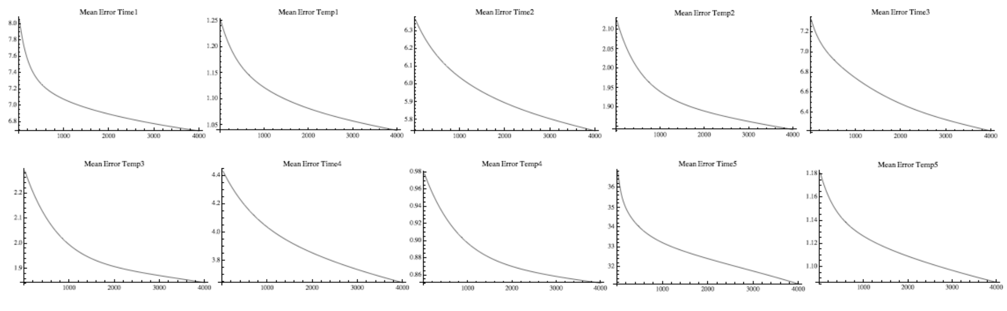
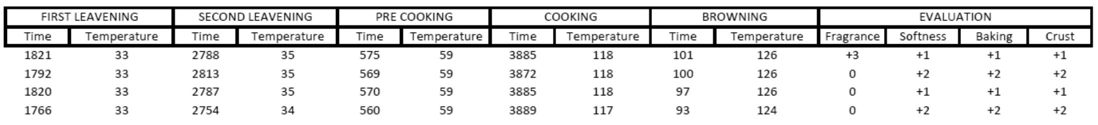


Figure 4: Error descent on the 10 parameters of the doped training set.

ξ is more or less big, depending on the evaluation values. The criterion is: if the overall evaluation is close to 0, then refrain FIS to produce new parameters that are far from the current ones. *Vice versa* do not hesitate moving far away from them if the current evaluation is poor (away from 0). Figure 5 reports a log of these on-line interactions leading to a gentle reduction of the leavening and cooking parameters in response to the user feeling, expressed in terms of slight defects of softness, baking, and crustiness.



FIRST LEAVENING		SECOND LEAVENING		PRE COOKING		COOKING		BROWNING		EVALUATION			
Time	Temperature	Time	Temperature	Time	Temperature	Time	Temperature	Time	Temperature	Fragrance	Softness	Baking	Crust
1821	33	2788	35	575	59	3885	118	101	126	+3	+1	+1	+1
1792	33	2813	35	569	59	3872	118	100	126	0	+2	+2	+2
1820	33	2787	35	570	59	3885	118	97	126	0	+1	+1	+1
1766	33	2754	34	560	59	3889	117	93	124	0	+2	+2	+2

Figure 5: Same as in Figure 2 when the FIS suggests recipes.

CONCLUSIONS

In this paper we introduce a new learning framework that we call *learning from nowhere*, which frames between the two cases investigated so far: the one where crisp variables are fuzzified to enter a fuzzy rule, and the other where fuzzy sets are dealt with as a whole within a fuzzy rule. The related learning procedure is a rather obvious extension of the common back-propagation procedure, inheriting both their robustness as for numerical results and their weakness as for theoretical results. As it is, we do not claim comparative efficiency w.r.t. other approaches to the problem. Rather we stress its suitability to solve in concrete a problem currently considered unfeasible by the white goods manufacturers: optimally conducting household appliances in a true fuzzy framework where the principal inputs come from the user feeling. Hence we consider all operational aspects of the related training problem, having as proof of evidence of its solution a record of people daily preparing his loft of bread according to the suggestions of the trained system. Future work will be devoted to generalize the framework to wide families of personal appliances and devise Key Performance Indicator (KPI) tools.

REFERENCES

[1] B. Apolloni, W. Pedrycz, S. Bassis, and D. Malchiodi. *The Puzzle of Granular Computing*, volume 138 of *Studies in Computational Intelligence*. Springer Publishing Company, Incorporated, 1 edition, 2008.

[2] M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16(3):307–354, 1992.

[3] S. Miu and Y. Hayashi. Neuro-fuzzy rule generatioion : Survey in soft. computing framework. *IEEE Transaction On Neural Networks,*, 11:748–768, 2000.

[4] D. Nauck, F. Klawonn, and R. Kruse. *Foundations of Neuro-Fuzzy Systems*. John Wiley & Sons, Inc., New York, NY, USA, 1997.

[5] L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE Acoutics, Speech and Signal Processing Magazine*, 3:4–16, 1986.

[6] Q.H. Wu, B.W. Hogg, and G.W. Irwin. A neural network regulator for turbogenerators. *IEEE Transactions on Neural Networks*, 3(1):95–100, Jan 1992.

[7] L.A. Zadeh. Fuzzy sets. *Information Control*, 8:338–353, 1965.