Learning a Fuzzy System from Training Data using the Münsteraner Optimisation System

Nicole Sprunk, Alejandro Mendoza Garcia, Alois Knoll
Technische Universität München,
Insititute of Robotics and Embedded Systems
Munich, Germany
Email: n.sprunk@tum.de

***Abstract*—For many classification or controlling problems a set of training data is available. To make best use of this training data it would be ideal to feed the data into a learning algorithm, which then outputs a finished, trained fuzzy controller, that is able to classify or control the original system. For the FUZZ-IEEE 2012 a competition was proposed to predict future volumes sold per day in a certain gas station. The training data includes a collection of gas prices at the current and the competitor's gas station and the according volume sold on every consecutive day in a period of about one year. This training data was analyzed and fit to a fuzzy learning algorithm based on the Münsteraner Optimisation System. As a base point a mean value comparison is used and then different features as fuzzy inputs are tested. Also different fuzzy set widths and and sequence of commands are compared. The final controller chosen shows promising results in the test with left out training data sets. Final results still have to be shown with the test data of the competition.**

I. Introduction

Fuzzy controllers or classificators are widely used. They are especially useful to fit data, which contains some uncertainities or inexactnesses, so to say fuzzy data. It is also useful to represent linguistic data. For example a linguitic rule could be: 'If the price is high and the competitors' prices are low, then the volume sold is low'. The advantage here is, that a person, even without knowledge about fuzzy theory can understand how such a fuzzy rule works. That is why our research group is working on a fuzzy controller to automate blood pressure regulation for intensive care patients [1]. Currently we are collecting data from the intensive care unit to analyze the medical doctor's behaviour in regulating the blood pressure manually by changing dosages of certain drugs. Our intention is to use this data then to further tune the fuzzy controller. For that reason we are very interested in this year's FUZZ-IEEE competition, where a similar task is to be performed.

II. Competition Training Data

The goal of this year's FUZZ-IEEE competition is to learn to predict gas volumes sold in a gas station for a day. As training data the competition comitee provided a collection of data for 25 different gas stations. The training data set for each of these gas stations include the month, weekday and the gas price of the regarded station as well as the competitor's gas stations. Furthermore the gas volume sold is provided for the training data. Additionally some test data sets are available, which do not include the volume sold on this specific date. Those will be used by the competition comitee to evaluate the prediction performance of the learning algorithm.

III. Methods

The goal is to be able to predict future volumes sold by learning from the provided training data. In the training data there are basically two different sources of information that influence the volume sold and can be used as inputs for the learning algorithm. On the one hand there is information on previous volumes sold depending on weekday and month. On the other hand there is the information on the gas price at the current station and all the competitors'. From this price information different characteristics can be calculated, for example price rank of own price compared to competitors' or difference between own price and mean price of all stations.

A. Mean Values of Previous Sold Volumes

The simplest method to predict a future volume is to take all previous volumes into account, calculate a mean value and use this as a prediction. The price set in the current station and the competitors' is disregarded in this case. Four different mean values were analyzed. First we tested the overall mean value, which is calculated over all training data sets of one station. Then, the mean value per day and the mean value per month were evaluated. Last, the mean value per 'day of a month' was also analyzed.

B. Fuzzy Learning System

The has been several research projects regarding the learning of a fuzzy system using training data. One well-known approach is the Adaptive Neuro-Fuzzy Inference System [2]. However the learning algorithm chosen is based on the approach of Münsteraner Fuzzy Optimisation System (MFOS) published by Steffen Niendieck [3]. It is also a Neuro-Fuzzy approach, where the fuzzy controller is interpreted as a neural network and trained accordingly. In this algorithm the pretuning of a fuzzy controller is done in several steps: creation, deletion and modification of rules and if needed sets are created, merged or modified. This approach was chosen as it is very comprehensive and was easy to reconstruct for us.

1) Rule Creation: As the competition application starts of with a controller, which has an empty rule set, all rules have to be created in the beginning. While more rules are created, more training data sets might fit the already created rules. To check if a new rule is needed, the degree of truth for each rule is calculated for each training data set. If none of the rules exceeds a certain truth threshold (0.5) a new rule will be created. In this case the input and output sets are analyzed and compared to the training data's input and output values. The sets with the maximum membership degree for these values are selected and used as input/output sets of the new rule. If for any set the maximum membership degree does not exceed the threshold (0.1), then a new set will be created for this input/output.

2) Set Creation: If requested in rule creation or modification, a new input/output set is created with the type and dimensions of the already existing set of this input/output and a mean value equal to the according value of the current training data set.

3) Rule Deletion: Through rule creation and modification it is possible that redundant rules arise, which are not needed anymore. To remove these rules three steps are executed.

- All equal rules (using same input and output sets) are deleted
- Search for maximum truth rules for a output set and remove rest: Check all training sets, which fit a certain output set, and save rule with maximum truth. If in all cases the same rule has the maximum truth, all other rules with this output can be deleted.
- Delete rules which always have minimum truth: Again all training sets, which fit a certain output set are checked and the rule with the minimum truth is determined in each case. If this was the same rule in all training sets then this rule is removed. This step is repeated until no minimum rule exists.

4) Rule Modification: For each rule the training data set is determined, which best fits the rule's input sets. Then the output of this rule is calculated and compared to the output of the training data set. If the difference between both exceeds a certain threshold (0.1), then a new output set needs to be assigned.

To find a suitable output set, the available output sets are analyzed and the one with the maximum membership degree is used. If the found output set does not exceed the error threshold (0.1) then a new output set will be created (see section III-B2 Set Creation)

5) Fuzzy Set Merging: The training data is analyzed for data sets, which differ only in one input set. All the other inputs match the same fuzzy sets. If also all outputs match the same output set, then the differing input sets can be merged.

To merge the two input sets, a trapzoid is used, the left values are used from the lower neighbour set, the right values from the upper neighbour set (see 1).

6) Fuzzy Set Removing: If an input or output set is not used in any rule, this set is deleted.

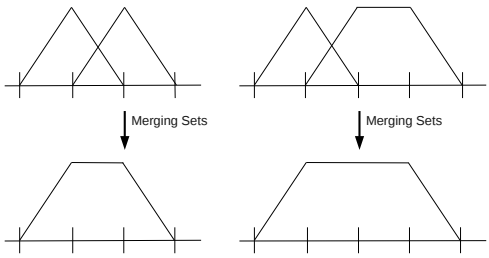

Fig. 1. Merging of Fuzzy Sets

7) Fuzzy Set Expanding: As the training data sets might not cover the whole area of the possible input space, we added a fuzzy set expanding method to close the gaps in between the created fuzzy sets. Each fuzzy set will be expanded to its neighbours, so the overlapping is 50%. In this application triangle fuzzy sets are used. That means that the left and right bottom x values of each set will be changed to the middle x value of the according neighbouring sets. If a set has no lower or upper neighbour, then the left or right x value will be set to the own middle x value (see figure 2).

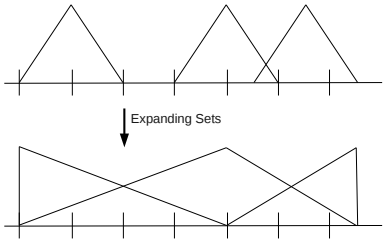

Fig. 2. Expanding of Fuzzy Sets

8) Fuzzy Set Adjusting: The adjusting of fuzzy sets is done exactly as the fine-tuning described in MFOS-M (MFOS for mamdani), with a gradient descent algorithm. It loops through all training data to shift input and outputs sets or change their widths, so that the controller is adapted further to the overall training data.

The sequence of all these steps influences the quality of the learning. So different sequences should be tested and compared.

IV. Test

To test the accurracy of the different fuzzy controllers some data sets were removed from the training data. To get a good overview on the performace the same trainings and tests were run with different subsets of the training data. Data sets were removed before training from either the beginning or end of the year (otherwise some algorithms depending on the previous day's volume would not work). Each time a different number of sets (0, 5, 10, 15 or 20) were removed. So in total the error was evaluated in 10 tests for each controller and then compared.

The error of one data set was calculated with the difference of the output of each controller to the volume defined in the training data (formula 1).

$$Error_i = \left| \frac{V_{iPredicted} - V_{iExpected}}{V_{iExpected}} \right| \quad (1)$$

To calculate the total error over all training data sets, all the errors of one station are summed up and divided by the number of data sets n of one station s (formula 2).

$$TotalError_s = \frac{\sum_{i=1}^{n} Error_i}{n} \quad (2)$$

A similar formula is used to calculate the error for the 'left out' test data sets, with the difference, that it is not calculated over n = number of training data sets, but over k = number of left out training data sets.

$$LeftOutError(k)_s = \frac{\sum_{i=1}^{k} Error_i + \sum_{i=1}^{k} Error_{n-i}}{2 * k} \quad (3)$$

This error is calculated only with the errors of the training data sets that were left out during the training period (k = 5, 10, 15 or 20 removed from beginning or end). So it is a good measurement of the accurrcy in predicting unknown data sets, which have not been trained.

After calculating all 'left out' errors of one station a mean value over all those values is determined to get the overall 'left out' error for this station.

$$LeftOutError_s = \frac{\sum_{i=5,10,15,20} LeftOutError(i)_s}{i} \quad (4)$$

Both the total error and the 'left out' error are summed up for all 25 stations and divided by the number of stations to compare test results.

$$TotalError = \frac{\sum_{i=1}^{25} TotalError_s}{25} \quad (5)$$

$$LeftOutError = \frac{\sum_{i=1}^{25} LeftOutError_s}{25} \quad (6)$$

V. Implementation

For each station a fuzzy controller is trained and tested. So each fuzzy learning algorithm has the training data of one station as input. At first we implemented a frame work to run through all test cases to be able to compare them to each other and choose the most suitable. After the mean values, features, initial set widths, etc. were chosen, the final fuzzy controller was implemented.

A. Comparison of Mean Values of Previous Date Information

Four different mean values calculations seemed promising to predict future sold volumes. The mean values were always calculated based on the training data of one station. First the mean value over all available sold volumes was calculated. Then the mean values per 'weekday' and per 'month' were evaluated and finally the mean value per 'weekday of month'. The results were compared by calculating the total and 'left out' errors as described in formulas 5 and 6.

TABLE I
Comparison of Mean Values

Mean Value	Total Error	Left Out Error
Station	15.5434	16.9009
Weekday	13.1982	15.2734
Month	14.0521	17.0122
Weekday in Month	10.6216	15.7045

The best results for the total error overall stations is achieved with the mean value per 'weekday in month', because this represents both seaonal trends and also weekday trends. However, when looking at the error of the test with the 'left out' sets, the mean value per 'weekday' shows better results. To predict unknown and untrained data the 'weekday' mean value might be sligthly better. The idea is to train the fuzzy controller to output the difference to one of these mean values. It is likely that either 'weekday in month' or 'weekday' will achieve the best results in this case too. The fuzzy learning will be tested and compared with all these mean values.

B. Comparison of Features for Fuzzy Learning System

The Comparison of features was done with several tests, comparing different mean values, different set widths and then run with the 10 described subsets of the training data (see section IV).

It was run with a sequence for pre-tuning of 'create rules - expand sets - delete rules - merge sets - expand sets - delete rules'. The runs were performed for all combinations of mean values and set widths, but only the results of the best mean value - set width combination are displayed in tables III and IV.

Following features were tested in different combinations:

- Weekday - Current weekday of data set
- Month - Current month of data set
- Price Rank: 1 defining the most expensive price in all competing gas stations, 2 defining the second most expensive and so on
- Difference from mean price: calculated by mean price on this particular day of all gas stations minus own price
- Last Volume: gas volume sold on the previous day
- Change in rank: difference between rank on the previous day and current day
- Change in mean price difference: difference between the difference from mean price on the previous day and current day

Different combinations of these features were used as inputs for the learning algorithm and the resulting fuzzy controller. As output the volume of gas sold on this particular day given in the training data is provided.

TABLE II
MEAN VALUE AS BASE VALUES FOR THE FUZZY CONTROLLER

Identification	Mean Value per
1	Station
2	Weekday
3	Month
4	Weekday in Month
0	NONE (absolute volume used)

As already the prediction just by the mean value (see section V-A) looked promising, these mean values will be used as base level for the predicted output. The fuzzy controller will then be trained to output the difference from the according mean value. That means that the predicted volume sold has to be calculated by adding the used mean value and the output of the fuzzy controller. The tests are run with the absolute volume and the difference to the four evaluated mean values. The identification for the mean values used in the following test are shown in table II. Furthermore the tests were run with different input/output set widths to determine, which is the most suitable.

Table III shows that the total error is minimal for the features:

– 'day-month-rank'

As there are only about four same weekdays in a month, and most of the time more than five possible price rank, this feature combination is very specialized to the training data and will not be very flexible classifying new data, because there will be a lot of rules missing. That is to say, the resulting fuzzy controller will be overadapted to the training data. It will perform well in known situations, but bad in unknown. For this reason the result for the same features in the 'left out error' table is worse.

The 'left out' error is minimal for the features 'day - last volume sold - change in mean price difference - change in rank'. This might still lead to an overadapted fuzzy controller. It is possible, that also the feature combinations with slightly worse results in 'left out error', which were achieved for

– 'day - last volume sold - change in mean price difference'
– 'day - last volume sold - change in rank'
– 'last volume sold - change in mean price difference - change in rank'

might be a reasonable choice for unknown future data. With one input variable less, the fuzzy controller will be less specific and dependant on the training data. This way an overadaption is not as likley.

C. Comparison of Initial Set Sizes for Fuzzy Learning System

The fuzzy controller will create new sets in the learning phase. The mean value of the newly created set will always be defined by the according value in the training data set. The width on the other hand is defined by an initial set created in the beginning. The width of this set will thus influence all succeeding sets and the number of sets needed to match the whole value area. Especially for the volume this set size may be important, as this variable has the highest variablity and is used both as input and output. That is why different set width were tested and compared for the volume: 250, 500, 1000, 2000, 3000 and 4000. If a mean value is used, the difference to the current mean value is represented instead of the absolute

TABLE III
COMPARISON OF TOTAL ERROR

Day	Month	Rank	Diff Mean Price	Last Volume	Change Mean Price	Change Rank	Best With Mean Value	Best With Set Width	Total Error
x	x						0	3000	12.3518
x	x	x					3	400	9.9829
x	x			x			4	400	11.0187
		x	x				4	200	11.6290
x				x			4	400	11.6854
x				x		x	4	400	11.1962
x				x	x		4	400	11.1436
x				x	x	x	4	400	10.1827
x					x		4	400	11.0481
x						x	4	200	11.9379
x					x	x	4	400	10.5074
					x	x	4	400	11.5272
				x	x	x	4	400	10.6567
x		x					4	200	11.7167
x			x				4	300	11.4378

TABLE IV
COMPARISON OF FEATURES - LEFT OUT ERROR

Day	Month	Rank	Diff Mean Price	Last Volume	Change Mean Price	Change Rank	Best With Mean Value	Best With Set Width	Left Out Error
x	x						2	400	17.3541
x	x	x					2	400	14.4855
x	x			x			2	400	13.8389
		x	x				2	400	14.9650
x				x			2	300	14.5003
x				x		x	2	300	13.2419
x				x	x		2	200	13.4255
x				x	x	x	2	400	12.5781
x					x		4	300	15.3895
x						x	4	200	16.5261
x					x	x	2	400	14.5567
					x	x	4	300	16.0825
				x	x	x	2	400	13.3559
x		x					2	300	15.7242
x			x				2	400	14.8368

volume. Then the widths have to be smaller: 25, 50, 100, 200, 300 and 400. Again the error was calculated as an average over various runs with 10 subsets, different mean values and different features.

As can be seen in tables III and IV, most of the time the results for the output of the absolute volume (mean value 0) only gives the best results in one test case. Most of the time both the 'total error' and the left error achieve the minimal value with the 400 width set for the volume difference from the mean value.

D. Comparison of Mean Value as Base for Fuzzy Learning System

After the test with only the mean values (see section V-A) an assumption was made, that the mean value per 'weekday' or the mean value per 'weekday in month' might be most useful as base for the fuzzy controller. Basing the output of the controller on a difference to those mean values takes care of weekday trends or additionally seaonal trends.
The results of the different test as shown in tables III and IV show that this is true. Most of the time the mean value per 'weekday in month' (4) achieves best results for the 'total' error, whereas for the untrained data evaluated with the 'left out' error the mean value per 'weekday'(2) gives better results.

Depending on the choice of features it will probably be better to choose the mean value per 'weekday'(2) as base, as the interesting task is to analyse untrained data in the end.

E. Comparison of Pretuning Sequences for Fuzzy Learning System

In the MFOS algorithm the sequence of how the different commands are executed influence the performance of the resulting fuzzy controller, both in accuarrcy and performance speed. Three sequences in total were evaluated regarding the accuracy:

- 1: 'create rules - expand sets - delete rules - merge sets - expand sets - delete rules' (as tested before in the previous sections)
- 2: 'create rules - remove sets - expand sets - delete rules - modify rules - merge sets - remove sets - expand sets - delete rules'
- 3: 'create rules - remove sets - delete rules - modify rules - merge sets - create rules - remove sets - expand sets - delete rules'

The highest accuracy is achieved with the longest pre-tuning sequence. This will make the learning algorithm run slower, as there are more steps to be executed. But still this choice is the best for accuracy reasons. Only for the 'rank-mean price difference' feature combination the first sequence is more reasonable.

F. Comparison of Controller Combinations

Out of the best controller, 5 were chosen to evaluate the final value. The controllers were trained and executed separetly but then a mean value of the outputs was evaluated. Also a sequence was considered, first trying the best adapted controller, which might be too adapted and have many missing rules. If it does not have any rules for this specific data set, then the output of the next controller is evaluated and so on.

The five controllers evaluated for this were the once with following specifications:

- Controller 1:
  - Features: Rank - Mean Value Difference
  - Set Width: 400
  - Mean Value: 2

TABLE V
COMPARISON OF PRETUNING SEQUENCES - SEQUENCE 1

Day	Rank	Diff Mean Price	Last Volume	Change Mean Price	Change Rank	Best with Mean Value	Best with Set Width	Total Error	Left Out Error
x			x	x	x	2	400	10.98	12.57
x			x		x	2	300	12.82	13.24
x			x	x		2	200	12.78	13.42
			x	x	x	2	400	12.07	13.36
	x	x				2	400	13.65	14.96

TABLE VI
COMPARISON OF PRETUNING SEQUENCES - SEQUENCE 2

Day	Rank	Diff Mean Price	Last Volume	Change Mean Price	Change Rank	Best with Mean Value	Best with Set Width	Total Error	Left Out Error
x			x	x	x	2	400	10.60	12.58
x			x		x	2	200	11.79	13.68
x			x	x		2	400	11.21	11.88
			x	x	x	2	400	11.98	13.71
	x	x				2	200	15.47	17.21

TABLE VII
COMPARISON OF PRETUNING SEQUENCES - SEQUENCE 3

Day	Rank	Diff Mean Price	Last Volume	Change Mean Price	Change Rank	Best with Mean Value	Best with Set Width	Total Error	Left Out Error
x			x	x	x	2	400	8.41	11.06
x			x		x	2	400	10.95	12.16
x			x	x		2	300	10.70	11.94
			x	x	x	2	400	10.42	12.41
	x	x				2	200	13.55	15.16

- Sequence: 3

– Controller 2:
  - Features: Day - Last Volume - Change in Rank - Change in Mean Value Difference
  - Set Width: 400
  - Mean Value: 2
  - Sequence: 3
– Controller 3:
  - Features: Day - Last Volume - Change in Mean Value Difference
  - Set Width: 200
  - Mean Value: 2
  - Sequence: 3
– Controller 4:
  - Features: Day - Last Volume - Change in Rank
  - Set Width: 300
  - Mean Value: 2
  - Sequence: 2
– Controller 5:
  - Features: Last Volume - Change in Rank - Change in Mean Value Difference
  - Set Width: 400
  - Mean Value: 2
  - Sequence: 3

Furthermore the mean value per 'weekday', and per 'weekday of month' were considered to be part of the mean value calculation or as a last resort for the sequence consideration.

This test was run with the whole original data as training input.

TABLE VIII
COMPARISON OF CONTROLLER COMBINATIONS: MEAN VALUES

	Total Error
All mean values	9.88
All mean values not equal 0	9.78
Mean per Day + Controller 5	11.25
Mean per Day + Controller 4	11.71
Mean per Day + Controller 3	12.03
Mean per Day + Controller 1	12.76
Mean per Day + Controller 2	9.90
Mean per Day of Month + Controller 5	9.70
Mean per Day of Month + Controller 4	10.12
Mean per Day of Month + Controller 3	10.55
Mean per Day of Month + Controller 1	11.20
Mean per Day of Month + Controller 2	8.51

The results with the combination mean value per 'Day of Month' is generally better. The best result was achieved with Mean Day of Month and controller 2, which has the features 'Day - Last Volume - Change in Rank - Change in Mean Value Difference'.

G. Final Implementation

After extensive test we chose to use a combination of controllers, combined with mean values and a defined sequence should a controller not have an output for a specific unknown data set.

The sequence we consider is similar to the results of the previous table VIII. If the best adapted controller with the smallest error in the test runs has a output, then this value is used as prediction. If it does not have an output, then the next controller in the sequence list is tested. In this case, if the output of controller 2 is defined then we use this 'Mean Day of Month + 2' combination, next 'Mean Day of Month + 5', and so on.

Th full sequence we consider is this:

– 1. Mean per Day of Month + Controller 2
– 2. Mean per Day of Month + Controller 5
– 3. Mean per Day of Month + Controller 4
– 4. All mean not equal 0
– 5. Mean Day of Month

VI. RESULTS

With the final implementation we get a total accuracy of 8.47 for the trained data. The accuracy in the actual test data can only be tested at the competition later on, but the results promise to be able to also analyse future data as several test with left out test data sets have shown.

For each of the gas stations three fuzzy controller are trained one for each of:

– Mean per Day of Month + Controller 2
– Mean per Day of Month + Controller 4
– Mean per Day of Month + Controller 5

The number of rules vary in the three controllers and from station to station from 106 to 448. Generally the controller 2 has the highest number of rules, as it is the most adapted. Hence it also has the most missing rules for new data sets, which don't fit into the highly adapted training space. The controller with the fewest rules is mostly the controller 5. It has less inputs and also does not use the day as input. This makes the resulting controller more general and not so well adapted to the training data, which is highly dependant on weekday information. On the other hand this controller will more likely have an output even to unknown future data, even if the controller 2 is not applicable.

VII. USAGE

The resulting program is called priceSalePredictions. It can be called in learning or predicting mode by using according switches:

```
priceSalePredictions -h -l -i infile.csv
-o outfile.csv
```

– h : displays a help
– l : triggers a new learning process with the provided training data
– i : defines the input file, where the training and test data is stored
– o : defines the output file, where the predictions for the test data will be saved

The format of the input file is a comma separated file in the format provided by the competition chair:

```
Site ID, Date Index, Weekday, Month, Vol,
ncomp, Price 1, Price 2, Price 3,
Price 4, Price 5, Price 6, Price 7,
Price 8, Price 9, Price 10, Price 11, Note
```

The output file is written in the same format, but ommiting everything after 'Vol'. The predicted volume will be written in this 'Vol' column. Only test data will be written to the output file, i.e. data, where the value of the 'Vol' column in the input file was equal to 0.

VIII. Conclusion

A simple learning algorithm for a fuzzy controller based on the MFOS approach was evaluated. The performance of different controllers was compared regarding initial mean value, different features as input for the fuzzy controllers, different initial set widths, different pre-tuning sequences and different combinations of controllers. The tests were evaluated with 10 subsets from the original training data. It still has to be shown, if the algorithm is also working reasonable for the test data in the competition. We look forward to compare with the results of other competitors.

Acknowledgment

This research was funded by the Graduate School of Information Science in Health (GSISH) and the Technical University of Munich (TUM) Graduate School

References

[1] N. Sprunk, A. Mendoza, A. Knoll, U. Schreiber, S. Eichhorn, J. Hörer, and R. Bauernschmitt, "Hemodynamic regulation using fuzzy logic," in *FSKD*. IEEE, 2011, pp. 515–519.

[2] J.-S. Jang, "Anfis: adaptive-network-based fuzzy inference system," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 23, no. 3, pp. 665 –685, may/jun 1993.

[3] S. Niendieck, "Optimierung von fuzzy-controllern - von untersuchungen hybrider neuro-fuzzy-systeme zum entwurf des universellen konnektionistischen modells mfos: Münsteraner-fuzzy-optimierungs-system," Ph.D. dissertation, 2004.