Detecting and Reacting on Drifts and Shifts in On-Line Data Streams with Evolving Fuzzy Systems *

Edwin Lughofer[1] Plamen Angelov[2]

1. Department of Knowledge-Based Mathematical Systems, Johannes Kepler University
Altenbergerstrasse 69, A-4040 Linz, Austria
2. Department of Communication Systems, Infolab 21, Lancaster University
Lancaster, LA1 4WA, UK
Email: edwin.lughofer@jku.at, p.angelov@lancaster.ac.uk

Abstract— *In this paper, we present new approaches to handle* drift *and* shift *in on-line data streams using evolving fuzzy systems (EFS), which are characterized by the fact that their structure is not fixed and not pre-determined. When dealing with* drifts *and* shifts *in data streams one needs to take into account two major issues: a) automatic detection of, and b) automatic reaction to this. To address the first problem we propose an approach based on the concepts of* age *and* utility *of fuzzy rules/clusters. The second problem itself is composed of two sub-problems concerning the influence of the* drifts *and* shifts *on: 1) the antecedent parts (fuzzy set and rule structure) and 2) the consequent parts (parameters) of the fuzzy models. To address the latter sub-problem we propose an approach that introduces a gradual forgetting strategy in the local learning process. To address the former sub-problem we introduce two alternative methods: one that is based on the evolving density-based clustering, eClustering (used in* eTS*); and one that is based on the automatic adaptation of the learning rate of the evolving vector quantization approach (eVQ) (used in* FLEXFIS*). The paper is concluded with an empirical evaluation of the impact of the proposed approaches in (on-line) real-world data sets where* drifts *and* shifts *occur.*

Keywords— drifts and shifts in data streams, evolving fuzzy systems, detection and reaction to drifts and shifts, age of a cluster/fuzzy rule, gradual forgetting

1 Introduction

1.1 Motivation and State of the Art

Nowadays data-driven fuzzy systems enjoy a great attraction in many industrial applications, as opposed to expert-based fuzzy systems. They can be automatically generated from process data such as measurements, images (features) or signal streams. Furthermore, they are proven universal approximators [27], i.e. being able to model a given problem to (theoretically) any degree of accuracy. Moreover, they also allow an insight in the form of linguistically and visually interpretable rules to be gained [8].

During the last decade, the research field of 'evolving fuzzy systems' (EFS) emerged as an important part of the fuzzy systems research [4] [11], as they are capable to include new information on demand into the models and on-the-fly without necessarily using prior knowledge. EFS can also permanently learn from their environment and are applicable in fast on-line identification [4] and modelling processes as well as huge data bases which cannot be loaded into the virtual memory at once [18]. Often, there are no enough data in advance (off-line) to build reliable models with high predictive quality which can also require application of EFS as in [23].

Various approaches for EFS have been set up during the last years, one of the most popular and pioneering approach is the *eTS* family which comes with a regression [4], [3] and a classification variant [5]. Another approach for adaptation and evolution of clusters inspired by the evolving vector quantization (eVQ) [19] is the so-called *FLEXFIS* family [21], in particular *FLEXFIS* for regression [20] and for classification [22] (denoted as *FLEXFIS-Class*). A range of other alternative approaches includes ePL [16], SAFIS [12], and evolving fuzzy neural networks such as SOFNN [15] or GDFNN [28].

All these methods have a common denominator: they all are life-long learning approaches, which means that they incorporate all the data samples into the fuzzy models with equal weights in the same order as they are coming in during the on-line process. Hence, fuzzy models reflect a compact information of all the samples seen so far with equal importance. This is a beneficial to adapt the models, especially when a convergence to an optimality criterion or stable state of the model structure is achievable [20]. However, this benefit is only true in case of data streams which are generated from the same underlying data distribution respectively which do not show any *drift* or *shift* to other parts of the input/output space [26]. *Drift* (respectively *shift*) indicate the necessity of out-dating of previously learned relationships (in terms of structure and parameters) as these are not valid any longer and hence should be eliminated from the model. In order to cope with this problem, *drift* handling was already applied in other machine learning techniques, e.g. in connection with SVMs [14], ensemble classifiers [24] or instance-based (lazy) learning approaches [7] but to the best of our knowledge this concept was not yet applied to fuzzy systems.

*This work was partially supported by the Upper Austrian Technology and Research Promotion and by The Royal Academy, UK. This publication reflects only the authors' views.

ISBN: 978-989-95079-6-8

1.2 Our Approach

Hence, in this paper we are dealing with approaches for an appropriate handling of *drifts* and *shifts* by EFS. However, parts of the concepts, especially the detection of, and reaction to *drifts* in the consequent parts of the rules, can be applied to a wider range of EFS approaches including *eTS* and *FLEXFIS*. A more detailed description will be given in Section 2. For achieving an automatic approach, we propose 1.) the detection of a *drift* respectively *shift* based on the concepts of fuzzy rule *age* and *utility* function, and 2.) the reaction to a detected *drift* respectively *shift*, both in rule antecedent (for *eClustering* as applied in *eTS* and *eVQ* as applied in *FLEXFIS*) as well as consequent parts of Takagi-Sugeno fuzzy models [25].

The paper is concluded with an empirical evaluation of the impact of the proposed approaches on predictive accuracy of the evolving fuzzy models when applying to real-world data streams where *drifts* and *shifts* occur (Section 5). Note, that no benchmark data from the Internet or well-known data bases can be applied for empirical evaluation as these are usually all smooth in the sense that no *drift* occurs therein.

2 Problem Statement

In machine learning literature, they distinguish between different types of 'concept change' of the underlying distribution of (on-line) data streams: a) *drifts*, and b) *shifts*, see [26]. *Drift* refers to a *gradual* evolution of the concept over time. The concept *drift* concerns the way the data distribution slides smoothly through the data/feature space from one region to another. For instance, one may consider a data cluster moving from one position to another. This concept is closely related to the time-space representation of the data streams. While the concept of (data) density is represented in the data space domain, *drift* and *shift* are concepts in the joint data-time space domain.

An (artificial) example of a *drift* is demonstrated in Figure 1. There, the original data distribution (in a 2-D data space) is marked by circular samples, which changes over time into a data distribution marked by rectangular samples. If a conventional clustering process is applied by weighting all new incoming samples equally, the cluster center would end up exactly in the middle of the whole data bunch, averaging old and new data. Such a clustering approach is applied in EFS approaches to identify the local regions that can be used to form (the antecedent parts of the) fuzzy rules.

On the other hand, the concept *shift* refers to a sudden, abrupt change of the underlying concept to be learned. A *shift* in the input space opens up a data cloud in an unexplored region and hence usually automatically causes a new rule to be evolved by the incremental/evolving clustering algorithm. In Figure 2 a case of a *shift* in the output variable is shown: the original trajectory (consisting of dense data samples) on the right-hand side is marked with a light line, whereas the *shift* is represented by the dark data samples forming a trajectory above the other one. In case if only apply the usual adaptation

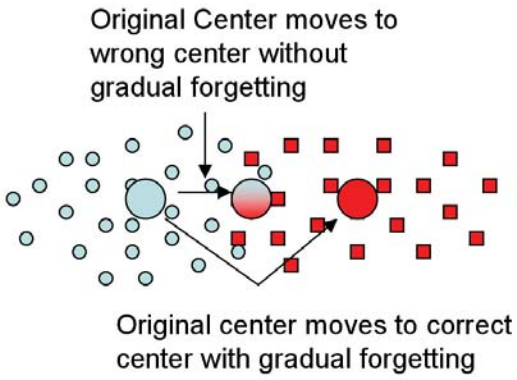

Figure 1: A *drift* in an evolving cluster (used for learning the antecedent parts of EFS), the distribution before the drift shown in circular, after the drift shown in rectangular samples.

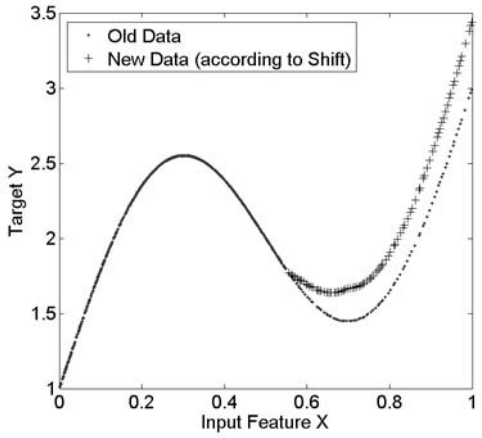

Figure 2: Example of a *shift* in the output variable; compare light dots (original data distribution) with dark dots (data distribution after the *shift*) on the right-hand side of the image (also marked by arrows)

of the consequent parameters with weighted recursive least squares without forgetting the old data due to the *shift* the approximation curve of the fuzzy model will end up exactly in-between these two.

In this paper, we demonstrate novel approaches for autonomous *drift* and *shift* detection and handling when learning fuzzy rule-based systems of Takagi-Sugeno type from on-line data streams in an evolving manner [4]. We focus on EFS approaches exploiting the Takagi-Sugeno model architecture [25].

3 Autonomous Detection of Drifts and Shifts in Data Streams by EFS

In this section the method for autonomous detection of *shifts* and *drifts* in data streams based on the *age* and *utility* of the cluster/fuzzy rule [1], [10] is described. This is an important step in the process of handling non-stationarity in data streams and for building au-

ISBN: 978-989-95079-6-8

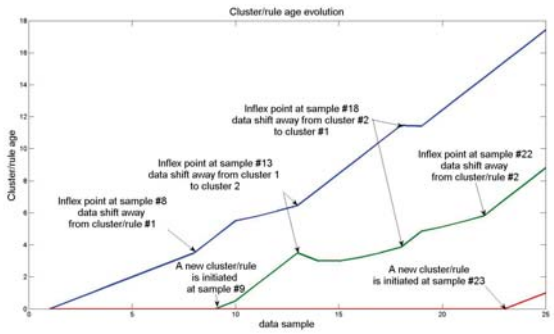

Figure 3: Evolution of the *age* of the clusters/fuzzy rules; *drifts* are related to the inflexion points; *shifts* are related to forming new clusters/fuzzy rules.

tonomous self-developing, self-learning, and evolving models and systems.

3.1 Detecting Drifts by Age of Clusters (Rules) [1]

In [1], the concept of cluster (respectively fuzzy rule) *age* was introduced. We extend this by including the membership degrees of the rules ($\Phi_{i,l}$ in case of the ith rule for the lth sample)

$$age^i = k - \frac{\sum_{l=1}^{n_i} I_l \Phi_{i,l}}{n_i} \quad (1)$$

where i is the rule index; n_i denotes the support of rule i; I_j denotes the time instance when the data sample was read; k is the current time instance. Since $\frac{\sum_{l=1}^{n_i} I_l}{n_i}$ can also be accumulated recursively, the *age* can be easily calculated when necessary. *Age* of the cluster/rule changes with each new data sample being read. If the newly read data sample does not fall into that cluster (supports that fuzzy rule with a low or even 0 value of $\Phi_{i,l}$ in case of fuzzy sets with finite support) the *age* grows by the rate of one sample at a time. That means, if a certain cluster (fuzzy rule) is not supported by any future samples after being initiated (say in the time instant, t_i) then its *age* at any time instant, k will be $k - t_i$. However, if any new data sample (between t_i and k) falls into that cluster (supports that fuzzy rule with a high value of $\Phi_{i,l}$), the *age* does not grow with the same rate, but with a smaller one and one can say that the cluster (fuzzy rule) is being *refreshed*.

The fuzzy rule *age* is important and closely linked to the data streams (which are sequences of data in time) and to the concept *drift*. We propose to analyze the *age* of clusters (respectively, fuzzy rules) on-line by using the gradient of the *ageing* curve as well as its second derivative which indicate a change in the slope of the *ageing* curve. When there is a **significant** change of the *ageing* which results in a significant change of the slope, then obviously the second derivative of the *age* curve will be indicative of this inflection points. An example is demonstrated in Figure 3, where the *drift* and *shift* are clearly marked on the *age* evolution curves.

3.2 Detecting Shifts by Utility Function

Shift in the data streams is a more significant, abrupt and sudden change of the data concept. Therefore, the distinction between *drift* and *shift* is the pace, the degree and speed of the changes, while both indicate a change in the data distributions with time. Essentially, the reaction to a detected *shift* has to be reflected in the structure evolution of the fuzzy rule base. While *drift* is related more to a smooth forgetting of parameters, learning and forgetting, *shift* is closely related to changes in the structural components.

Shift can be detected by the *utility* of the fuzzy rule. *Utility* is a parameter of the quality of the fuzzy rule that is defined [2] as the accumulated firing level of each rule given by Ψ_i summed over the life of each rule:

$$U_k^i = \frac{1}{k - I_k^i} \sum_{l=1}^{I_k^i} \Psi^l \quad (2)$$

$Utility$, U accumulates the weight of the rule contributions to the overall output during the life of the rule (from the current time instant back to the moment when this rule was generated). It is a measure of importance of the respective fuzzy rule comparing to the other rules (comparison is hidden in the relative nature of Ψ).

If the *utility* of a fuzzy rule is *low* then this rule becomes obsolete or not used very much. A *shift* away from or to a cluster/fuzzy rule can be detected by the derivative of the *utility* — if the value of the first derivative of the *utility* is *large* then there is a *shift* towards the cluster. The *shift* is away from a cluster if the derivative of the *utility* is *large* negative. Then this fuzzy rule is deemed obsolete (does not contribute significantly to the overall prediction/estimation/classification) and can be removed from the rule base [10].

4 Reactions to Drifts and Shifts in Data Streams by EFS

4.1 Reaction to Drift and Shift in the Antecedents

In this section, we introduce methods for addressing both *drifts* and *shifts* in data streams by adapting learning mechanisms for consequent parts and by evolving antecedent part. We demonstrate this on the example of the popular *eTS* method [4] and on *FLEXFIS* [20].

4.1.1 *By* eClustering

Reaction to a detected *shift* is by either a) forming a new rule around a new data sample which becomes an attraction point for the global data distribution, or b) replacement of a fuzzy rule which itself consists of; i) forming a new rule around the new point, and ii) removal of the rule which has lower density and is close to this newly added one [2] — see Figure 4. If locally optimal learning is being applied then removing a cluster and respectively a locally valid Kalman filter/RLS does not affect the overall learning significantly (only through the fuzzy weight Ψ [4]). If globally optimal learning is being applied, removal of a cluster (respectively fuzzy rules) does affect n columns and n rows of the covariance matrix directly and the remaining values of the covariance

ISBN: 978-989-95079-6-8

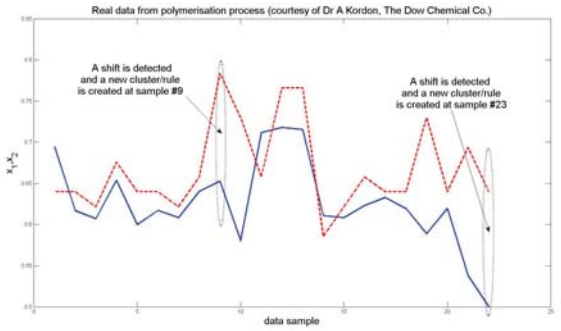

Figure 4: Shift in the time domain representation and two new rules evolved as a reaction to it.

matrix indirectly [4]. Deleting *old* fuzzy rules was applied in classification in [5] and in prediction [2].

4.1.2 *By* eVQ

When *shifts* in the data stream occur, usually new clusters are evolved automatically in *eVQ*, as they cause new data clouds in previously unexplored regions of the data space (in regions further away than a fraction of the space diagonal in the input space to which the vigilance parameter ρ is usually set, i.e. 0.1 to 0.3 of the space diagonal, see [19]. Synchronously, older clusters can be deleted by the concept of *utility* [9] [10] as discussed in Section 3.2. This does not affect the learning of consequent parameters for the other rules if local learning is applied.

For reacting to *drifts* in the antecedent parts of the rules we propose to re-adjust the parameter in the eVQ clustering algorithm η [19] which steers the learning gain. We define the tracking concepts for the ith rule throughout this section, which can be generalized to any rule in a straightforward manner. Currently, in *eVQ* the learning gain is defined by the following formula [18]:

$$\eta_i = \frac{0.5}{n_i} \tag{3}$$

If a *drift* occurs in a data stream, the centers and widths of the cluster(s) should change to the new data distribution (as shown in Figure 1). For re-activating the converged clusters, i.e. re-animating them for stronger movements in a drifting case, we suggest a sudden increase for the first sample in the *drift* phase, followed by a gradual decrease for the next samples in order to balance in the new sample distribution in the same manner as is done for original ones.

Here, we propose the following mechanisms for the learning gain η: first we transform the forgetting factor λ, used in the gradual out-weighting when doing consequent learning (see next section) and denoting the intensity of a *drift*, to a value in $[0, 1]$. Hereby, 0.9 (minimal value for λ) is mapped to 0.99, whereas 1 is mapped to 0, hence:

$$\lambda\_trans = -9.9\lambda + 9.9 \tag{4}$$

Then, when a *drift* occurs, we re-set the number of samples forming the ith cluster (n_i) (used in the denominator of the calculation of η_i) by

$$n_i = n_i - n_i * \lambda\_trans \tag{5}$$

This means that the stronger the drift is, the more n_i is decreased and hence the stronger the forgetting effect will be. In Figure 5 it is demonstrated how η_i develops (lines) in usual (non-drift) case (for the first 100 samples), then a drifting scenario is artificially caused with three intensities leading to the three λ values (in different line styles). After the *drift* indicator (at sample 100), it is decreased in usual way such that the jumped center can converge to the new data distribution.

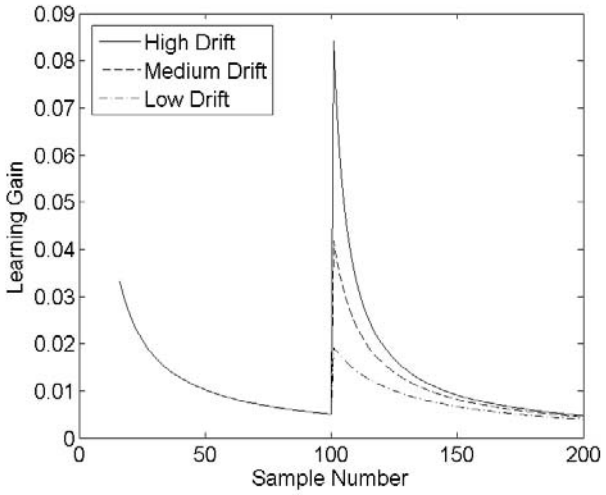

Figure 5: Abrupt increasing learning gain η during in case of a *drift* (after 100 samples) when applying different values for the forgetting factor λ.

4.2 *Reaction to Drifts and Shifts in the Consequents*

For the rule consequents *drifts* and *shifts* can be handled in one sweep as it is just a matter of the setting of the forgetting factor as we will see below. Whenever a *drift* in the output variable occurs (as shown in Figure 2) and is detected, it is necessary to apply a specific mechanism in the sample-wise incremental learning steps of the consequent parameters in Takagi-Sugeno fuzzy systems. When locally optimal fuzzily weighted Recursive Least Squares (wRLS) learning approach [4] is used this can easily be accommodated as detailed later on.

If we do not take *drift* into account all newer samples lying in the same local positions relative to the rules as the older samples are included with the same rule weights in the update process. Then the fuzzy model will end up with approximation curve as shown in the left image of Figure 6 with dotted lines. Obviously, the approximation ends up in the middle of the two trajectories (the newer after the *drift* shown in light font, the older shown in darker font), as trying to minimize the quadratic errors (least squares) of all samples to the curve. Hence, it is necessary to include a parameter in the update process, which forces older samples to be out-dated over time. Gradualism is important here in

ISBN: 978-989-95079-6-8

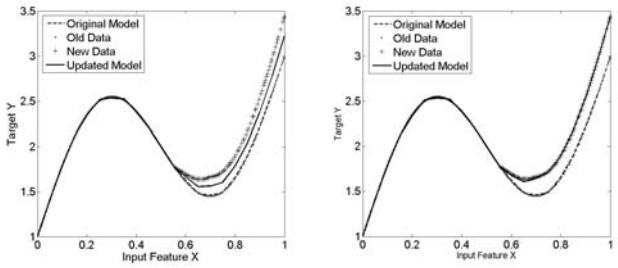

Figure 6: Left: The adapted model (dotted line) with the new incoming samples (dark dots) when applying conventional recursive weighted least squares approach; right: the adapted model (dotted line) by including a forgetting factor of 0.95 and using (7)-(9), the approximation surface lying exactly on the trajectory of the data samples denoting the novel data distribution

order to guarantee a smooth forgetting and to prevent abrupt changes in the approximation surface. For doing so, we re-define the least squares optimization function for the ith rule by

$$J_i = \sum_{k=1}^{N} \lambda^{N-k} \Psi_i(\vec{x}(k)) e_i^2(k) \longrightarrow \min_w \quad (6)$$

with $e_i(k) = y(k) - \hat{y}(k)$ the error of the ith rule in sample k. Assuming N the number of samples loaded so far, this function out-dates the sample processed i steps ago by λ^{N-k}. Usual values of λ lie between 0.9 and 1, where a value near 1 means a slow forgetting and a value near 0.9 a fast forgetting of former loaded data samples and the exact choice depends strongly on the strength of the *drift* (see below). Following a similar recursive deduction scheme as in conventional recursive least squares (RLS) [17], we obtain the following incremental update formulas for consequent parameters $\vec{w}_i$ for the ith rule:

$$\hat{\vec{w}}_i(k+1) = \hat{\vec{w}}_i(k) + \gamma(k)(y(k+1) - \vec{r}^T(k+1)\hat{\vec{w}}_i(k)) \quad (7)$$

$$\gamma(k) = \frac{P_i(k)\vec{r}(k+1)}{\frac{\lambda}{\Psi_i(\vec{x}(k+1))} + \vec{r}^T(k+1)P_i(k)\vec{r}(k+1)} \quad (8)$$

$$P_i(k+1) = (I - \gamma(k)\vec{r}^T(k+1))P_i(k)\frac{1}{\lambda} \quad (9)$$

with $P_i(k) = (R_i(k)^T Q_i(k) R_i(k))^{-1}$ the covariance matrix and $\vec{r}(k+1) = [1 \;\; x_1(k+1) \;\; x_2(k+1) \;\; \ldots \;\; x_p(k+1)]^T$ the regressor values of the $k+1$th data sample, Ψ_i the fulfillment degree of the ith rule, serving as weight in the recursive least squares algorithm.

The final question is how to set the parameter λ in order to guarantee an appropriate *drift* tracking. We propose a strategy to deduce it directly from the *age* curves analysis [10] since they are indicative of the speed of a *drift* (see Section 3). In Section 3, it was mentioned that the age of a rule always lies in $[0, k]$. Hence, we normalize the *age* of the ith rule to $[0, 1]$ by $age_i\_norm = \frac{age_i}{k}$ in order to achieve gradients of the normalized rule ages Δage_i_norm also lying in $[0, 1]$. Whenever the change of the gradient in the rule *age* curve is significant, recursive weighted RLS with forgetting (wRLSf) should be triggered. We use the following estimation for λ:

$$\lambda = 1 - 0.1\Delta^2 age_i\_norm \quad (10)$$

This guarantees a λ between 0.9 (strong forgetting) and 1 (no forgetting), according to the degree of the gradient change (1 = maximal change, 0 = no change). The forgetting factor is then kept for a while at this level (otherwise only one single sample would cause a gradual forgetting) and set back to 1, after a stable gradient phase is achieved (usually after around 20 to 30 samples showing a moderate value of $\Delta^2 age_i\_norm$). Setting back to 1 is necessary, as otherwise the forgetting will go on inside the new data distribution. This cause the drift phase in the antecedents to stop.

5 Evaluation

This section deals with the evaluation of the impact of reacting on *drifts* and *shifts* in case of data streams where actually *drifts* and *shifts* occur. This is done by implementing the approaches discussed throughout this paper. One application example is coming from a rolling mill, where the task was to identify a prediction model on-line for the resistance value of a steel plate at a rolling mill. The other application concerns a polymerization process in chemical industry.

5.1 On-Line Prediction Models at Rolling Mills

The task was to identify a prediction model for the resistance value of a steel plate at a rolling mill. This should be done in a first step with some off-line (precollected) measurement data in order to obtain a feeling about the achieved quality of the fuzzy model and then to refine the prediction model with newly recorded online data. The later step was possible as first a prediction for the resistance is given, influencing the whole process at the rolling mill, whereas a few seconds later (after the steel plate is passed), the real value for the resistance is measured, which can then be incorporated into the model adaptation process. In this sense, the correct measured value not the predicted (which might have been wrong)one is taken for learning. In fact, an improvement in terms of predictive power could be achieved when updating the fuzzy models trained in batch mode with 6000 samples during on-line operation mode with further 6600 samples. For details of the experimental setup and results see [20] (and also Table 1 below summarizing all the results).

Now, in this paper we want to examine whether a reaction onto drifts by a gradual forgetting of older samples during the on-line process may further improve the quality of the models. A justification of an application of reaction onto *drifts* is that the operation process at rolling mills is divided into different "stitches". One stitch represents one closed cycle in the rolling process. In the on-line mode the measurements come in continuously from stitch to stitch. However, for the current processed stitch, the previous stitch should play only little or even no role. However, the measurements from the

ISBN: 978-989-95079-6-8

Table 1: Comparison of prediction with and without applying gradual forgetting

Method	**MAE**	**Max MAE Too High / Max MAE Too Low / # MAEs > 20**
Analytical	7.84	63.47 / 87.37 / 259
Static fuzzy models	6.76	45.05 / 81.65 / 176
FLEXFIS	5.41	38.05 / 78.88 / 159
FLEXFIS with forg.	4.65	31.99 / 74 / 68

previous stitch are already included in the fuzzy models as updated by their samples. Thus, this means that older samples from the previous stitch should be forgotten when including samples from the current stitch. Another aspect is that here we do not need any drift/shift detection, as the drift/shift is indicated by the beginning of a new stitch. As no *drift* detection was carried out, λ was set to 0.97, which is a good compromise between fast forgetting (=strong locality of models) and low forgetting (=weak locality of models). The results are demonstrated in Table 1. Here, we also demonstrate the improvement of the predictive accuracy over analytical models by both, static and evolving fuzzy models. Three different types of errors are reported: the mean absolute error over all on-line samples (note that first a prediction is done and then the model updated with the same samples and based on feedback), the number of mean absolute error greater than 20, the maximal mean absolute error over all samples where the prediction was too low and the maximal mean absolute error over all samples where the prediction was too high. The latter value is the most important one as harming the steel plate is more dangerous as in case of predicting too low values. The results (Table 1) demonstrate the impact of the gradual forgetting.

Another interesting aspect is that the error on the single measurements starts to *drift* over time when gradual forgetting is not applied. This is underlined in the left image of Figure 7 which shows the single errors over the 6600 on-line samples: note the *drift* of the main error area away from the zero line at the end of the data stream.

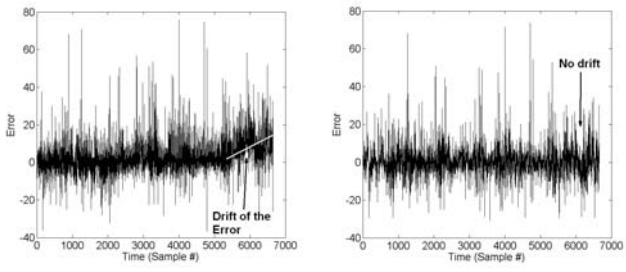

Figure 7: Left: The error curve for the 6600 on-line samples when no forgetting is applied: at the end the error starts drifting; right: no *drift*.

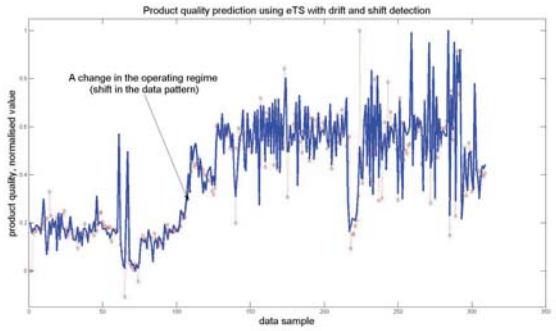

Figure 8: A comparison of the predicted and real data of product quality. Solid line - predictions by *eTS* with *shift* and *drift* detection; diamonds - real data.

5.2 *Another application example*

A case study based on real data (courtesy of Dr. Arthur Kordon, The Dow Chemical Co.) from the chemical industry is used as an illustration of the detection and reaction to *drift* and *shift* in *eTS* [6]. The *eTS* has been applied for prediction of the properties of a chemical composition by modelling the product composition in a distillation tower. The data set includes a change of the operating regime of the process which brings a challenge to the structure of the model (fuzzy rule based system). The data set includes also a number of other challenges, such as noise in the data, a large number of initial variables, etc. These problems cover a wide range of real issues in the industry. The process data is retrieved from physical ('hard') sensors used as inputs to the *eTS* applying hourly averages for every eight hours. The product composition (real output) is estimated by a laboratory analysis for comparison (it is given with diamonds in the Figure 8).

The estimation of the product composition contains noise due to the nature of the analysis. A significant operating condition change takes place after sample 127 (please see the Figure 8). The *eTS* was able to efficiently detect and react to this *shift* as well as to the *drifts* as depicted in Figures 3 and 4. The overall prediction is very good and compares favorably with the conventional models as detailed in the Table 2. Note, that the non-dimensional error index (NDEI) is defined as the ratio of the root mean square error over the standard deviation of the target data and should be ideally 0 while the variance accounted for (VAF) is defined as the ratio between the variance of the real data and the model output and is given out of a maximum of 100 (when the predictions coincide with the real data).

6 Conclusion

In this paper, we propose novel strategies and techniques for addressing concept *drift* and *shift* in on-line data streams. Therefore, two EFS approaches (*eTS* and *FLEXFIS*) are exploited as on-line modelling methodologies. These are extended by mechanisms which are 1.) able to detect *drifts* and *shifts* with fuzzy rule *age* and *util-*

ISBN: 978-989-95079-6-8

Table 2: Error measures when applying *eTS* with (Column #3) and without (Column #2) drift detection and reaction in chemical composition modelling

Measure	Without	With	Best (theor.) value
NDEI	0.3559	0.33465	0
VAF, %	87.319	88.807	100
correlation	0.9357	0.94285	1

ity, and 2.) to react on such occurrences appropriately. The latter is applied 1.) to the rule antecedent parts directly in the cluster space for reacting on *drifts* and *shifts* in the input space and 2.) to rule consequent parameters for reacting on *drifts* and *shifts* in the output variable, applicable to any EFS technique exploiting Takagi-Sugeno type fuzzy systems. Evaluation on real-world data sets shows that the novel techniques are able to improve the accuracy and stability of the fuzzy models, whenever the occurrence of *drifts* and *shifts* is present.

References

[1] P. Angelov and D. Filev. Simpl_eTS: A simplified method for learning evolving Takagi-Sugeno fuzzy models. In *Proceedings of FUZZ-IEEE 2005*, pages 1068–1073, Reno, Nevada, U.S.A., 2005.

[2] P. Angelov and X. Zhou. Evolving fuzzy-rule-based classifiers from data streams. *IEEE Transactions on Fuzzy Systems*, 16(6):1462–1475, 2008.

[3] P.P. Angelov. Evolving Takagi-Sugeno fuzzy systems from streaming data, eTS+. In P. Angelov, D. Filev, and N. Kasabov, editors, *Evolving Intelligent Systems: Methodology and Applications*, page to appear. John Wiley & Sons, New York, 2008.

[4] P.P. Angelov and D. Filev. An approach to online identification of Takagi-Sugeno fuzzy models. *IEEE Trans. on Systems, Man and Cybernetics, part B*, 34(1):484–498, 2004.

[5] P.P. Angelov, E. Lughofer, and X. Zhou. Evolving fuzzy classifiers using different model architectures. *Fuzzy Sets and Systems*, 159(23):3160–3182, 2008.

[6] P.P. Angelov and X.-W. Zhou. Evolving fuzzy systems from data streams in real-time. In *2006 International Symposium on Evolving Fuzzy Systems*, pages 29–35, 2006.

[7] J. Beringer and E. Hüllermeier. Efficient instance-based learning on data streams. *Intelligent Data Analysis*, 11(6):627–650, 2007.

[8] F. Hopner and F. Klawonn. Obtaining interpretable fuzzy models from fuzzy clustering and fuzzy regression. *Proc. 4th Intern. Conf. on Knowledge-based Intell. Eng. Syst. (KES)*, Brighton, UK, pages pp.162–165, 2000.

[9] P. Angelov. *Machine Learning (Collaborative Systems)*, patent (WO2008053161, priority date: 1 November 2006).

[10] P. Angelov and X.-W. Zhou. On Line Learning Fuzzy Rule-based System Structure from Data Streams. In *2008 IEEE International Conference on Fuzzy Systems within the IEEE World Congress on Computational Intelligence*, Hong Kong, June 1-6, 2008, pages 915–922, 2008.

[11] P.P. Angelov *Evolving Rule-based Models: A Tool for Flexible Systems Design*, Springer Physica Verlag, Heidelberg, Germany, February, 2002.

[12] N. Sundararajan H.-J. Rong, G.-B. Huang, and P. Saratchandran. Sequential adaptive fuzzy inference system (SAFIS) for nonlinear system identification and prediction. *Fuzzy Sets and Syst.*, 157(9):1260–1275, 2006.

[13] N. K. Kasabov and Q. Song. DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE Trans. on Fuzzy Systems*, 10(2):144–154, 2002.

[14] R. Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, vol. 8 (3), pages 281–300, 2004.

[15] G. Leng, T.M McGinnity, and G. Prasad. An approach for on-line extraction of fuzzy rules using a self-organising fuzzy neural network. *Fuzzy Sets and Syst.*, 150(2):211–243, 2005.

[16] E. Lima, F. Gomide, and R. Ballini. Participatory Evolving Fuzzy Modeling. In P. Angelov, D. Filev, and N. Kasabov, editors, *2006 International Symposium on Evolving Fuzzy Systems*, IEEE Press, pages 36–41, 2006.

[17] L. Ljung. *System Identification: Theory for the User*. Prentice Hall PTR, Prentic Hall Inc., Upper Saddle River, New Jersey 07458, 1999.

[18] E. Lughofer. *Evolving Fuzzy Models — Incremental Learning, Interpretability and Stability Issues, Applications*. VDM Verlag Dr. Müller, Saarbrücken, 2008.

[19] E. Lughofer. Extensions of vector quantization for incremental clustering. *Pattern Recognition*, 41(3):995–1011, 2008.

[20] E. Lughofer. FLEXFIS: A robust incremental learning approach for evolving TS fuzzy models. *IEEE Trans. on Fuzzy Systems (sp. issue on EFS)*, 16(6):1393–1410, 2008.

[21] E. Lughofer. Towards robust evolving fuzzy systems. In P. Angelov, D. Filev, and N. Kasabov, editors, *Evolving Intelligent Systems: Methodology and Applications*, to appear. John Wiley & Sons, New York, 2009.

[22] E. Lughofer, P. Angelov, and X. Zhou. Evolving single- and multi-model fuzzy classifiers with FLEXFIS-Class. In *Proc. FUZZ-IEEE 2007*, pages 363–368, London, UK, 2007.

[23] E. Lughofer, J. E. Smith, M. A. Tahir, P. Caleb-Solly, C. Eitzinger, D. Sannen, and M. Nuttin. Human-Machine Interaction Issues in Quality Control Based on On-Line Image Classification. *IEEE Trans. on Systems, Man and Cybernetics part A: Systems and Humans*, 2009, to appear.

[24] S. Ramamurthy and R. Bhatnagar. Tracking recurrent concept drift in streaming data using ensemble classifiers. In *Proc. 6th Intern. Conf. on Machine Learning and Applic. (ICMLA), 2007*, pages 404–409, 2007.

[25] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. on Systems, Man and Cybernetics*, 15(1):116–132, 1985.

[26] A. Tsymbal. The problem of concept drift: definitions and related work. Technical Report TCD-CS-2004-15, Dept of Comp Science, Trinity College Dublin, Ireland, 2004.

[27] L.X. Wang. Fuzzy systems are universal approximators. In *Proc. 1st IEEE Conf. Fuzzy Systems*, pages 1163–1169, San Diego, CA, 1992.

[28] S. Wu, M.J. Er, and Y. Gao. A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks. *IEEE Trans. on Fuzzy Systems*, 9(4):578–594, 2001.

ISBN: 978-989-95079-6-8