Incorporating Dynamic Uncertainties into a Fuzzy Classifier

Jens Hülsmann,[1] **Andreas Buschermöhle,**[1] **Werner Brockmann**[1]

[1]University of Osnabrück
{jens.huelsmann, andreas.buschermoele, werner.brockmann}@universitaet-osnabrueck.de

Abstract

Dealing with classification problems in practice often has to cope with uncertain information, either in the training or in the operation phase or both. Modeling these uncertainties allows to enhance the robustness or performance of the classifier. In this paper we focus on the operation phase and present a general, but simple extension to rule based fuzzy classifier to do so. Therefor uncertain features are gradually and dimension wise faded out of the classification process. An artificial two–dimensional dataset is used to visualize the effectiveness of this approach. Investigations on three benchmark datasets shows the performance and gain in robustness.

Keywords: fuzzy classifier, uncertainty, trust management

1. Motivation and Background

In complex technical systems, the likelihood of data being uncertain increases due to disturbances like noise or anomalies, uncertain system properties and/or the combination of uncertain information in multi-staged signal processing. Typical examples are automotive driver assistant systems which rely on camera, laser or radar data to perceive their environment in order to identify other traffic participants and to figure out the current traffic situation. But the sensors themself and the processing of input signals introduce uncertainty, e.g. due to occlusion and/or glaring of a sensor. The decision the driver assistant system has to come to is thus based on these uncertain data. And what is more, these uncertainties vary over time, depending on the current situation at runtime.

The sources of these uncertainties can be classified in general as follows:

- physical effects, e.g. sensor-noise, outliers, drift, faults
- informational uncertainties, e.g. unknown or conflicting information
- insufficient specification, e.g. suboptimal or incomplete parametrization
- interaction or actuator faults, e.g. unforeseeable interaction patterns of actuators with the environment

This point of view covers and complements considerations of the classical fault tolerance community, e.g. [1].

In this paper we focus on technical systems that perform a classification algorithm. The features, which serve as the inputs of the classifier, hence transfer all these uncertainties to the classifier. For classification tasks, the first three categories of uncertainty-sources are important, in closed loop applications also the fourth.

Fortunately, varying degrees of uncertainty are known for example for many sensor systems and data sources. E.g., the spatial resolution of stereo camera images depends on the distance of an object. The accuracy of a calculated object position hence varies over its relative position. Or, the degree of certainty of sensor readings may vary dynamically due to ageing or de- and re-calibration. Even in pure offline applications the degree of certainty may vary, e.g. depending on the source or maturity of the data.

Also the classification algorithm itself has to be seen as a source of uncertainty. Decision making by classification of data is generally associated with some ambiguity if a pattern falls to a certain extent between two (or more) classes. Most classification algorithms hence calculate similarity or distance measures to clusters (or cluster centers) in order to find the most appropriate one. These (internal) distance and/or additional discriminance measures can be used to determine a confidence measure, or a degree of certainty, respectively, for a decision.

In critical classification tasks, classifiers are often used with a reject option. If the discriminant measure gets too low, the classifier assigns no class to the input feature vector. This is especially crucial in embedded systems and safety critical applications, where the costs of a misclassification are too high.

Additionally, the knowledge which is incorporated in the classifier may be incomplete because the feature space was not completely covered by training data, or uncertain, because the training data were uncertain or conflicting.

As the design of dependable complex systems is a challenge today, these uncertainties need to be tackled explicitly. This should not reduce engineerability on the one hand. On the other hand, it is important to still achieve the best possible performance under the given circumstances without sacrificing a dependable, trustworthy system operation.

In [2] we therefor introduced a generic framework called *trust management.* One main focus of trust management is to deal with (dynamic) uncertainties explicitly and to make them as comprehensible as possible. The knowledge about uncertainties is represented by complementing system variables and components with attributes, which are called *trust level.* They indicate the trustworthiness of a signal value or of a module by a scalar measure ϑ. In this way, modules can explicitly consider the trustworthiness of inputs and internal information and fade adaptively between a high performance behaviour in case of trustworthy information and a robust behaviour for low trust.

Our work also shows how trust management can be incorporated into a continuous signal processing task [2]. Complementary to that, the work presented here is intended to show how to balance the performance and trustworthiness of classification algorithms, i.e. a discrete operation. Because we aim at complex systems, a classification algorithm with trust management must be easy to engineer and the incorporation of uncertainty processing has to be understood easily. In order to be suited for embedded systems (e.g. automotive systems), operation speed and costs also matters. Finally, (hard) real-time capabilities may be required.

2. Related Work

2.1. Classification and Uncertainty

When dealing with uncertainty in classification systems, one has to distinguish when and how the uncertainty comes into play, namely in the *training* or in the *operation* phase. During the operation phase, the source of uncertainty can be an external one, i.e. the feature vector or a part of it is uncertain, or an internal one, i.e. the classifier itself is uncertain. During the training phase uncertainty can be present within the training feature vectors or in the assignment of a feature vector to a class. In addition, uncertainty can be present during the whole phase to the same degree, but also to a varying degree.

In all major classification algorithms a discriminant value of the class assignment at the output can be obtained. This is derived e.g. based on internal distance measures. A classifier is then less certain the larger the distance or the less the discriminance is in the operation phase.

An approach to get a more significant discriminance measures is the concept of conflict and ignorance [3] . For a given feature vector, the ignorance is a measure for the distance to any class, the conflict a measure for the degree of overlap between classes at the feature vector. In [3] this is used to monitor the reliability of a fuzzy rule-based classification.

Another method to deal with uncertain classifications is to combine different classifiers [4] and to use the discriminant measures of different classification algorithms to determine, which result is the most trustworthy.

There are different approaches to incorporate knowledge about the uncertainty of the training data. In a rule based classifier, the assignment of the rules to the classes can be handled with evidential reasoning [5]. This yields a better measure of the classification uncertainty in the presence of static, known uncertainty in the training data.

Given statistical properties of the uncertain training data, one can reformulate the support vector machine (SVM) training problem to accommodate this additional information offline at training time, which results in an optimization problem of the second order cone program type [6]. It can be shown that one obtains more robust solutions by this approach.

The problem is that these approaches work statically, i.e., the uncertainty is only either used during training, or assumed to be known but constant. Hence the same degree of certainty is applied in both phases. Because of this, these approaches are not capable to adapt to dynamically changing uncertainties of the input data in the operation phase.

A totally uncertain input can be viewed as a missing input. The problem is similar then to the well studied task of classification with missing features [7], but only for this borderline case. It is hence not gradually applicable.

A gradual fade out of input dimensions for rule-based fuzzy classifiers is shown in [8]. The weights for each feature dimension are determined by measures for its importance.

For neural networks with Gaussian basis functions, [9] employs a weighted integration over the missing or uncertain input. This can be done during training as well as operation phase, but requires an approximation of the local probability distribution of the input. This (weighted) integration approach is a general solution to the problem of a missing or uncertain input feature, but it is a very expensive operation. One has to sample over all dimensions, that are missing or uncertain. It further requires a priori knowledge about the probability distribution.

In [10] we extended a SVM in order to increase the robustness by explicitly modeling the uncertainty of the input data at runtime, i.e., dynamically during the operation phase. In terms of engineerability and interpretability, a fuzzy classifier is superior to the SVM. The rules can be parameterized in detail and learned rules can be investigated easily. The uncertainty of the training data can be assumed to be stored implicitly within the fuzzy rules considered here. Thus, the aim of this work is to extend a fuzzy-classifier to handle dynamic uncertainties of in the operation phase, i.e. within the feature vector.

2.2. Uncertainty Representation

The information about the trustworthiness of a part of a technical system can be determined in different ways. For sensors often a sensor model or additional information from other sensors is used. For example, a sensor reading near to the end of a measurement range often is not very trustworthy due to non-linear sensor effects. In the further processing steps the results can not be better then the data they rely on, except there was redundancy in the source data or additional information is provided e.g. by a system model.

In the framework of *trust management* [2], a *trust signal* is a meta-information to a normal signal, e.g. a sensor-reading, or to an internally generated signal, which depends on uncertain information. Also system components or functional modules can be attributed with a trust signal. In our case, all these uncertainty information is reflected by trust signals.

A trust signal has a scalar value from the interval $[0, 1]$, called the *trust level*, and indicates the trustworthiness of the signal/component it is associated with.Two rules generally apply here: If the trust level is 0.0, the value has not to influence the output. If it is 1.0 the data can be fully trusted, hence it can be handled as normal. It is important to note that the trust signal is not a probabilistic representation, hence it does not declare anything about the statistical properties of the data assigned to.

The module, which receives the trust signal afflicted data, has to decide, in which way it incorporates the regular and the trust level data into the processing of its output. If the input data are not trustworthy enough, it can switch to a fall back strategy or gradually fade out the influence of the affected input(s). From its processing, a module can again make a statement about the trustworthiness of its output.

3. Extended Fuzzy Classifier Approach

3.1. Fuzzy Classifiers

In fuzzy rule-based classifiers, classes are associated to fuzzy partitions of the feature space by the rule antecedents. For a given feature vector $\mathbf{x} \in \mathbb{R}^D$, the rules are evaluated according to the FITA scheme. I.e., for a given feature vector $\mathbf{x}$ the input membership functions of the rules determine their firing strengths. Then the inference is done by a t-norm and the consequent parts of the rules are aggregated by a s-norm. This yields the class label confidences, or a compatibility measure $\mu_c(\mathbf{x})$ for any class c, respectively. In case of N classes, the result of the classification procedure is a compatibility vector with N entries.

$$\mu(\mathbf{x}) = (\mu_1(\mathbf{x}), \mu_2(\mathbf{x}), \ldots, \mu_N(\mathbf{x}))$$

Because this is a very general inference scheme, many different forms of fuzzy classifier systems exist [11]. They differ in the type of membership functions [12], in the applied t- and s-norms as well as in pre and post processing methods [13, 14].

The final decision is determined out of these compatibility measures $\mu_c(\mathbf{x})$, e.g. by choosing the class which fits best to the given feature vector x, i.e. which has the highest compatibility measure μ_{m1} because it is most likely to be the right class for a given feature vector.

In cases where a misclassification is too expensive and has thus to be avoided, a classification is only accepted if additional constraints are met. E.g. the compatibility measure μ_{m1} of the chosen class has to exceed a certain threshold. In order to improve the trustworthiness of a classification, additional measures can be calculated [3]. E.g. the difference to the second largest compatibility measure μ_{m2}, should also be greater than another threshold . Otherwise the classification is rejected. This avoids classifications in areas of the feature space where classes are conflictiong.

Rejecting a classification is important for us because we are aiming at safety-critical applications. Due to the same reason ease of engineering and interpretability of the classifier's behavior come also into play. We thus design our classifiers such that their behavior can be controlled and interpreted easily in the training phase as well as in the operation phase and that the number of rules is limited. Hence trapezoidal membership functions are used because of their limited support, and the rules are evaluated according to the max-min-inference scheme. But the following extensions are not limited to this choice. The important point is now how a classifier can be incorporated into the trust management framework such that a good classification performance is achieved, misclassifications are avoided as far as possible and a safe and trustworthy operation of the overall system is supported.

One first step is that the high compatibility measure μ_{m1} as well as the discriminant measure

$$\Delta\mu(\mathbf{x}) = \mu_{m1} - \mu_{m2}$$

can be treated as trust signals of (ordinary) fuzzy classifiers. The important point is now how to incorporate dynamic uncertainties of the feature vector into a fuzzy classifier.

3.2. CLARISSA-Approach

The CLARISSA-approach (CLAssification by a Rule-based Inference System for Safety-critical Applications) extends rule-based classifier approaches as follows.

Because the fuzzy classifier is embedded in the trust management framework, we can assume a trust signal ϑ_d is given for every feature x_d in the feature vector $\mathbf{x}$. In order to incorporate the classifier consistently into this framework, our approach is to fade out the influence of uncertain/untrustworthy features on the classification result. The respective class membership functions $\mu_{c,d}(x_d)$ are adjusted dimension wise if they have a $\vartheta_d < 1.0$ in order to take the uncertainty of the particular feature into account.

To do so, the respective membership functions are raised according to the trust level of each feature by a s-norm depending on the degree of membership to a class c in each dimension $d \in [1, D]$ and its trust level ϑ_d by:

$$\forall x_d \in \mathbf{x} : \mu'_{c,d}(x_d) = s(\mu_{c,d}(x_d), 1 - \vartheta_d) \quad (1)$$

As the calculation of the $\mu'_{c,d}$ is done dynamically at runtime individually for each incoming feature vector, the membership functions are kept unchanged.

This fairly simple, but general approach has several consequences. At first, the usage of an s-norm assures that the degree of membership is limited to $[0, 1]$. Additionally, the degree a rule fires and hence its support is getting higher over the whole dimension if the trust signal of a feature x_d is getting lower. This behavior causes that a rule applies to a certain degree over the whole universe of discourse of a feature x_d for a $\vartheta_d < 1.0$. This models the desired behavior that a given value of a feature can only be trusted to a lesser extend the lower it trust level is.

For the borderline case of a $\vartheta_d = 0$, the feature x_d is not used anymore to determine firing strength of a rule. This corresponds a projection, as the degree of membership over the whole dimension is one. This way only the remaining dimensions contribute to the firing strength of a rule. As shown in [8] this mechanism can also be used to cope with the curse of dimensionality.

By the selection of a particular s-norm one can easily select the wariness of the generalization in presence of gradual uncertain features. E.g. the maximum norm results in a slow fade out of the uncertain dimension, whereas the use of the co-Łukasiewicz norm results in quickly discarding the respective feature.

The second main consequence is that the discriminant measure $\Delta\mu(\mathbf{x})$, which is used as the output trust signal, automatically distinguishes the case of a single and multiple classes being affected by $\vartheta_d < 1.0$. In case of a single class, the compatibility vector $\mu(\mathbf{x})$ remains unchanged. The result is thus as trustworthy as without trust management because the respective feature is not relevant for separation for the current feature vector $\mathbf{x}$.

If multiple classes are affected, all their firing strengths are increased and yet the discriminant measure $\Delta\mu(\mathbf{x})$ is reduced. Hence the trust level of the classification result gets reduced. It is important to note that the discriminant measure is dependent on the specific local class distribution in the feature space.

Hence, the third and most important consequence is that this treatment of uncertainty does not imply any information about the class distributions and also of the value distribution of the uncertain feature along its dimension. In contrast,if one for example, would just enlarge the support of rules in the feature space, this would imply that uncertainty means a gradual deviation from the given feature value. As this is not intended in the framework of trust management, our approach yields the most general treatment of uncertainties.

4. Investigation Examples

4.1. Common Setup

This section investigates two scenarios in order to demonstrate the feasibility of the approach and its effectiveness. For the sake of clarity, the first one is a simple, artificial two-dimensional test scenario that can easily be visualized. The second scenario compares the fuzzy classifier with CLARISSA to the basic version without any trust management concerning the effect of randomly scattered input data. The classification quality is compared for three benchmark datasets. In both cases trapezoidal membership functions are used. The influence of the trust signals is realized according to (1) by the co-sum s-norm, which results in a medial fading behavior which is given by:

$$\mu'_{c,d}(x_d) = \mu_{c,d}(x_d) \cdot \vartheta_d + (1 - \vartheta_d)$$

4.2. Two–dimensional Test Scenario

The simple, artificial scenario consists of a two–dimensional input space with three classes and five training vectors for each class (dot, square and star in Figure 1). The rules are created with a core size of 0.8 and a support of 1.3. In Figure 1, the input space is sampled after training to get the approximate class regions in order to demonstrate the effect of considering the trust signal. These class regions are shown in Figure 1 from the left to the right for a a gradual degradation of the trustworthiness of feature 2 for decreasing values of its trust level ($\vartheta_2 \in \{1.0, 0.7, 0.4, 0.0\}$). The first feature is assumed to be certain ($\vartheta_1 = 1.0$).

With a decreasing value of ϑ_2, the classification result depends less on the uncertain input. The regions of the classes hence grow, if there is no other class region in the uncertain dimension. Otherwise the classifier rejects the classification of a feature vector due to a too low discriminant measure $\Delta\mu(\mathbf{x})$ in order to avoid any misclassification. As soon as the second feature has a trust level of $\vartheta_2 = 0.0$, i.e., no trust, it is completely ignored for the classification.

Two important properties of the approach can already be seen: A classification is still possible for regions of the feature space where any value of the

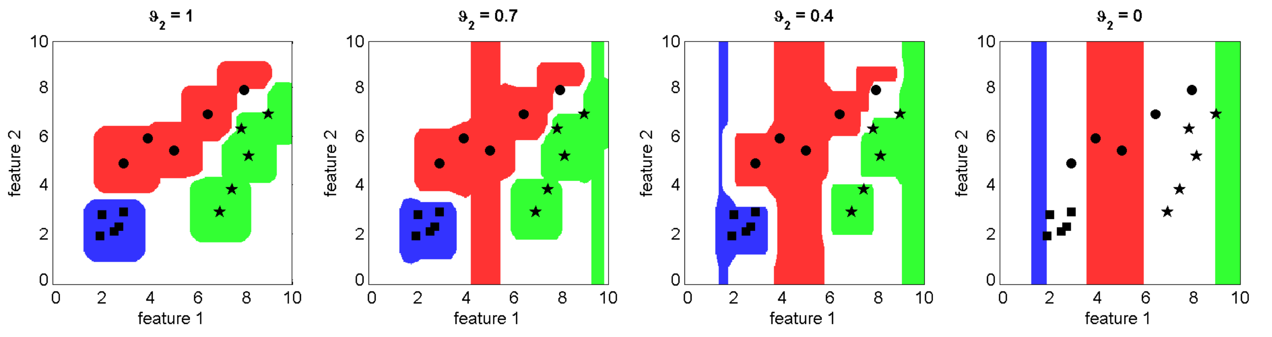

Figure 1: Training vectors (dot, square, star) and class regions (red, blue, green) for different trust levels ϑ_2 of the second feature in the extended fuzzy classifier approach (white area: classification rejected due to a discriminant measure $\Delta\mu(\mathbf{x})$ smaller than 0.3).

uncertain feature would not change the classification result (e.g. around a value of 5 for feature 1). Additionally, the gradual fading of the more and more uncertain feature x_2 does not result in a linear scaling of the class region in its dimension.

4.3. Higher Dimensional Test Scenarios

4.3.1. Generating reproducible experimental setups

In order to test the extended fuzzy classifier on higher dimensional benchmark data, uncertainty has to be added in a controlled fashion to the feature vectors which shall be classified. The most general kind of uncertainty is replacing a value from the feature vector by a random one (i.e. not just adding some noise to it). By using many combinations of such randomly selected values, all kinds of noise, faults and even the worst cases of changes in the feature vector are covered. As uncertainty in general means, a specific value may but must not be wrong, our trust level in this scenario reflects the probability of a value being drawn from an equal distribution. To simplify matters and to distinguish it from added noise, this procedure is called scattering in the following.

The investigation is done for three different benchmark data sets from the UCI machine learning repository [15] (Iris, Wine, Wisconsin Breast Cancer Diagnostic (WBCD)). The Iris dataset is based on 4 features, Wine on 13 and WBCD on 30.

In order to incorporate the scattering of the input data in a reproducible way, the different features of the data sets are first scaled into the interval $[0; 100]$ in each dimension $d \in [1, D]$. For a feature value from the feature vector x_d, the scattering then means, the respective value is replaced with a random value from the interval $[0; 100]$ according to the following probability:

$$p(x_d \in \{[0, 100]/\hat{x}_d\}) \quad = \quad 1 - \vartheta_d$$

Consequently this means, it remains its original value $\hat{x_d}$ with probability

$$p(x_d = \hat{x}_d) \quad = \quad \vartheta_d.$$

This means, that the ϑ_d is the probability to modify the value of the feature. Consequently they serve as the trust signals for the feature vector. For example, for a value x_0 with its a trust level $\vartheta_0 = 0.8$ this means, with a probability of 80% x_0 remais its original value. With a probability of 20% it is replaced with a random value.

To get an intuitive overall scalar measure, we introduce a *total trust* Θ. I.e. the particular trust levels ϑ_d of the feature vector are determined such that

$$\Theta = \prod_{d \in [1,D]} \vartheta_d.$$

This means, a feature vector with a given Θ consists of D features with their individual trust levels ϑ_d such that they multiply to Θ. With the given trust levels the scattering is then applied to each value $\hat{x}_d$ to get the x_d.

In order to eliminate random effects and to cover many cases of scattering in a reproducible way, a fixed set of random trust signal vectors are determined in such a way that one gets 1000 vectors $\vartheta = (\vartheta_0, \dots, \vartheta_D)$ for 100 different values of Θ, linearly distributed between 1 and 0. The test set hence comprises 100,000 different trust signal vectors. So this scenario can be interpreted as a situation in which there is a representative mixture of different degrees of certainty. This covers cases, where single features are very uncertain, as well as cases where many features are moderately uncertain.

The concrete procedure is then the following: For training, 35 randomly selected training vectors per class are used for each dataset without any artificially induced uncertainty. The rules are created with a core size of 10 and a support of 28 in each dimension. Afterwards, the remaining vectors of the dataset are used for testing after being scattered according to the given a set of trust signal vectors. Finally, the scattered values and their trust levels are passed to the classifier.

Obviously, the number of correctly classified feature vectors will drop as scattering is increased.

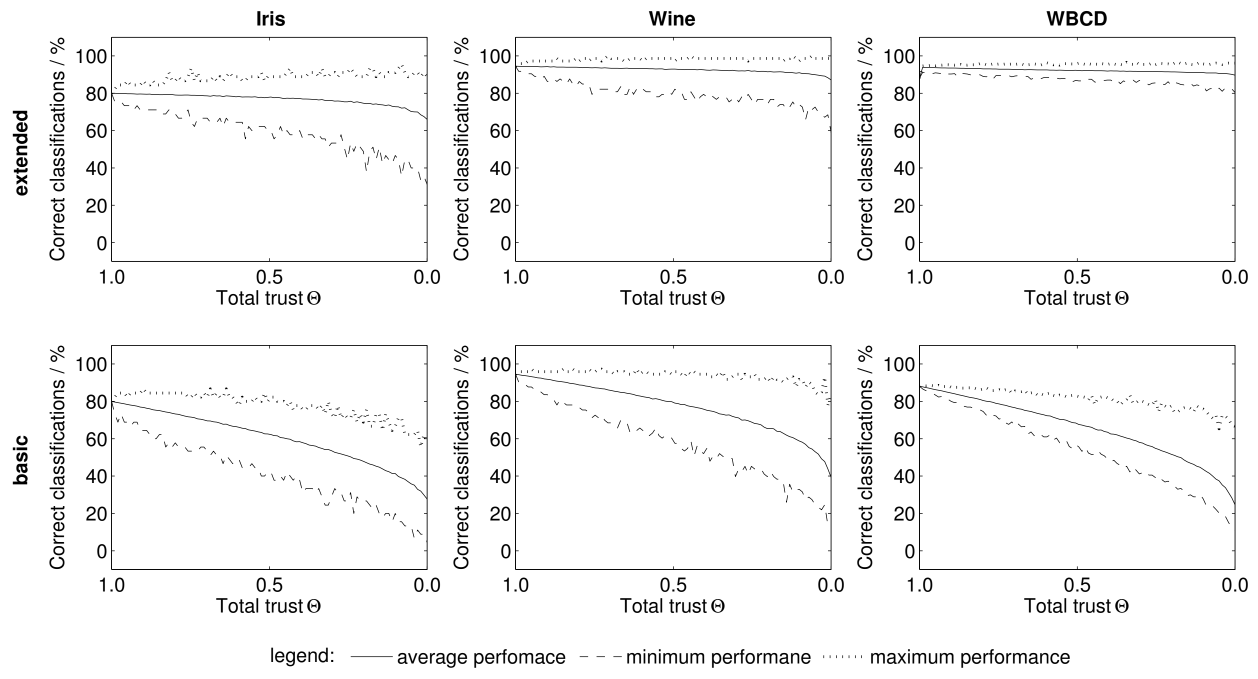

Figure 2: The percentage of correctly classified test vectors which have been scattered. The horizontal axis shows the (decreasing) measure for the trustworthiness of the feature vector (total trust Θ) which corresponds to the (increasing) uncertainty caused by the scattering (see the text for details). The extended fuzzy classifier (upper row) performs better than the basic one (lower row) on all three benchmark datasets.

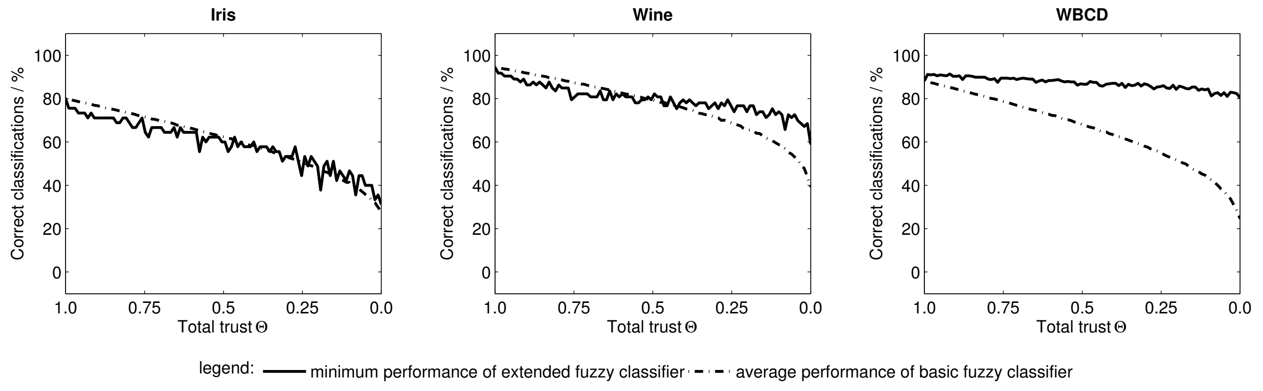

Figure 3: Comparison of the average classification performance of the basic fuzzy classifier (without trust management) to the minimal performance of the extended version (with trust management).

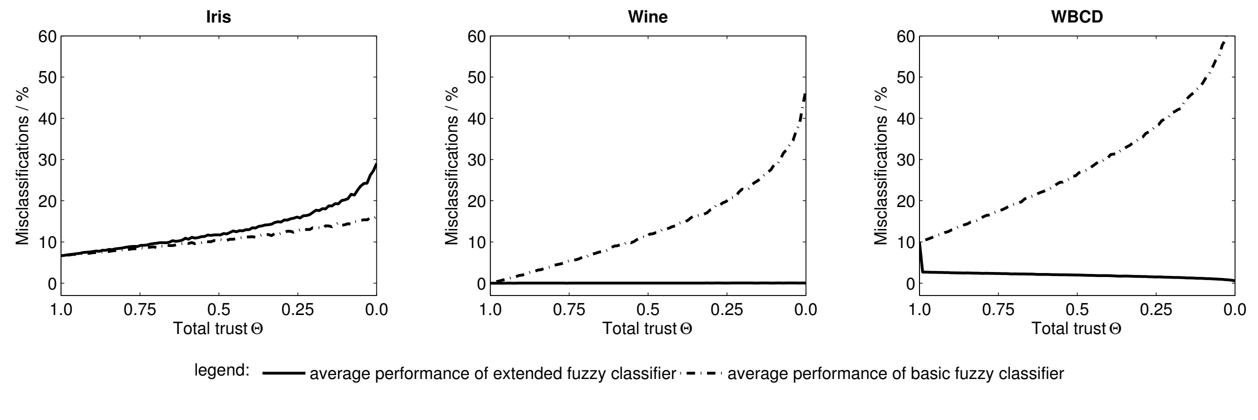

Figure 4: Comparison of a basic fuzzy classifier to an extended one concerning the average number of misclassifications.

This has three reasons. First, a feature vector belonging to a class A might be changed, so that it fits to no class. In this case (case I), the feature vector will not be classified. Second, a feature vector belonging to a class A might be changed to fit to class B. So in this case (case II), the sample will be misclassified. And third, a feature vector might fit to more than one class, so that it causes a conflict and is rejected, hence not classified (case III).

4.3.2. Results

Figure 2 shows the results for the three data sets for a gradually increasing degree of scattering in the feature vector, or total trust, respectively. The figure shows the minimal, the average and the maximal performance on each dataset (the percentage of correct classifications) for each of the 1000 investigations per degree of total trust. One can see that the extended fuzzy classifier and the basic one perform equally well on the test data as long as the trust is high. But as the total trust decreases, the extended fuzzy classifier quickly outperforms the basic one. It can be seen that the maximal performance of the extended fuzzy classifier stays at a high level even for very low trust, while the maximal performance of the basic fuzzy classifier drops to a rather low level. The average performance of the extended fuzzy classifier is also always better than the average performance of the basic one.

The performance gain is due to different reasons for each case of reduced classification quality (cases I to III, see above). As the extended fuzzy classifier adapts its classification boundaries dynamically depending on the current trust levels of the features, it is capable of reducing the effect of case I by still classifying such samples correctly. In the cases II and III, the extended fuzzy classifier is more likely to reject a classification than the basic one because of adapting the compatibility vector $\mu(\mathbf{x})$. This behavior is more cautious and thus avoids misclassifications. As a consequence, samples of case I will increase the number of correct classifications of the extended fuzzy classifier, while samples of the cases II and III will reduce it. So depending on the characteristics of the data set, case I or cases II and III will prevail, and accordingly, the extended fuzzy classifier will outperform the basic one if case I is dominant. This is more likely to be the case in high dimensional data sets, as then the classes tend to be further apart from each other within the feature space. Figure 2 confirms this, as the extended classifier yields the strongest performance gain for the dataset with the highest number of features (WBCD).

In contrast to [10], where similar investigations with gaussian noise (which is somewhat easier for a classifier) instead of scattering were made , we gain much better performance on the WINE dataset and in case of high uncertainty also for the IRIS data. For WBCD the performance is lower overall, which would very likely be remedied by optimizing the parameter of the fuzzy classifier for the dataset.

In order to compare the effect of the proposed extension in more detail, Figure 3 shows a comparison of the minimum quality of the extended fuzzy classifier and the average quality of the basic fuzzy classifier on the benchmark datasets. It shows that for high uncertainties even the minimal quality of the extended fuzzy classifier is at least as good as the average basic one or even outperforms it. The extended fuzzy classifier achieves a slightly lower minimal quality only for small uncertainties.

In addition, Figure 4 shows the occurrence of misclassifications with a raising uncertainty. For the Iris dataset the number of misclassifications grows. This is due to relatively dense allocation of the feature space by the classes. Case II dominates the classification quality.

In higher dimensional datasets like Wine or WBCD the extension does not introduce a new tendency towards misclassification Here our approach provides a more robust solution to dealing with uncertainties when there is an dynamic estimation of the degree of uncertainty.

5. Discussion

The investigations show that incorporating an explicit treatment of uncertainties in the operation phase improves the classification performance and robustness of a fuzzy classifier under uncertain conditions severely. No assumption about the concrete influence of the uncertainty on a feature value has to be made (e.g. probability distributions). It is only stated how confident it is. This is coherent with the fact that the approach does not scale the supporting region of a class along an uncertain dimension.

The proposed extension allows to accommodate a gradual decrease in trustworthiness without permanently changing the membership functions. Because of this, such an extended fuzzy classifier can deal with dynamically changing uncertainties robustly. This is complementary to other methods which only deal with static uncertainties or uncertainties during the training phase.

Within a conventional fuzzy classifier, one can also determine a measure of the *classification certainty* by looking at the discriminant measure. The same holds for the extended approach given in this paper But there is an additional effect. Any decrease of the certainty of the feature vector keeps the discriminant measure or decreases it. It thus additionally reflects the effect of uncertainty of the classifiers input on the classification certainty. Because of this, the approach given in this paper is ideally suited for the integration into a more complex system architecture, e.g. with multi–staged decision processes. Any form of uncertainty, be it disturbed features, missing data or classification uncertainties, can be treated in a uniform way throughout the different system stages.

Hence, the presented extension of fuzzy classifiers allows an explicit treatment of trustworthiness at runtime, although it is fairly simple and easy to compute. The approach presented here basically acts similar to a gradual dynamic selection of the most trustworthy input signals. It is thus easy to understand as well as to engineer. And what is more, with the proposed approach, the uncertainty is handled in the most general way which is easy to integrate into a fuzzy classifier. This contributes to our idea of a system wide and easy to engineer trust management in (complex) embedded systems.

6. Conclusion

To summarize, this paper has introduced a concept for extending rule based fuzzy classifiers in order to deal explicitly with dynamic uncertainties at runtime. It is independent of specific kinds of membership functions and inference methods. It hence can be used as an addition to existing fuzzy classifier methods. Its simplicity allows an easy understanding of its effect in order to keep the classification system transparent. The degree of certainty of the classification result expresses the uncertainty implied in the classifier itself as well as the dynamic uncertainty of the current feature vector. Thus it fits well into the trust management framework, where the trustworthiness of the output can be processed further. This enables a trustworthy operation for critical applications because misclassifications are avoided as far as possible and the trustworthiness of a made classification decision can be judged. Nevertheless, the approach is simple, easy to handle and fast to engineer.

In future work we will take a more detailed look at uncertainty induced by the training process. E.g., more detailed information about conflicting rules will be incorporated into the trust level of a result and thus be a valuable contribution towards a overall trustworthy classification.

Acknowledgment

The work has been partly supported by the German Federal Ministry of Economics and Technology, support code KF2312001ED9.

References

[1] A. Avizienis, J.C. Laprie, and B. Randell. Dependability and its threats: a taxonomy. In *Building the information society: IFIP 18th World Computer Congress, Toulouse, France*, pages 91–120. Kluwer Academic Pub, August 2004.

[2] W. Brockmann, A. Buschermöhle, and J. Hülsmann. A Generic Concept to Increase the Robustness of Embedded Systems by Trust Management. In *Proc. Int. Conf. Systems, Man, and Cybernetics*, pages 2037–2044. IEEE, 2010.

[3] J. Hühn and E. Hüllermeier. FR3: A Fuzzy Rule Learner for Inducing Reliable Classifiers. *IEEE Transactions on Fuzzy Systems*, 17(1):138–149, 2009.

[4] L. Gonçalves, C. Fonte, and M. Caetano. Using uncertainty information to combine soft classifications. In *Computational Intelligence for Knowledge-Based Systems Design*, volume 6178 of *Lecture Notes in Computer Science*, pages 455–463. Springer Berlin / Heidelberg, 2010.

[5] T. Denoeux and L.M. Zouhal. Handling possibilistic labels in pattern classification using evidential reasoning. *Fuzzy Sets and Systems*, 122(3):409–424, 2001.

[6] J. Yang and S.R. Gunn. *Knowledge–Based Intelligent Information and Engineering Systems*, chapter Input Uncertainty in Support Vector Machines, pages 148–155. Springer, 2007.

[7] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification.* Wiley-Interscience, 2001.

[8] E. Lughofer. On-line incremental feature weighting in evolving fuzzy classifiers. *Fuzzy Sets and Systems*, 163(1):1 – 23, 2011.

[9] V. Tresp, R. Neuneier, and S. Ahmad. Efficient methods for dealing with missing data in supervised learning. *Advances in Neural Information Processing Systems*, 7:689–696, 1995.

[10] A. Buschermöhle, N. Rosemann, and W. Brockmann. Stable Classification in Environments with Varying Degrees of Uncertainty. In *Proc. Int. Conf. on Computational Intelligence for Modelling Control & Automation*, pages 441–446. IEEE Computer Society, 2008.

[11] R. Jain and A. Abraham. A comparative study of fuzzy classification methods on breast cancer data. *Australasian Physical & Engineering Science in Medicine*, 27:213–218, 2004.

[12] P. Angelov, X. Zhou, D. Filev, and E. Lughofer. Architectures for Evolving Fuzzy Rule-based Classifiers. *Proc. Int. Conf. on Systems, Man and Cybernetics*, 2007.

[13] E. Lughofer, P. Angelov, and X. Zhou. Evolving Single- and Multi-Model Fuzzy Classifiers with FLEXFIS-Class. *Proc. Int. Conf. IEEE on Fuzzy Systems*, pages 363–368, 2007.

[14] J. Roubos, M. Setnes, and J. Abonyi. Learning fuzzy classification rules from data. *Proc. RASC Conference. Leichester, UK*, 2000.

[15] D.J. Newman A. Asuncion. UCI machine learning repository, 2007.