VOL. 10, NO. 3, FEBRUARY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

AN EVIDENCE-BASED APPROACH FOR GPS ACCURACY CLASSIFICATION

Haitham M. Amar[1], Nabil M. Drawil[2] and Otman A. Basir[3]
[1]Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario
[2]Department of Computer Engineering, University of Tripoli, Tripoli, Libya
[3]Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario
E-Mail: hamar@uwaterloo.ca

ABSTRACT

This paper investigates the accuracy of a GPS device. The GPS accuracy is treated as a pattern recognition problem. Each location estimate is classified into a certain accuracy class. Various observation conditions provided by the GPS device are used as features relating a location estimate to an accuracy band.

In this paper we introduce an evidence-based classifier (EBC) in which three independent classifiers are used: namely, feed forward neural network, K-nearest neighbor and the support vector machine. The decisions of these classifiers are combined by a reasoning-based-engine using dempster-shafer (DS) evidence theory for decision fusion. The DS engine will produce the final classification decision. As proof of concept, a comprehensive experimental work including two use-cases is conducted in this paper. Experimental results are discussed at the end of this paper.

Keywords: dempster-shafer, pattern recognition, accuracy classification, basic probability assignment.

1. INTRODUCTION

Global Positioning System (GPS) devices play major role in our daily life. They are widely used whether by individual in daily trips or by more sophisticated applications for threat assessment and collision avoidance purposes. However, GPS devices are error prone devices. There are several types of errors associated with them either caused by internal functionalities or by external effects. The error of the location estimate, depending on the GPS device, ranges from few centimeters to hundreds of meters. This wide range of error may become a source of confusion for applications that wish to rely on certain GPS estimate. An indication of accuracy, thus, became a major demand when processing GPS device's location estimate. The accuracy represents the range of error to which a certain location estimate may belong.

GPS devices use observation conditions to estimate the current location. These observation conditions span various parameters such as the number of satellites, the mean and variance of the signal to noise ratio (SNR) as well as the speed and the dilution of precision. These observation conditions can be treated as features and then used to classify each GPS location estimate so as it belongs to a single class of localization accuracy. The accuracy classes, which represent a range of errors, can be defined linguistically or numerically. Linguistic definitions such as Accurate and Non-accurate should represent a numerical range of errors. The definition of these ranges is an application related problem. Different approaches can be found in the literature to provide a solution for the GPS localization accuracy classification. For e.g., see [1] and [2].

We aim to design an Evidence Based Classifier (EBC) that maps each location estimate into an accuracy class that represents an error band. For the general case, we would have the following classes {High Accuracy, Mild Accuracy, Low Accuracy}. Several classifiers are used to classify each GPS estimate into certain class. Through combination a better result might converge. The combination, in our classifier, is to be achieved through an evidence reasoning combination approach. Dempster and Shafer (DS) theory is one of the most notable combination techniques. In this paper we use it at the core of our combination engine.

Dempster and Shafer (DS) theory [3] is widely used in classification applications. In [4] Hegarat-Mascle et. al. have established that DS theory can be used successfully for unsupervised classification. This success is driven by the ability of DS to integrate the imprecision and uncertainty as part of the classification process. In [5] Bloch used DS's subsets of more than one class to model the impression and uncertainty in the classification of multi-modality medical images. In [6] DS was used to enhance the performance and mitigate the drawback that may occur when using certain classifiers such as K-NN and neural network. In addition, in the context of combining classifiers, DS was used to fuse the outputs of multiple classifiers and produce a fused decision which may lead to a better overall performance for various applications; cf. [7] [8] [9].

Our developed EBC utilizes the confidence each classifier has in its decision. Through DS's evidence fusion rule we can combine the decisions of all three classifiers. Furthermore, it's possible, through an adequate representation of ignorance and belief, to control what types of misclassification error can be avoided. In general, it's more crucial for the classifier to avoid misclassifying a non-accurate location estimate as an accurate than other types of misclassifications.

The remainder of this paper is organized as the following: Section I provides an introduction to the topic tackled in this paper. Section II describes the developed classifier with an emphasis on DS application in this paper and the basic probability assignment. Section III describes

VOL. 10, NO. 3, FEBRUARY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

the experimental work including description of the used dataset, experimental set up, use cases design, results and discussion. This paper concludes with conclusion and remarks on future work in Section IV.

2. EVIDENCE BASED CLASSIFIER (EBC)

The developed evidence based classifier consists of three basic classifiers: Feed Forward Neural Network (FFNN), K-Nearest Neighbor (K-NN) and Support Vector Machine (SVM). These classifiers will process a pattern simultaneously. Once the classification is concluded, the results will be communicated to the DS engine as in Figure-1.

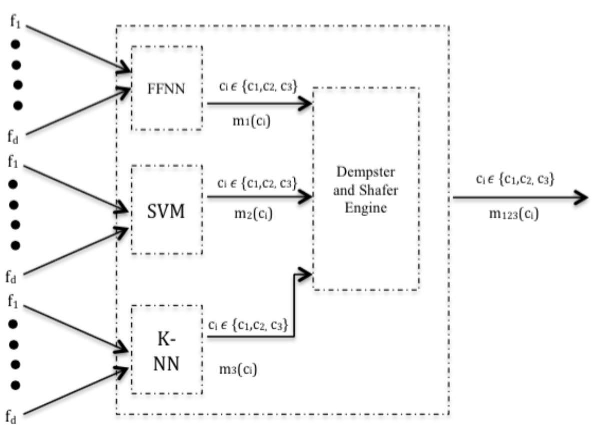


Figure-1. Generic description of the evidence based classifier.

The input of EBC is the set of features that represent each estimate. The classification completion time may vary from one classifier to another. However, if two classifiers produced their classifications, the DS engine should start its fusion process. Next, we describe the design of the DS engine.

A. DS engine

Dempster-shafer theory revolves around combining evidences of observations from independent sources. Each source provides an observation and a mass function (basic probability) related to that observation. Then the DS engine fuses the collected observations based on their masses.

In our developed classifier, we have three sources of information: FFNN, K-NN and SVM. Due to space limitation we will describe DS engine as it pertains to our application. We have

$$\Theta = \{c_1, c_2, c_3\} \quad (1)$$

Where Θ is the frame of discernment and C_1, C_2 and C_3 are the chosen accuracy classes. Let

$$\Omega = 2^{|\Theta|} \quad (2)$$

The power set. Now we have

$$\Omega = \{\emptyset, c_1, c_2, c_3, \{c_1, c_2\}, \{c_1, c_3\}, \{c_2, c_3\}, \{c_1, c_2, c_3\}\} \quad (3)$$

Let's define a mass function m, where

$$m \in [0,1] \quad (4)$$

For independent evidence, focal point with non-zero mass function, we would have the following

$$\sum_{A \subset \Omega} m(A) = 1 \quad (5)$$

m(.) is referred to as basic probability. We have another condition for the subset and that is

$$m(\emptyset) = 0 \quad (6)$$

Now, the DS rule combines the evidences from L classifiers as the following

$$\oplus_1^L m_i(A) = \frac{1}{1-k} \sum_{B_1 \cap B_2 \ldots\ldots \cap B_n = A} m_1(B_1)\ldots.m_n(B_n) \quad (7)$$

Where K is the conflict in the provided evidences and can be calculated as the following

$$K = \sum_{B_1 \cap B_2 \ldots\ldots \cap B_n = \emptyset} m_1(B_1)\ldots.m_n(B_n) \quad (8)$$

$$A, B \subset \Omega \quad (9)$$

Since in our application we have the following:

$$\hat{c}_i \in \{c_1, c_2, c_3\} \quad (10)$$

The remaining subsets in Ω represent the ignorance factor in the observations.

When dealing with systems that rely on DS theory there are few issues that must be resolved in order to get good results. One of the most important issues is to have independent sources of information. For our purpose in this paper, we assume that since our classifiers are of different types then they are independent. Another issue when dealing with DS theory is how to deal with bad sources that have a high confidence in their decision. Describing a classifier as being bad implicitly means that we have a priori knowledge of the classifier's actual performance. This priori knowledge can be used to discount the confidence produced by the said classifier. A third issue with DS is the basic probability, m(.), assignment. Given our classifiers, how can we compute the basic probability of each subset in Ω. Next section discusses our approach for basic probability assignment.

B. Basic probability assignment

In this paper, our approach for basic probability assignment takes into consideration the discounting of our classifiers when computing $m_i(.)$. It, also, considers each classifier's confidence, CC, regarding its classification decision. To discount any classifier, we need to have a priori knowledge of its performance. For each classifier, a confusion matrix computed during the training stage will be used to produce $P(h(pk)|c_i)$. This $P(h(pk)|c_i)$ is used as a discounting factor for each classifier. Each classifier produces the confidence index, CC, during the process of classifying the associate pattern. The process of computing CC differs from one classifier to another and will be discussed in the next section. In general, the basic probability, m(.), is computed as follows:

$$\text{for } c_i \in \{c_1, c_2, c_3\} \quad (11)$$

$$\text{and for } CC \in [0,1] \tag{12}$$

$$m(c_i) = CC * P(h(p_k)/c_i) \tag{13}$$

Where h(.) is the chosen classifier and p_k is the classified pattern. The next section describes the FFNN classifier and how it is used to compute $m_1(c_i)$.

1) Forward neural network: Feed Forward Neural Network (FFNN) has a unique attribute related to its output, which is the activation function. Depending on the activation function, we can range the output between $\dot{y}_i \in \{0,1\}$. One of the most common activation functions used in NN's is the logistic sigmoid function. To estimate the $m_i(c_i)$ we will be replacing CC in Equation (13) by $\dot{y}_i$ and the output is used to compute the belief function m(.) for the primary FFNN classifier and the non-parametric fusion engine, as the following:

$$m(c_i) = \dot{y}_i * P(h(p_k)/c_i) \text{ where } i \in \{1,2,3\} \tag{14}$$

$$m(c_i) = 0 \text{ where } j \neq i \tag{15}$$

$$m(\Theta) = 1 - m_1(c_i) \tag{16}$$

In Equation (16) the complement of the basic probability is assigned to the set Θ. However, this is not a must as it can be assigned to or distributed among other subsets, A_i's. Where $A_i \subset \Omega$ and $|A_i| \geq 2$.

2) K- nearest neighbor: There are various approaches as to how does the K-NN perform its classification. In our application, K-NN classifies each pattern based on its location with respect to its K-closest neighbors. For K-NN the computation of $m_2(c_i)$ is relatively easier than that of the FFNN. It is computed as the following: if K is the number of nearest neighbors to be considered and ζ is the number of votes for certain class then $CC = \frac{\zeta}{K}$ and we have:

$$m_2(c_i) = \frac{\zeta}{K} * P(h(p_k)/c_i) \text{ where } i \in \{1,2,3\} \tag{17}$$

$$m_2(c_i) = 0 \text{ where } j \neq i \tag{18}$$

$$m_2(\Theta) = 1 - m_2(c_i) \tag{19}$$

3) Support vector machine: A non-linear support vector machine can be best described as the solution to the following quadratic equation:

$$f(p_k) = \sum_i \alpha_i * Kr(p_{k_i} * p_k) + b \tag{20}$$

$$\text{Classification decision} = \text{sign}(f(p_k)) \in \{-1,1\} \tag{21}$$

Where Kr is the kernel function, we used quadratic function.

As we can see in Equation (20), the classification is made based on the distance between the examined pattern and the hyper-plane created by support vector machine. The smaller the distance, the more doubtful the result might be. In order to compute $m_3(c_i)$ we need to find a function $\sigma(f(p_k))$ that represent $f(p_k)$ in a way that satisfies Equation (12). An example of a function that can translate $f(p_k)$ into a measure of CC is the sigmoid function. Consider the following equation:

$$\sigma(f(p_k)) = \frac{1}{1+e^{f(p_k)}} \tag{12}$$

The sigmoid function in Equation (22) is simple in a sense that as $f(p_k) \to 0$, $\sigma(f(p_k)) \to 0.5$ which can translate in entropy as complete doubt. That is, p_k could belong to either one of the two classes. However, for DS, we might seek different approach to compute σ. In [10], Platt computes $\sigma(f(p_k))$ as the following:

$$\Pr_{A,B}(y=1 \mid p_k) = \frac{1}{1+e^{A*f(p_k)+B}};\ A \neq 0 \tag{23}$$

There are various method presented in the literature to estimate A and B [10, 11, 12]. In our work, we define $A = -c_i$ and $B = 0$ multiplied with tuning factor η to compute the basic probability for the SVM classifier.

$$0 \leq \eta \leq 1 \tag{24}$$

$$m_3(c_i) = \frac{\eta}{1+e^{-c_i f(p_k)}} * P(h(p_k)/c_i) \text{ where } i \in \{1,2\} \tag{25}$$

$$m_3(c_i) = 0 \text{ where } j \neq i \tag{26}$$

$$m_3(\Theta) = 1 - m_3(c_i) \tag{27}$$

η is empirically computed using the training data to produce an overall good measure of confidence of the SVM classification. We avoid using the term probabilistic to refer to CC as it may require us to adhere to the general axioms of probability, which might not be feasible at this stage.

3. EXPERIMENTAL WORK

A commercial GPS in a roving vehicle was used to collect the data set for this experiment. The GPS devices provide parameters that are related to how the location estimate is made and can be used as features for classification. Five parameters, in particular, were chosen to represent our classes: 1) Dilution of Precision (DOP), 2) Number of Satellites, 3) $\mu_{(SNR)}$, 4) $\sigma_{(SNR)}$, 5) Vehicle speed. The dataset used for our experimental work is thoroughly detailed in [13].

Two use-cases are deployed to prove the efficiency of the developed classifier. For the first use case, the three classifiers are behaving as binary classifiers for the first use-case. The first use case is tackling the problem of classifying our data into two classes, accurate, non-accurate. It gears towards certain applications that tolerate specific accuracy and can provide further processing to mitigate the classification error. Anything beyond that is deemed as a Non-Accurate and not worthy of any further processing. For the second use case, three hierarchical classifiers are used: FFNN, K-NN and SVM. Then the results are introduced to the DS engine that produces a final classification. The second use case tackles the problem of classifying the accuracy into three classes: High accuracy, Mild Accuracy, Low Accuracy. Use-case-2 is geared towards application that can accommodate different accuracy ranges. For instance, applications for message routing in Vehicular Ad-hoc Networks (VANETs), data dissemination and map localization consider the accuracy range of 10-20m, the Mild Accuracy class, acceptable for their implementation requirements [14].

A. Experimental setup

For use-case-1, the training data was partitioned into two regions: region 1, R_1, and region 2, R_2, as defined in Table-I. The classification will decide to which region a pattern belongs and the classification will be concluded.

Table-I. GPS accuracy classes.

Classes	R_1	R_2
Error range	0-15m	>20m

For use-case 2, region 1, R_1, contains the whole of c1, and region 2, R_2 contains c_2 and c_3. All three classes are defined in Table-II. The first level of classification will determine to which region a pattern belongs. If it belongs to R_1, then the classification process is concluded and we have our class. If a pattern belongs to R_2 then we move on to the second level of classification. In this level, a pattern is classified either into c_2 or c_3.

Table-2. GPS accuracy classes.

Classes	C_1	C_2	C_3
Error range	0-10m	10-20m	>20m

The dataset was divided into two parts. Out of 6680 patterns, 65% is used for training and validation. The remaining is used for testing. The training set was divided into 10 folds. 9 folds were used for training and one fold was used to test the generalization of the trained classifier. Next, we discuss the set up of our three classifiers.

1) Feed forward neural network: The neural network uses three layers: the input, the output and the hidden layer. The input layer had five nodes corresponding to the number of features. The hidden layer consisted of 10 nodes. The number of nodes in the output layer was similar to the number of classes for each use case. A tangent sigmoid activation functions was used in all nodes.

2) K-Nearest neighbor: The K-NN classifier used an odd K which was set to 3. To reach a decision about certain class a majority vote was used to determine to which class a test point would belong. To find the closest neighbors, the Euclidean distance was measured between the test point and the training points.

3) Support vector machine: For the SVM, we used soft margin SVM to relax the optimization problem of finding the support vector. For the optimization function the Least Square (LS) was used to find the support vector. For the choice of the kernel function, it was found that the quadratic kernel function would produce the least error rate.

B. Comparative results

The goal is to classify a GPS location estimate into certain accuracy class. We will use several performance assessment measures to assess our proposed classifier. The first measurement is the class accuracy rate A_{Ci} which computed as the following:

$$A_{c_i} = \frac{\#\ \text{of correctly classified patterns}\ \in c_i}{\#\ \text{of patterns}\ \in c_i} \qquad (28)$$

Also, we will be using the Receiver Operating Characteristic (ROC) figure from which we can find the precision and recall of each class. Furthermore, for use-case 2, we will compare our results against results of other models using the same data set as reported in [13] as well as the performance of Ada-boost and forest tree classifiers to indicate how well the designed classifier fair against other well known ensemble of classifiers. Next, we discuss the experimental results for use-case-1.

1) Experimental results for use-case-1: This use case investigates the EBC performance as a binary classifier. EBC's decision $\hat{c}_i$ was made with respect to $\oplus_1^3 m(c_1)$ and $\oplus_1^3 m(c_2)$ as the following:

$$\hat{c}_i = \begin{cases} c_1 \ \text{if} \ \oplus_1^3 m(c_1) > \oplus_1^3 m(c_2), \\ c_2 \qquad\qquad \text{otherwise.} \end{cases} \qquad (29)$$

FFNN, K-NN and SVM classifiers were tested individually to compare their performances with the performance of their fused results using the EBC. The accuracy rates shown in Table-III indicate the classification rate per class for each classifier.

Table-3. Accuracy rate for the second use-case.

Class	$^{A}C_1$	$^{A}C_2$
FFNN	76.2%	95.4%
SVM	75.65 %	95.95%
K-NN	99.9 %	74.32%
EBC	89.13 %	96.13%

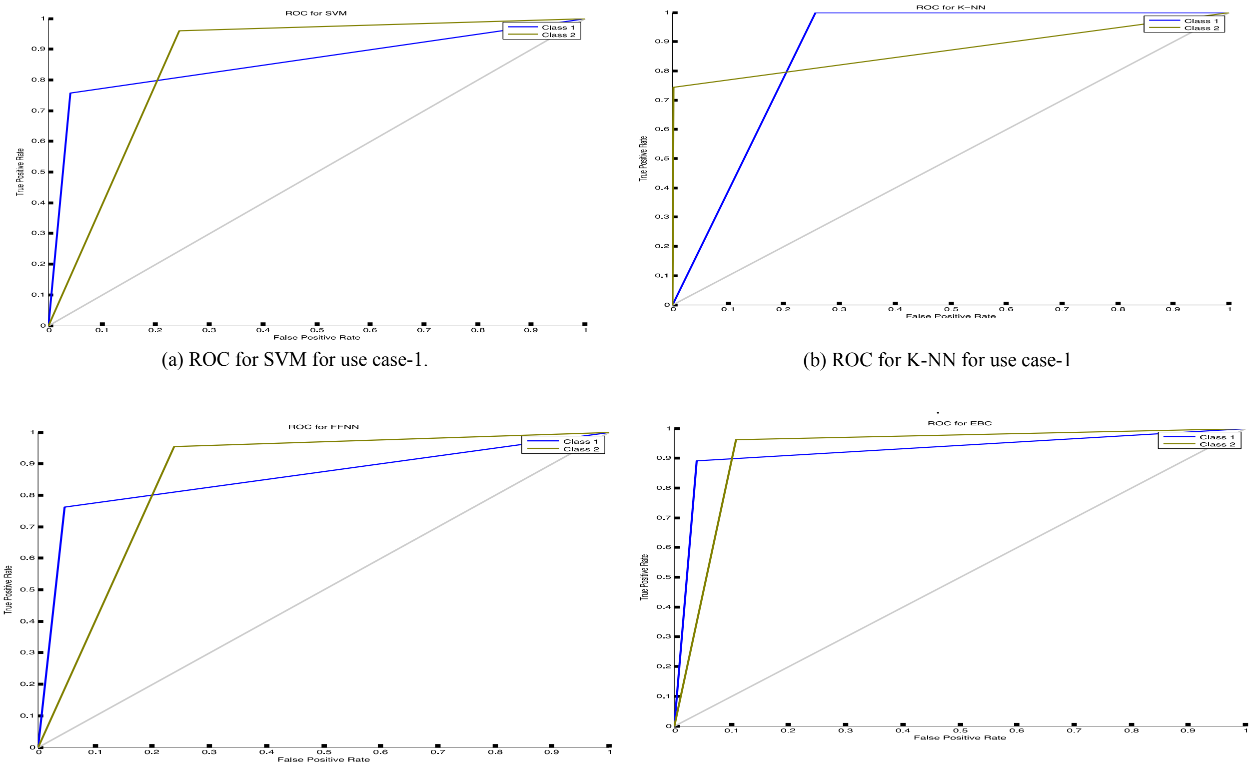
For the use-case-1, we can see the EBC has achieved the best performance for c1 and c_2 with the exception of K-NN's classification of c_1. Figure-2 shows that the performance of EBC is uniform for both classes with most of the patterns that are classified to c_1 do indeed belong to c_1. While for K-NN classifier, even though it

VOL. 10, NO. 3, FEBRUARY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

recognizes almost 100% of the patterns that belonged to c_1, i.e., high recall rate, it has a less than ideal precision rate, less than 80% of what was classified as c_1 is correct. The rest were misclassified patterns from c_2.

2) Experimental results for use-case-2: In this use case, a pattern is classified into one of three classes. First, we discuss the performance of hierarchical FFNN, hierarchical K-NN and hierarchical SVM. Then, we compare their results with the performance of the EBC. EBC's classification was made with regard of $\oplus_1^3 m(c_1), \oplus_1^3 m(c_2)$ and $\oplus_1^3 m(c_3)$ as the following:



(a) ROC for SVM for use case-1.

(b) ROC for K-NN for use case-1

(c) ROC for FFNN for use case-1.

(d) ROC for EBC for use case-1.

Figure-2. Comparison between a variety of ROCs using use-case-1.

$$\hat{c}_i = \begin{cases} c_1 \text{ if } \oplus_1^3 m(c_1) > \oplus_1^3 m(c_2) \,\&\, \oplus_1^3 m(c_3), \\ c_2 \text{ if } \oplus_1^3 m(c_2) > \oplus_1^3 m(c_1) \,\&\, \oplus_1^3 m(c_3), \\ c_3 \qquad \text{otherwise.} \end{cases} \quad (1)$$

As we can see from Table-4, all classifiers were able to detect good percentage of c1. However, their performance was different for c2 and c3. A good performance on one class results in a worse performance for the other class; with adaboostM1 being the worst-case scenario. EBC performed the best and most consistent.

Table-4. Accuracy rate for multiple hierarchical classifiers.

Class	$^A C_1$	$^A C_2$
RBF Network [13]	73.12%	45.99%
Bayesian Network [13]	77.88%	43.5%
Tree-J48 [13]	79.5%	61.2%
HCCU [13]	80.37%	64.25%
CBAC [13]	82.37%	67.37%
AdaboostM1	74.24%	0%
Random Forest	87.4%	28.2%
FFNN	81.28%	38.40%
SVM	81.1%	56.87%
K-NN	99.89%	73.39%
EBC	90.15%	76.31%

Granted that EBC classifier should represent the fusion of the other classifiers' decision, the assumption is that it should outperform them across all classes. Indeed, it outperformed them in c_2 and c_3, but not the same for c_1. The explanation is that, as seen in Figure-3, EBC has a uniform performance across all three classes. The recall is below 0.1 across all classes; especially in c_1, we can see that c_1 has 96% of precision.

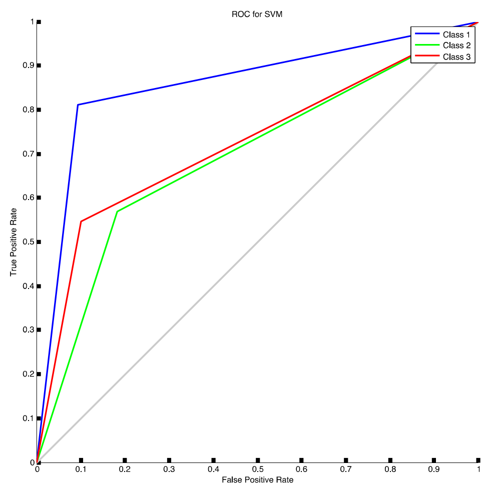
4. CONCLUSIONS

In this paper, several classifiers have been implemented for the GPS localization accuracy problem. This paper has presented an Evidence Based Classifier (EBC) that combines the decisions and evidences from the implemented classifiers into one classification decision using Dempster-Shafer fusion rule. A method to compute the basic probability assignment for each primary

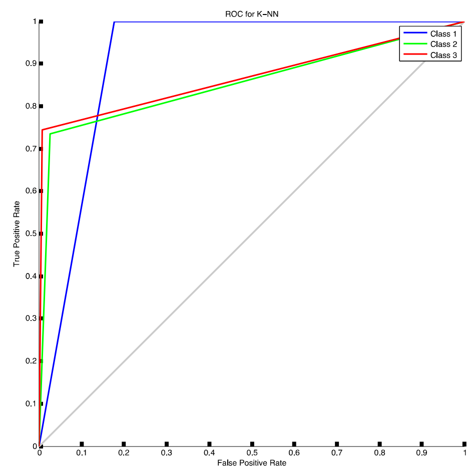
VOL. 10, NO. 3, FEBRUARY 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

classifier was devised for the purpose of DS engine. Two use cases, covering the two-class and the multi-class cases, were used to examine the performance of the various developed classifiers as well as the EBC. The results were plausible and further signified our proposed evidence based classifier.

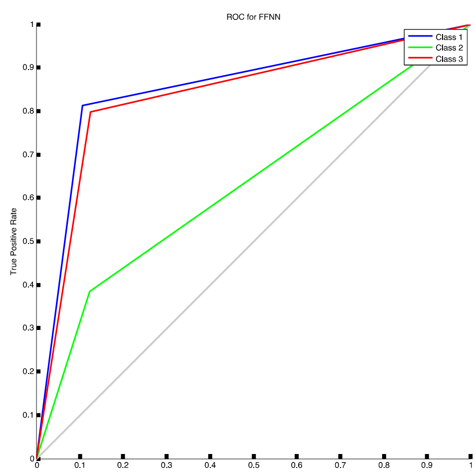
For future work, further investigation on how to the basic probability assignment might lead to improved results. Also, different training procedures may improve the performance of our chosen classifiers; as well as using other classifiers might yield to different results.



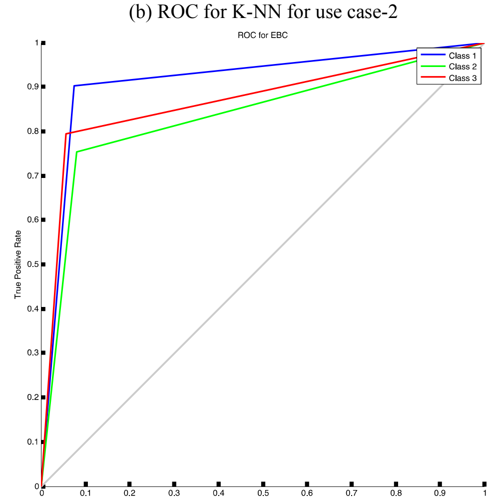
(a) ROC for SVM for use case-2



(b) ROC for K-NN for use case-2



(c) ROC for FFNN for use case-2



(d) ROC for EBC for use case-2

Figure-3. Comparison between a variety of ROCs using use-case-2.

REFERENCES

[1] M. Mosavi, K. Mohammadi, and M. Refan, “Fuzzy processing on gps data to improve positioning accuracy, before and after s/a is turned off,” in The Asian GPS Conference, 2002, pp. 117–120.

[2] C. R. Parikh, M. J. Pont, and N. Barrie Jones, “Application of dempster–shafer theory in condition monitoring applications: a case study,” Pattern Recognition Letters, vol. 22, no. 6, pp. 777–785, 2001.

[3] G. Shafer, A mathematical theory of evidence. Princeton university press Princeton, 1976, vol. 76.

[4] L. Hegarat-Mascle, I. Bloch, D. Vidal-Madjar et al., “Application of dempster-shafer evidence theory to unsupervised classification in multisource remote sensing,” Geoscience and Remote Sensing, IEEE Transactions on, vol. 35, no. 4, pp. 1018–1031, 1997.

[5] I. Bloch, “Some aspects of dempster-shafer evidence theory for classification of multi-modality medical images taking partial volume effect into account,” Pattern Recognition Letters, vol. 17, no. 8, pp. 905–919, 1996.

[6] T. Denœux, “A neural network classifier based on dempster-shafer theory,” IEEE Transactions on Systems, Man, and Cyberneticspart A: Systems and humans, vol. 30, no. 2, p. 131, 2000.

[7] G. Rogova, “Combining the results of several neural network classifiers,” Neural networks, vol. 7, no. 5, pp. 777–781, 1994.

[8] D. Ruta and B. Gabrys, “An overview of classifier fusion methods,” Computing and Information systems, vol. 7, no. 1, pp. 1–10, 2000.

[9] T.-H. Chang, L.-S. Wang, and F.-R. Chang, “A solution to the ill-conditioned gps positioning problem in an urban environment,” Intelligent Transportation Systems, IEEE Transactions on, vol. 10, no. 1, pp. 135–145, 2009.

[10] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in Advances in Large Margin Classifiers. MIT Press, 1999, pp. 61–74.

[11] S. Ruping,¨ “A simple method for estimating conditional probabilities for svms,” Technical Report/Universitat¨ Dortmund, SFB 475 Komplexitatsreduktion¨ in Multivariaten Datenstrukturen, Tech. Rep., 2004.

[12] H.-T. Lin, C.-J. Lin, and R. C. Weng, “A note on platts probabilistic outputs for support vector machines,” Machine learning, vol. 68, no. 3, pp. 267–276, 2007.

[13] Drawil, N.M.; Amar, H.M.; Basir, O.A., "GPS Localization Accuracy Classification: A Context-Based Approach," Intelligent Transportation Systems, IEEE Transactions on , vol.14, no.1, pp.262,273, March 2013 doi: 10.1109/TITS.2012.2213815.

[14] A. Boukerche, H. A. Oliveira, E. F. Nakamura, and A. A. Loureiro, “Vehicular ad hoc networks: A new challenge for localization-based systems,” Computer communications, vol. 31, no. 12, pp. 2838–2849, 2008.