Online Learning for Fuzzy Bayesian Prediction

N.J. Randon[1], J. Lawry[1], and I.D. Cluckie[2]

[1] Artificial Intelligence Group, Department of Engineering Mathematics, University of Bristol, Bristol, BS8 1TR, United Kingdom
Nick.Randon, J.Lawry@bris.ac.uk

[2] Water and Environmental Management Research Centre, Department of Civil Engineering, University of Bristol, Bristol, BS8 1TR, United Kingdom
I.D.Cluckie@bris.ac.uk

Summary. Many complex systems have characteristics which vary over time. Consider for example, the problem of modelling a river as the seasons change or adjusting the setup of a machine as it ages, to enable it to stay within predefined tolerances. In such cases offline learning limits the capability of an algorithm to accurately capture a dynamic system, since it can only base predictions on events that were encountered during the learning process. Model updating is therefore required to allow the model to change over time and to adapt to previously unseen events. In the sequel we introduce an extended version of the fuzzy Bayesian prediction algorithm [6] which learns models incorporating both uncertainty and fuzziness. This extension allows an initial model to be updated as new data becomes available. The potential of this approach will be demonstrated on a real-time flood prediction problem for the River Severn in the UK.

1 Introduction

Many data modelling approaches in Artificial Intelligence (AI) rely on an offline learning strategy where a static model is learned from historical data. This type of modelling is appropriate if the underlying dynamics of the system under consideration does not change over time. However, often this is not the case as the behaviour of a system varies and evolves over time. In this situation an offline learning approach cannot account for these changes unless the model is completely re-learned.

2 Fuzzy Bayesian Methods

The fuzzy Bayesian learning algorithm proposed in Randon and Lawry [5] allows for the induction of prediction models that incorporate both uncertainty and fuzziness within an integrated framework. In the following we give a brief exposition of this approach.

Consider the following formalization of a prediction problem: Given variables $x_1,\ldots,x_{n+1}$ with universes $\Omega_1,\ldots,\Omega_{n+1}$, each corresponding to a compact interval

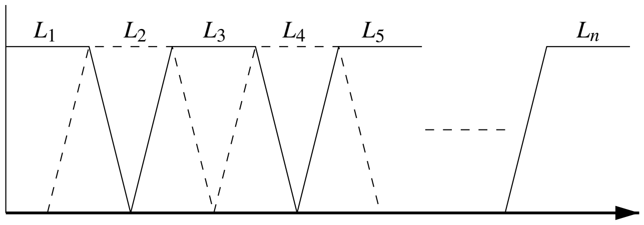

Fig. 1. Trapezoidal fuzzy sets discretizing a continuous universe

of $\mathbb{R}$, suppose that x_{n+1} is dependent on $x_1,\ldots,x_n$ according to some functional mapping $g : \Omega_1 \times \cdots \times \Omega_n \rightarrow \Omega_{n+1}$ (i.e. $x_{n+1} = g(x_1,\ldots,x_n)$). In fuzzy Bayesian algorithms fuzzy labels are used to partition both input and output and probability values are then estimated for the corresponding imprecise regions of attribute space. Figure 1 illustrates how trapezoidal fuzzy sets can be used to discretize a continuous universe. Such a discretization can be generated from a crisp partition where we now identify a fuzzy label L_i with each partition interval (i.e. a bin) such that L_i applies to those elements in the interval to degree 1 and also applies to some points in neighbouring intervals to a non-zero degree. Examples of labels include *small*, *medium*, *large*, *tall*, *high* etc. and in the fuzzy Bayesian model the membership of x in L is interpreted as the probability that L is a valid or appropriate label given value x. i.e.

$$\forall x \in \Omega \;\; \mu_L(x) = P(L|x)$$

Now unlike crisp partitions the labels L_i overlap so that more than one label may be appropriate to describe a particular x value, and hence we cannot directly define probability distributions on the set $LA = \{L_1,\ldots,L_n\}$. Instead, we must base our analysis on the set of atoms generated from $LA = \{L_1,\ldots,L_n\}$, each identifying a possible state of the world and taking the form:

$$\alpha = \bigwedge_{i=1}^{n} \pm L_i \text{ where } +L_i = L_i \text{ and } -L_i = \neg L_i$$

For example, in the case that we have only two labels L_1 and L_2 then there are 4 atoms; $\alpha_1 = L_1 \wedge L_2$, $\alpha_2 = L_1 \wedge \neg L_2$, $\alpha_3 = \neg L_1 \wedge L_2$ and $\alpha_4 = \neg L_1 \wedge \neg L_2$. In general, if we have n labels then there are 2^n atoms, however, if as in figure 1 at most two labels can be applied to any x then only $2n-1$ atoms can have non-zero probability. For example, the atoms generated by the fuzzy labels in figure 1 are shown in figure 2. Let $\mathscr{A}$ denote the set of atoms with non-zero probability for at least some $x \in \Omega$.

For a given $x \in \Omega$ the distribution on atoms $P(\alpha|x) : \alpha \in \mathscr{A}$ can be represented by a mass assignment $m_x : 2^{LA} \rightarrow [0,1]$ on the power set of LA as follows:

$$\forall T \subseteq LA \;\; P(\alpha_T|x) = m_x(T) \text{ where } \alpha_T = \left(\bigwedge_{L\in T} L\right) \wedge \left(\bigwedge_{L\notin T} \neg L\right)$$

For example, if $LA = \{L_1,\ldots,L_n\}$ then:

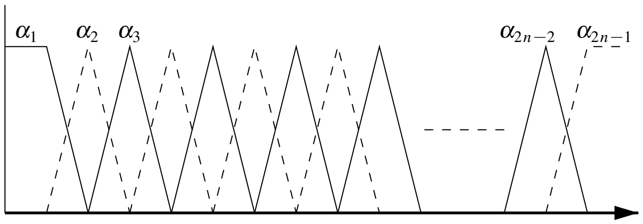

Fig. 2. Probability functions $P(\alpha|x)$ for atoms $\alpha \in \mathscr{A}$ where $\alpha_1 = L_1 \wedge \neg L_2 \wedge \neg L_3 \wedge \cdots \wedge \neg L_n$, $\alpha_2 = L_1 \wedge L_2 \wedge \neg L_3 \wedge \cdots \wedge \neg L_n$, $\alpha_3 = \neg L_1 \wedge L_2 \wedge \neg L_3 \wedge \cdots \wedge \neg L_n$ etc

$$P\left(L_1 \wedge L_2 \wedge \bigwedge_{i=3}^{n} \neg L_i | x\right) = m_x\left(\{L_1, L_2\}\right)$$

Intuitively $m_x(T)$ is the probability that the set of all labels appropriate to describe x is T. A consequence of this translation process is that the current algorithm can be embedded in the label semantics framework as proposed by Lawry ([2] and [3]). Now under certain circumstances label semantics can be functional, allowing a mapping from the fuzzy label definitions $\mu_L(x) : L \in LA$ to the conditional probabilities on atoms $P(\alpha|x) : \alpha \in \mathscr{A}$. One such possibility, as discussed in Lawry [2], is that for each x there is a natural ordering on the appropriateness of labels and that the values of m_x are evaluate so as to be consistent with this ordering. This means that the mass assignment m_x is consonant or nested and consequently can be determined from $\mu_L(x) : L \in LA$ as follows: If $\mu_{L_1}(x) \geq \mu_{L_2}(x) \geq \ldots \geq \mu_{L_n}(x)$ then:

$$P\left(\bigwedge_{j=1}^{n} L_j | x\right) = m_x\left(\{L_1, \ldots, L_n\}\right) = \mu_{L_n}(x)$$

$$P\left(\bigwedge_{j=1}^{i} L_j \wedge \bigwedge_{j=i+1}^{n} \neg L_j | x\right) = m_x\left(\{L_1, \ldots L_i\}\right)$$

$$= \mu_{L_i}(x) - \mu_{L_{i+1}}(x) : i = 1, \ldots, n-1$$

$$P\left(\bigwedge_{j=1}^{n} \neg L_j | x\right) = m_x(\emptyset) = 1 - \mu_{L_1}(x)$$

and $P(\alpha|x) = 0$ for all other atoms α.

In the case where for any $x \in \Omega$ at most two labels have non-zero probability (as in figure 1) then the above consonant mapping is simplified further so that if for a given $x \in \Omega$ $\mu_{L_i}(x) \geq \mu_{L_j}(x) > 0$ and $\mu_{L_k}(x) = 0 : k \notin \{i, j\}$ (for the labels in figure 1 either $j = i + 1$ or $j = i - 1$) then only two atoms have non-zero probability given by:

$$P\left(L_i \wedge L_j \wedge \bigwedge_{k \notin \{i,j\}} \neg L_k | x\right) = m_x\left(\{L_i, L_j\}\right) = \mu_{L_j}(x)$$

$$\text{and } P\left(L_i \wedge \bigwedge_{k \neq i} \neg L_k | x\right) = m_x\left(\{L_i\}\right) = \mu_{L_i}(x) - \mu_{L_j}(x)$$

For the membership functions shown in figure 1 the probability functions on atoms inferred in this way are shown in figure 2 across all values of x.

In the fuzzy naïve Bayes algorithm each input universe is partitioned using trapezoidal fuzzy sets (as in figure 1) and the probability function for the atoms generated as in figure 2. Let $\mathscr{A}_i$ denote the atoms generated for variable x_i where $i = 1, \ldots, n$. Then for output atom $\alpha_{n+1} \in \mathscr{A}_{n+1}$ and input atom $\alpha_j \in \mathscr{A}_j$ we infer the conditional probability $P(\alpha_j|\alpha_{n+1})$ from the training database $DB = \{\langle x_1(i), \ldots, x_n(i), x_{n+1}(i)\rangle : i = 1, \ldots, N\}$ as follows:

$$P(\alpha_j|\alpha_{n+1}) = \frac{\sum_{i \in DB} P(\alpha_j|x_j(i)) P(\alpha_{n+1}|x_{n+1}(i))}{\sum_{i \in DB} P(\alpha_{n+1}|x_{n+1}(i))}$$

From this we can use Jeffrey's rule [1] (an extension of the theorem of total probability) to infer a marginal density conditional on α_{n+1} such that:

$$f(x_j|\alpha_{n+1}) = \sum_{\alpha_j \in \mathscr{A}_j} P(\alpha_j|\alpha_{n+1})\, f(x_j|\alpha_j)$$

Where assuming a noninformative uniform prior distribution on the input universe Ω_j, from Bayes' theorem we have:

$$\forall x_j \in \Omega_j,\ \alpha_j \in \mathscr{A}_j\ f(x_j|\alpha_j) = \frac{P(\alpha_j|x_j)}{\int_{\Omega_j} P(\alpha_j|x_j)\ dx_j}$$

From this we can apply Bayes' theorem together with the naïve Bayes conditional independence assumption as in the standard Bayesian model [4] to obtain the conditional probability $P(\alpha_{n+1}|x_1, \ldots, x_n)$ of each output atom given a vector of input values, as follows:

$$P(\alpha_{n+1}|x_1, \ldots, x_n) = \frac{P(\alpha_{n+1}) \prod_{j=1}^{n} f(x_j|\alpha_{n+1})}{\sum_{\alpha_{n+1} \in \mathscr{A}_{n+1}} P(\alpha_{n+1}) \prod_{j=1}^{n} f(x_j|\alpha_{n+1})}$$

Hence, given an input $\langle x_1, \ldots, x_n\rangle$ we can now obtain a density function on output values using Jeffrey's rule as follows:

$$f(x_{n+1}|x_1, \ldots, x_n) = \sum_{\alpha_{n+1} \in \mathscr{A}_{n+1}} P(\alpha_{n+1}|x_1, \ldots, x_n)\, f(x_{n+1}|\alpha_{n+1})$$

A single output prediction can then be obtained by taking the expected value so that:

$$\hat{x}_{n+1} = \int_{\Omega_{n+1}} x_{n+1} f(x_{n+1}|x_1, \ldots, x_n)\, dx_{n+1}$$

3 Online Updating

In a dynamically changing prediction problem the functional mapping $g_t : \Omega_1 \times \ldots \times \Omega_n \to \Omega_{n+1}$ from input to output variables is time dependent. In this section we introduce a version of the fuzzy naïve Bayes algorithm which learns incrementally in an online manner, updating the prediction model at each step.

Suppose we now receive the data as a series where our current data index is $i-1$. Given a new training example $\mathbf{x}(i) = \langle x_1(i), \ldots, x_n(i), x_{n+1}(i) \rangle$ we update the conditional probabilities for each output atom as follows:

$$P'(\alpha_j|\alpha_{n+1}) = \frac{|\alpha_{n+1}|P(\alpha_j|\alpha_{n+1}) + wP(\alpha_j|x_j(i))P(\alpha_{n+1}|x_{n+1}(i))}{|\alpha_{n+1}| + wP(\alpha_{n+1}|x_{n+1}(i))}$$

Here $P(\alpha_j|\alpha_{n+1})$ is the current probability estimate obtained through updating on the first $i-1$ examples and $P'(\alpha_j|\alpha_{n+1})$ denotes the updated probability taking into account example i .$|\alpha_{n+1}|$ indicates the degree to which output atom α_{n+1} has been previously encountered during learning given by:

$$|\alpha_{n+1}| = \sum_{k=1}^{i-1} P(\alpha_{n+1}|x_{n+1}(k))$$

w is a learning parameter controlling the updating impact of a new training example and is typically assumed to be a decreasing function of $|\alpha_{n+1}|$ with limit 1 (see figure 3). For example, one possibility is $w(|\alpha_{n+1}|) = \frac{c}{|\alpha_{n+1}|} + 1$ where c is a constant controlling the level of initial updating. Note that if $c = 0$ then after updating on all N training examples the conditional probabilities are identical to those obtained using the offline algorithm as described in section 2. In the absense of any data concerning the atom α_{n+1} conditional probabilities are a priori assumed to be uniform so that:

$$P(\alpha_j|\alpha_{n+1}) = \frac{1}{|\mathscr{A}_j|} : \alpha_j \in \mathscr{A}_j$$

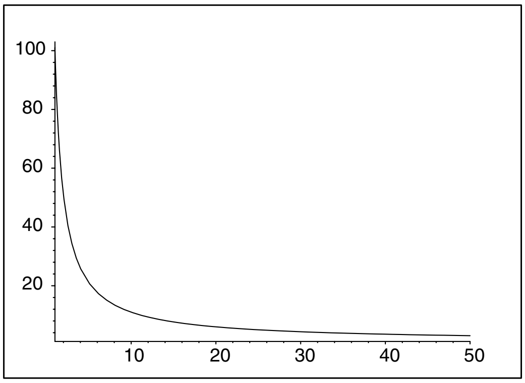

Fig. 3. Weight function $w = \frac{100}{|\alpha_{n+1}|} + 1$

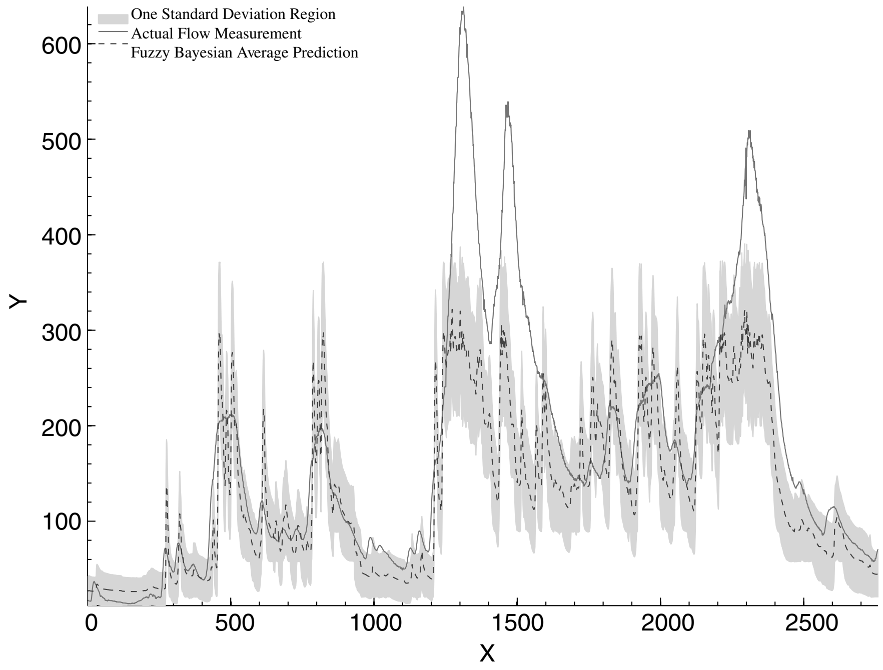

Fig. 4. Fuzzy Bayesian model using offline learning to predict flow at Buildwas 33 hours in advance from upstream flow measurements at Abermule

4 River Flow Modelling

The River Severn is situated in the west of the U.K. and is about 350km long. The source of the river is in the Cambrian Mountains in Wales and its mouth is in the the Bristol Channel. In this paper we look at one section of the River called The Upper Severn which runs from Abermule near Powys, a small village in mid Wales, to Buildwas in Shropshire. The data used in this experiment was taken from flow gauges situated upstream at Abermule which has a catchment area of 580 km^2, and downstream at Buildwas which has a catchment area of 3717 km^2. The flow data for these gauges was obtained from level measurements by applying the rating curve conversion. See [7] for a more detailed description of this catchment.

The offline version of fuzzy naïve Bayes was trained on 1 hourly data consisting of 13119 examples between 01/01/1998 and 01/07/1999. The aim of the learning process was to infer a model to predict flow levels 33 hours ahead [3] at Buildwas from earlier flow data both at Buildwas and upstream at Abermule. Hence, the functional mapping was assumed to take the form $x_{t+33}^B = g(x_t^A, x_t^B)$ where x_t^A and x_t^B denote the flow levels at time t for Abermule and Buildwas respectively. The offline model was then tested on 1 hourly data between 07/09/2000 and 30/12/2000. As well as the actual flow values at Buildwas, figure 4 shows the predicted value for x_{t+33}^B

[3] A 33 hour lead time was selected so as to be consistent with the study reported in [7]

taken as the expected value of the conditional output distribution $f\left(x^B_{t+33}|x^A_t, x^B_t\right)$. Also shown is a region of uncertainty associated with each prediction corresponding to one standard deviation on either side of the mean.

The three large peaks in flow during this period resulted in a major flood event and are not representative of peak flow in the training data. This results in relatively poor performance of the offline algorithm at these peak values. The online algorithm was then applied directly to the year 2000 data, with learning parameter $c = 100$ so that the updating weight function corresponded to $w = \frac{100}{|\alpha_{n+1}|} + 1$ as shown in figure 3. The results for the online learning algorithm on the year 2000 data are then shown in figure 5. Clearly, the overall performance is significantly improved from that of the offline approach, with the first and the third peak being captured, albeit with some time delay. Notice, however, that the second peak is still being significantly underestimated. This may be due to the influence of an upstream tributary not included in the model, but further research is required to resolve this issue.

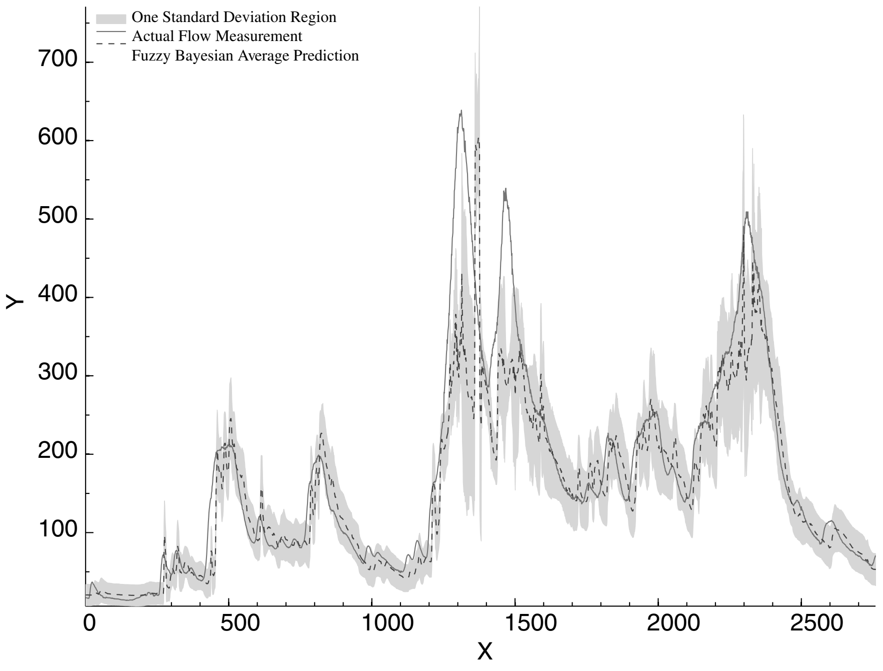

Fig. 5. Fuzzy Bayesian model using online learning to predict flow at Buildwas 33 hours in advance from upstream flow measurements at Abermule

5 Conclusion

An online updating version of the fuzzy naïve Bayes algorithm [5] has been introduced to model systems, the characteristics of which change over time. The potential of this approach has been demonstrated on a river flow modelling problem for the River Severn U.K.

Acknowledgements

This research has been funded by the EPSRC, Flood Risk Management Research Consortium.

References

[1] R. C. Jeffrey, *The Logic of Decision*, Gordon and Breach,New York, 1965.

[2] J. Lawry, A Framework for Linguistic Modelling, *Artificial Intelligence*, Vol. 155, pp 1-39, 2004.

[3] J. Lawry, *Modelling and Reasoning with Vague Concepts*, Springer, 2006.

[4] D.D. Lewis, Naïve (Bayes) at Forty: The Independence Assumption in Information Retrieval, *Machine Learning ECML-98*, LNAI 1398, 1998, pp 4-15.

[5] N.J. Randon, J. Lawry, Linguistic Modelling using a Semi-naïve Bayes Framework, In *Proceedings of IPMU 2002*, pp 1243-1250, 2002.

[6] N.J. Randon, J. Lawry, D. Han, I.D. Cluckie, River Flow Modelling based on Fuzzy Labels, In *Proceedings of IPMU 2004*, 2004.

[7] R.J. Romanowicz, P. C. Young, K. J. Beven, Data Assimilation in the Identification of Flood Inundation Models: Derivation of On-line Multi-step ahead predictions of Flows, In *Hydrology: Science and Practice for the 21st Century*, Vol. 1, pp 348-353, 2004.