Cogn Neurodyn (2014) 8:227–238
DOI 10.1007/s11571-013-9278-5

RESEARCH ARTICLE

A novel approach for pilot error detection using Dynamic Bayesian Networks

Mohamad Saada · Qinggang Meng · Tingwen Huang

Received: 1 February 2013 / Revised: 10 December 2013 / Accepted: 26 December 2013 / Published online: 19 January 2014

Abstract In the last decade Dynamic Bayesian Networks (DBNs) have become one type of the most attractive probabilistic modelling framework extensions of Bayesian Networks (BNs) for working under uncertainties from a temporal perspective. Despite this popularity not many researchers have attempted to study the use of these networks in anomaly detection or the implications of data anomalies on the outcome of such models. An abnormal change in the modelled environment's data at a given time, will cause a trailing chain effect on data of all related environment variables in current and consecutive time slices. Albeit this effect fades with time, it still can have an ill effect on the outcome of such models. In this paper we propose an algorithm for pilot error detection, using DBNs as the modelling framework for learning and detecting anomalous data. We base our experiments on the actions of an aircraft pilot, and a flight simulator is created for running the experiments. The proposed anomaly detection algorithm has achieved good results in detecting pilot errors and effects on the whole system.

Keywords Anomaly detection · Pilot error detection · Dynamic Bayesian Networks · Outlier detection · Machine learning

M. Saada (✉) · Q. Meng
Department of Computer Science, Loughborough University, Loughborough, UK
e-mail: M.Saada@lboro.ac.uk

Q. Meng
e-mail: Q.Meng@lboro.ac.uk

T. Huang
Texas A&M University at Qatar, Doha, Qatar
e-mail: tingwen.huang@qatar.tamu.edu

Introduction

It has been over a century since the Wright brothers made history by building man's first fixed wing controlled heavier than air aeroplane. During that century human kind made a giant leap in the development and use of aeroplanes in many aspects of life, and along with that came aeroplane related disasters, which lead to a great focus on aviation safety measures and protocols.

Aviation disasters started since the first days of aviation, and are still occurring up until this present day, although an enormous amount of effort has been done to prevent these from occurring and to a certain extent it has been very successful, there is still a very long way before preventing further disasters. According to statistics from Kebabjian (2013), there has been over 1085 commercial aeroplane accidents involving fatalities over the past half century, as shown in Fig. 1. There have been many reasons behind these accidents, but they can be generally categorised into a limited number of main causes, including:

- Pilot error related accidents
- Other human error related accidents
- Weather related accidents
- Mechanical failure related accidents

These and other main causes are listed in more detail and by the number of crashes per cause per decade in Table 1. Figure 2 shows that pilot caused errors are the main reason, about 51 % of all these crashes.

Therefore it is crucial to take all possible measures to detect a pilot error at the first time when it occurs before causing a big problem and take the right action to recover if any mistakes made. The method developed in this paper

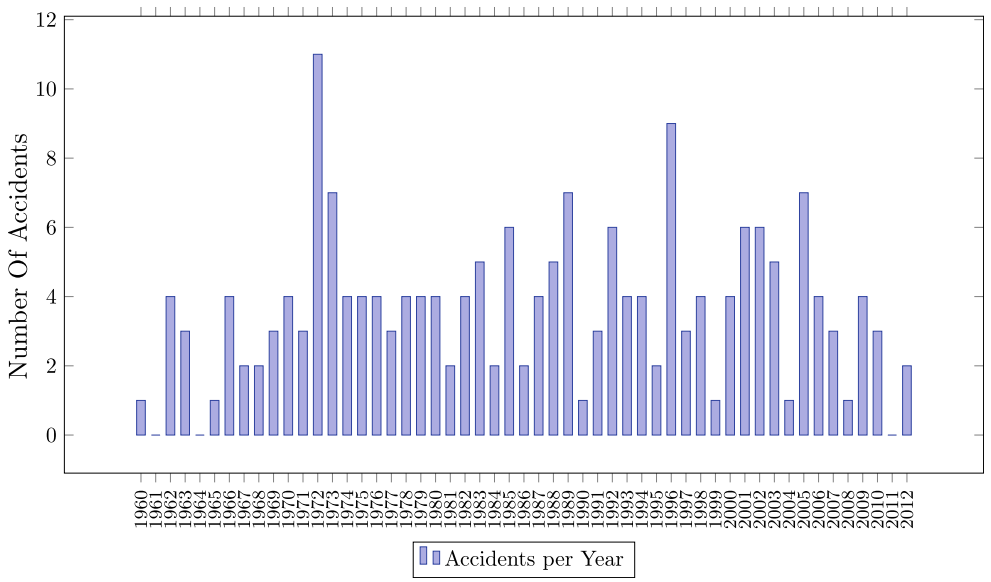
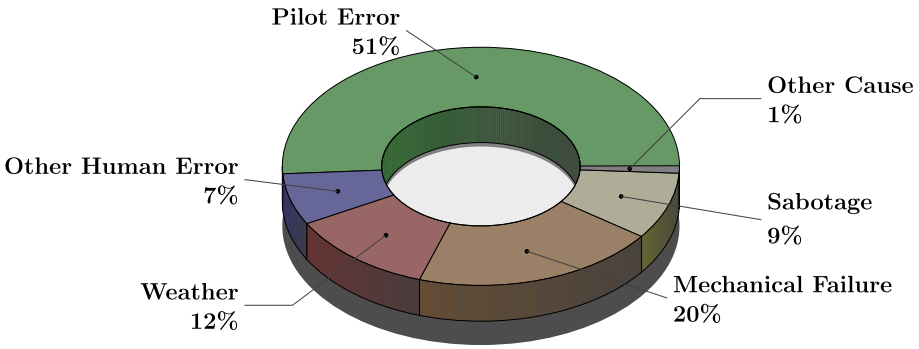


Fig. 1 Number of accidents with 100 or more fatalities by year (Kebabjian 2013)

Table 1 Causes of fatal accidents by decade (Kebabjian 2013)

Cause	1950s (%)	1960s (%)	1970s (%)	1980s (%)	1990s (%)	2000s (%)	All
Pilot error	41	34	24	26	27	30	30
Pilot error (weather related)	10	17	14	18	19	19	16
Pilot error (mechanical related)	6	5	5	2	5	5	5
Total pilot error	57	56	43	46	51	54	51
Other human error	2	9	9	6	9	5	7
Weather	16	9	14	14	10	8	12
Mechanical failure	21	19	20	20	18	24	20
Sabotage	5	5	13	13	11	9	9
Other cause	0	2	1	1	1	0	1

Fig. 2 Flight accident causes



help to build such capability by reasoning the data collected from the flight parameters and pilot actions.

A vast amount of effort has been made towards preventing pilot errors that can have serious consequences, and most of modern day aeroplanes come equipped with many safety and alarm devices that help pilots achieve their tasks as safely as possible. Most airline companies have their own pre-flight check lists which insure everything is implemented according to the latest safety measures, and they also have emergency check lists which pilots use in the case of emergencies to handle the problem in hand.

However, there are still circumstances where pilot actions do lead to accidents, some of these accidents are due to the fact that pilots were not following regulations. But another major cause behind many of the remaining accidents is something called "error chain" or a chain of events that lead to an accident, some of the events in this chain are due to pilot actions but the decisive factors are caused by other unforeseen elements in the environment. An example of such disaster is a recent aeroplane crash in Nigeria, the preliminary accident investigation report (AIB 2013) has shown that the main reason behind this crash was a series of events, including a pilot's action of not deploying flaps during the take-off(which is considered a standard procedure). However this action by itself could not have brought the aeroplane down (AIB 2013), the other events in the chain were that an error in the engines reduced the aeroplane's thrust, and that accompanied with an overly steep rotating or climbing angle, eventually lead the aeroplane into what is known as an aerodynamic stall, ultimately causing the aeroplane to crash into the ground. If the pilot had the knowledge of his action's effects and what set of events caused this stall the outcome might have been very different. This is why we are focusing our efforts on detecting the types of pilot errors that if were combined with other events the result would be an undesirable outcome.

The anomalies that we are investigating in our paper, are those of pilot actions in the context of a flight between two airports, and these types of anomalies are contextual meaning that they are only considered as anomalies or errors when put in a certain context, so the main way for our approach is to detect the effects of these anomalies sometime after their occurrence, and the algorithm that we will discuss later will detect these anomalies through tracing their effects back from the unwanted system state.

This anomaly detection algorithm can help to modify pilot activities in an effort to decrease flight accidents caused by pilot actions. this can be implemented in two ways:

Diagnoses approach A real time method for detecting the cause of current system behaviour, (i.e. what is the most likely sequence of events that lead the aeroplane to the current unwanted state). This could be very helpful in emergency circumstances as it might save valuable time (needed to deal with the emergency itself) through early detection of the causes.

Prognosis approach A real time method that will use approximate inference to predict the state of the flight in the future given all the system observations up until this moment. This method will work with each change in the whole system, it will predict the system state up until a predefined point $t + k$ in the future. If the system state was found to be of an unwanted type with a high level of certainty, the pilot is warned of the possible outcomes of his action and urged to take an alternative action.

There are a number of approaches for detecting anomalies from data which inspire our algorithm development. Many studies have been made just to give a clearer understanding of anomalies, and to answer all the questions around them. For a more in-depth look at data anomaly detection and its variations, see survey papers: (Hodge and Austin 2004; Chandola et al. 2009; Patcha and Park 2007; Markos and Singh 2003a, b; Bakar et al. 2006). For this paper we will only give a brief insight to the major approaches taken to solve anomaly detection.

According to Hodge and Austin (2004) and Chandola et al. (2009), most of the approaches to solving the anomaly detection problem can be categorised into a few main categories:

Statistical approaches

Statistical approaches for anomaly detection mainly use statistical models to model normal data of the environment. Once a model is trained, it is then used to calculate statistical probabilities of a given data instance and label it as normal or anomalous, depending on its likelihood of belonging to the learnt model. This is often done by comparing its probability to a certain threshold. An example of a statistical approach is that of Aggarwal and Yu (2001) where the authors deal with high dimensional data. Their data could have up to hundred of dimensions, and the main focus of their work is to introduce a technique that finds outliers through the behaviour of projections from datasets, as high dimensional data cannot be approached by the regular data proximity algorithms.

Classification approaches

Classification is one of the most popular approaches towards anomaly detection, the basic idea behind classification is to classify data instances and decide whether it's normal or anomalous. The classifier is trained on labelled instances of data. The trained model or classifier is then used to detect anomalies within the unlabelled testing data. Chandola et al. (2009) and Upadhyaya and Singh (2012) summarised various different algorithms and methods in this category. Support vector machines (SVMs) are used to build an intrusion detection system which monitors the access to the Windows registry key (Heller et al. 2003). Bayesian Networks have been used for anomaly detection in a multi-class setting. In the testing phase the posterior probability of the most likely class is calculated, leading to the classification of a normal class, or an anomaly. Das and

Schneider (2007) have addressed the problem of anomaly detection in high arity categorical data, through modelling normal data using a Bayesian Network. The novelty in their approach is that they compare test instances against marginal distribution of attribute subsets. Neural Networks are used as a multi-class or one-class detector. Han et al. (2004) have proposed an intrusion detection technique based on evolutionary neural networks. It takes shorter time to obtain a superior neural network than traditional neural networks, because it learns the structure and weights simultaneously. for more information please refer to (Markos and Singh 2003b; Chandola et al. 2009) survey papers. Zhang et al. (2013a) proposed a method that extracted psychophysiological features to characterise the operators functional state (OFS), then used a Fuzzy c-mean (FSM) algorithm to classify the OFS. this approach is very promising if implemented in the context of detecting the unwanted OFS of a pilot during flight operations. Zhang et al. (2013b) have used multiple psychophysiological and performance measures to build a data-driven model, that is used to estimate the human operator cognitive state(HCS) in a safety-critical human-machine interaction system. They have used an improved sparse least squares support vector machine (LS-SVM) and a Sparse and Weighted one(WLS-SVM) to model the HCS. Both approaches have shown great performance in detecting temporal fluctuation trends of the HCS.

Clustering approaches

Clustering approaches work under different assumptions, it mainly focuses on the idea that normal data instances occur in clusters (these could be large and dense) given a similarity measure whilst anomalous data occur outside these clusters or further away from their centre or in a smaller and less dense clusters. Nearest neighbour is one of the most common clustering approaches. Noh et al. (2006) have proposed a method for network anomaly detection based on clustering sequences of patterns, these patterns represent one TCP network session which is based on the packets of the session.

Not many researchers have used DBN models as basis for anomaly detection. Hill et al. (2007) have developed coupled and uncoupled DBN anomaly detectors which aim to detect erroneous data in two different windspeeds data streams, including single or multiple data streams in real time. Shotwell and Slate (2011) suggested an anomaly detection algorithm using a new implementation of the Dirichlet process precision parameter. Outlier detection is done by calculating a maximum a posteriori (MAP) of the data partition, where observations forming small or singleton clusters are deemed as anomalies. Babbar and Chawla (2010) have used a Bayesian Network to model the outliers as an "unlikely events under the current favored theory of the domain". They used a Bayesian network to model the background knowledge coupled with two rules to detect the outliers. It does not only focus on detecting outliers but also on explaining why these data are considered outliers. Other researchers use an unsupervised approach towards detecting fraud operations in a stock exchange market. (Ferdousi and Maeda 2006) is one of such examples, they use peer group analysis (PGA) technique to characterise the expected pattern of behaviour around the targeted time series financial sequence in terms of the behaviour of similar objects and then detect outliers through analysing the difference in evolution between abnormal behaviour and expected behaviour.

Anomaly detection using Dynamic Bayesian Networks

Dynamic Bayesian Network model

Bayesian Networks are a type of probabilistic models that are based on directed acyclic graphs (DAGs) (Pearl and Russell 2003), the nodes in this model represent propositional variables of interest and the links between them represent the dependencies among these variables. These dependencies are quantified by conditional probabilities of each node given its parents in the network. They have been used extensively by the research community. Tu et al. (2009) introduced an action relationship database (ARDB) structured as a Bayesian Network, which used Bayesian statistics to update its knowledge with new input examples. A model of memory reconsolidation provided the input and predicted relevant activities based on the ARDB.

A Dynamic Bayesian Network is the extension of Bayesian Networks to model probability distributions of sets of random variables over time (Murphy 2002a). Nodes in our DBN model Z_t^k are divided into two sets where t represents the slice number which indicates the time variable, and k is the number of nodes in each slice. The first set contains the hidden state nodes $X_t^n = \{ X_t^1, X_t^2, X_t^3, \ldots, X_t^n \}$, where n represents the number of hidden states in each slice. Hidden states represent immeasurable variables in our model, and these are usually the variables that we aim to gather information about. In Fig. 3 the hidden nodes are the pilot action nodes. And the second set is the set of observable nodes $Y_t^m = \{ Y_t^1, Y_t^2, Y_t^3, \ldots, Y_t^m \}$, where m represents the number of observable nodes in each slice. In Fig. 3 the observable nodes are the aircraft sensor nodes. Observable nodes represent variables that can be measured and are completely or partially observable. These are sometimes called evidence nodes. Note that $n + m = k$ in our model.

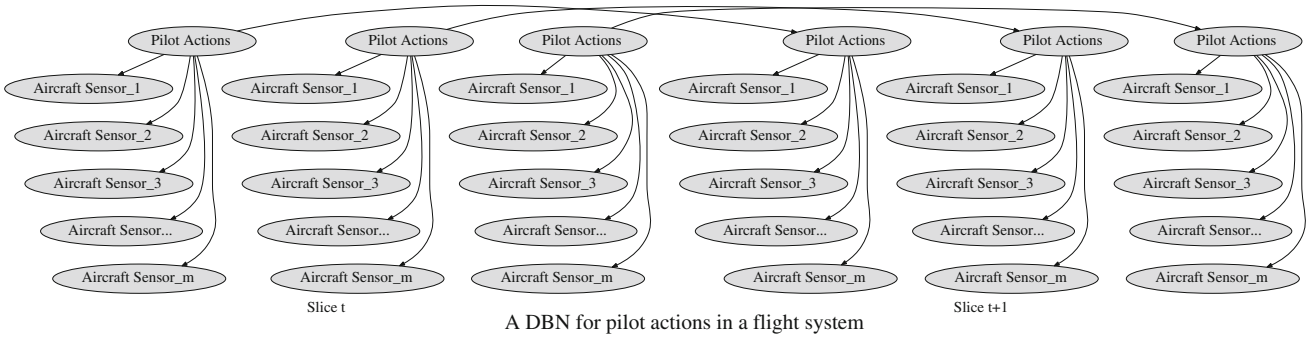


Fig. 3 A Simplified DBN model for pilot actions in a flight system

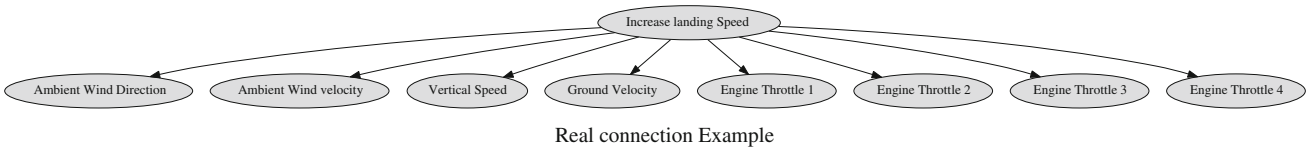


Fig. 4 An actual model connection between one pilot action and different variables in the environment

Before learning of the model, the structure of the network has to be specified. This is done through specifying the parameters of the network, and then the arcs between them which specify the relationship between different variables. The priori information is represented by prior probability distribution over the model's structure and parameters. This is something known as the initial knowledge, in which we use with training data to get a posterior probability distribution over the model and parameters as described in (Ghahramani 1998). We want to compute the maximum likelihood estimate of the parameters given the model and data. Since only partial observability of data is available, the expectation maximisation (EM) algorithm will be used, which works by alternating between two steps to maximise the log likelihood with respect to Q and θ, where Q is some distribution over the hidden variable and θ represents the parameters. The two EM steps (Ghahramani 1998) are:

$$\text{E step} : Q_{k+1} \leftarrow \underset{Q}{\operatorname{argmax}}\, P(X|Y, \theta_k) \quad \text{where} \quad Q \quad \text{is any}$$

distribution over the hidden variables X, Y is the set of observable variables and θ_k represents the parameters at point k.

$$\text{M step} : \theta_{k+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_{X} P(X|Y, \theta_k)\, logP(X|Y, \theta)$$

where θ_{k+1} represent the model's parameters at point $k + 1$, X is the set of hidden variables, Y is the set of observable variables and θ represents the models parameters.

Each DBN slice contains n hidden variable nodes which represent pilot actions, and m observable and measurable nodes which represent different simulation variables, and these are all observable in our model as you can see in Fig. 3. The connections between model nodes are set according to actual relationships between the modelled environment variables see Fig. 4 for an example. Inter slice connections are restricted to hidden nodes. In the DBN model we set a prior probability distribution over the structure and parameters $P(X_1)$, and we learn a state-transition model $P(X_t|X_{t-1})$, and an observation model $P(Y_t|X_t)$ from the data through computing the maximum likelihood estimate over each parameter, this is done through the EM algorithm. The model is limited to first-order Markov:

$$P(X_t|X_{1:t-1}) = P(X_t|X_{t-1})$$

This is primarily done to reduce the complexity of the model and to make all calculation with the number of parameters in the model feasible. The observations are also limited to conditionally first Markov.

$$P(Y_t|Y_{t-1}, X_t) = P(Y_t|X_t)$$

Therefore inter slice relations are only between hidden states in consecutive slices.

After the model is built with different variables in the environment and their relationships, it is trained by the EM algorithm. After training, inference techniques are applied to gather the information needed about hidden variables, including filtering, prediction, classification, control, abduction and smoothing (Murphy 2002a).

Filtering: calculate state over time.

$P(X_t|y_{1:t})$

Prediction: calculate

$P(X_{t+K}|y_{1:t})$ for some point k > 0 into the future

Fixed-lag smoothing: calculate

$P(X_t|y_{1:t})$ i.e., estimating the variable in $m > 0$ slices in the past given all the evidence up-to now

Fixed-interval smoothing: calculate

$P(X_t|y_{1:t})$ This is used as part of training

Viterbi decoding: calculate

$\mathbf{argmax}_{x_{1:t}} P(x_{1:t}|y_{1:t})$ that is finding the most likely explanation

Classification: calculate

$P(y_{1:t}) = \sum_{x_{1:t}} P(x_{1:t}|y_{1:t})$ More detail in (Murphy, 2002a, b, 2012)

Anomaly detection using a DBN

Anomaly detection is the process of detecting patterns in data that do not conform to the expected normal patterns. Anomalies are also referred to as outliers which Hawkins (1980) defines as "an observation that deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism". Our approach to anomaly detection is not based on the typical approach which usually focuses on detecting anomalies in general within data of a given model, instead we take another route. When anomalies occur during the prediction or classification process they often have a ripple-like effect on the descendent states in the same slice and consecutive slices. If the anomaly occurs in one slice, its affect will spread to related states in the same slice and to consecutive slices, albeit the effect is shortly lived and soon all values turn back to normal. So the longer the anomaly occurs, the longer and bigger the effect is. In adaptive online learning models if an anomaly continues to occur for a certain period of time, the model will adapt to it and this anomaly will be then considered to be normal. During the inference of trained models new data is used. Data could be considered as an anomaly due to its value which does not belong to the range of acceptable values of a given variable; or it could have a normal value most of the time, but it is not normal for this value to occur at that point of time. The second type of anomalies could pass undetected by the experts, and thus affecting descendent states. If it continues to occur, it could lead to unexpected values when inference is applied to the model. During take-off if the pilot does not set the flaps to the correct setting, and when accompanied by other unforeseen events, this could lead the aeroplane into a aerodynamic stall. In that scenario the expected value of the vertical speed variable is different from the one recorded as it would be a negative value(opposed to a positive expected value) in the state of an aerodynamic stall. Our algorithm aims to detect this type of data anomaly.

During the inference phase, the model is supplied with a data set containing some anomalies. The anomalies are of an acceptable value but do not occur at the expected time, their effect is propagated to related states in the same and succeeding slices. We suppose that we are able to detect these effects on other state/states Z_t at slice t. Our assumption is that all states Z_t of slice t are observable with known state values. Our objective is to go back trough the slices until we can identify in which states Z_{t-k} an anomaly started to occur which have caused the values of future states to be affected and changed. As we mentioned in the previous section, our DBN model is first-order Markov, and observations are conditionally first-order Markov, this leads to the conclusion that hidden states has an affect only on observable states of the same slice and hidden states in the next slice and are only affected by hidden states of the previous slice. We aim to find the node/set of nodes X^i_{t-k} in slice $t - k$, where k is unknown, that effectively caused a considerable change of value in state Z^j_t in slice t in comparison with the data of the trained model.

Algorithm 1 Detectanomaly(Y)

Require: A set of states $Y = \{Y_1, Y_2, ..., Y_n\}$ representing unwanted flight state observations.

Ensure: A set of states $X = \{X_1, X_2, ..., X_m\}$ representing the anomaly state path.

foreach (Parent Z_t in $Pa(Y_t)$) where t is starting slice. **do**
  Z^j_t.ExpectedValue = ComputeExpValue(Z^j_t, Y_t)
  if (((P(Z^j_t).ExpectedValue) − P(Z^j_t.Value)) > Threshold)&&(($t-1$) >= 0)) **then**
    $X^i_{t-1} = Pa(Z^j_t) \in$ Slice($t-1$)
    foreach (X^i_{t-1}.PossibleValue in X^i_{t-1}.Values) **do**
      X^i_t.ExpectedValue = StateTransitionFunction(X^i_{t-1}, X^i_{t-1}.PossibleValue)
      if (X^i_t.ExpectedValue == Z^j_t.Value) **then**
        AnomalyEffect = true
        TempStateList·*Add*({X^i_{t-1}, X^i_{t-1}.PossibleValue})
      end if
    end for
    if (AnomalyEffect == true) **then**
      foreach (State in TempStateList) **do**
        TempAnomalyList·*Add*(Detectanomaly(State))
      end for
      if (TempAnomalyList ≠ null) **then**
        AnomalyList·*Add*(TempAnomalyList)
      end if
      return AnomalyList
    else
      AnomalyList·*Add*(Z^j_t)
    end if
  else
    return null
  end if
end for
return AnomalyList

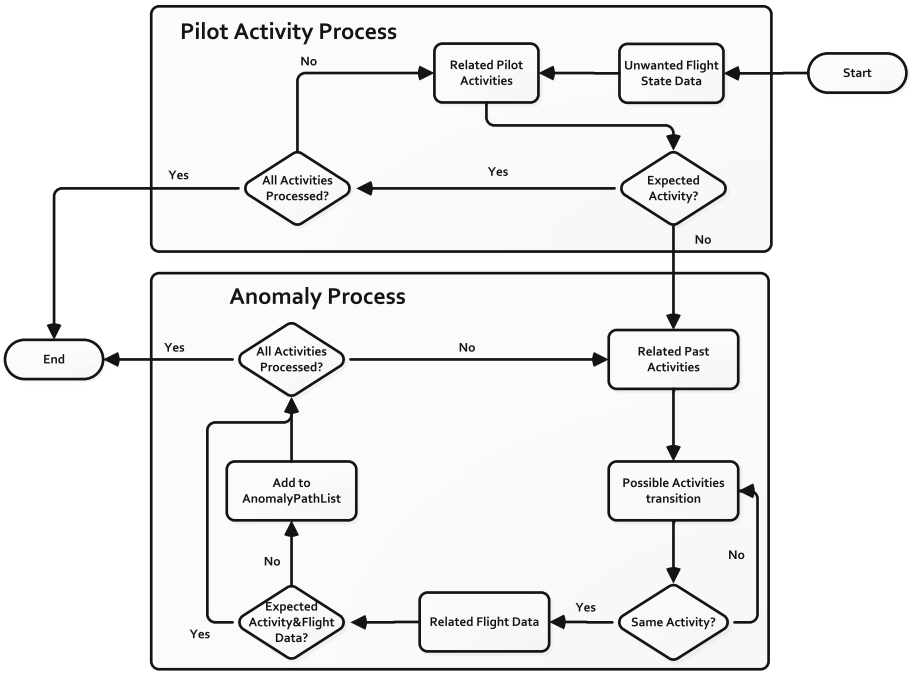


Fig. 5 Flow chart of anomaly detection algorithm

Algorithm 1 is the pseudo code for pilot error detection using DBNs. It takes as input a state/set of states Y where an abnormal value is detected, and produces as output the state or set of states X^i_{i-k} that started an anomaly which caused this abnormal change of value in Y. At first the algorithm retrieve Y related parents in the same slice, which are a state/set of states denoted Z. For every parent state the algorithm calculates the highest probability of any expected value of state Z^j_t at slice t given the trained model.

$$\text{argmax } P(Z^j_t|Y_{1:t}, M)$$

where Y is all observation data and M is the predefined model.

Then this value is compared to probability of the actual value of the state occurring, if there is a large difference between these two values then this data is considered anomalous. Otherwise the algorithm exits.

$$(\text{argmax } P(Z^j_t|Y_{1:t}, M) - P(Z^j_t|Y_t, M)) > Threshold$$

Next step is to go back one slice and to compute the probability of Z^j_t occurring with its current values given all possible values for its parent state and the trained model, this is calculated through the state transition function of the DBN model.

$$P(Z^j_{t-1}|Z^j_t, Y_{1:t}, M)$$

If this is not equal to the expected value at that time slice then the state is added to the anomaly path, otherwise it is considered a normal state. This process is repeated for all parent states as long as the difference in probability between predicted and real values is above the threshold. When this difference drops below the threshold, the state in that slice is considered normal, and the descendent state in next slice is considered as the first anomalous state in the anomalous path, see Fig. 5 for the anomaly detection process.

Experiments and results

Experimental setup and the test scenario

We started by building a DBN model based on a flight scenario, the flight is routed between London Heathrow and East Midlands airports in the UK. The flight duration is 40 min on average. We have used Microsoft® Flight Simulator X as the basis for our simulation, it has an SDK which was used to build our software. The simulator is very realistic and accurate, and it can give us over 1,100 different data variables in high frame rates. We have built our custom software that interacts with the simulator and records all of the flight data online with the desired frame rates.

The first phase of our work was to define the types of anomalies that we want to work with. An anomaly can be a single point anomaly or a list of anomalies. It can be always as an anomaly or just anomalous for some circumstances. We want to base our anomalies on errors in actions of a pilot, as we mentioned before. Figure 6 shows flight data in our simulation combined with accident data

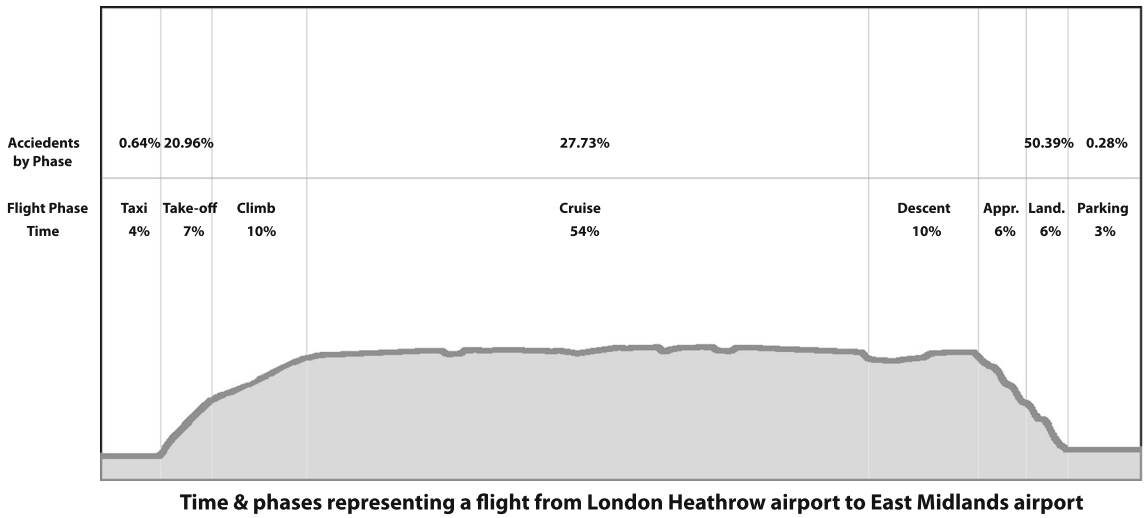


Fig. 6 Accidents by flight phase

from Aircraft Crashes Record Office (ACRO 2012). The figures reveal that most high risk phases of flying are the Landing and taking-off phases. Therefore in this paper we have focused on pilot errors that could happen or affect the flight status within these two phases.

The type of anomalies that we focus on in this paper is that of the contextual type which means they are only considered as anomalies when put in a certain context. Detection techniques for simple point anomalies or errors have already been incorporated into most modern day flight systems, including collective errors. When investigating contextual anomalies in flight related scenarios, we focus on errors that are known as chain errors, or event chain errors, which is a chain of events that can lead to an unwanted state of flight. Typically some of these events are related directly to pilot errors and the rest are related to circumstances that are unforeseen by the pilot, such as the state of mechanical parts of the flight system or even outside events relating to weather or other variables in the environment. Under most circumstances pilot actions would have not lead to an unwanted flight state, but at a given context with a specific chain of events they would cause an undesirable outcome.

The three types of errors that have been chosen in this study including excess speed error, landing gear error and flaps error can lead the aeroplane to an unwanted state. We have focused on accidents caused by an unstable approach. According to FSF (1998) any approach that does not meet any of the certain recommended criteria can be considered as unstable. According to Airbus (2006) Airbus's Flight Operations Briefing Notes, "continuing an unstabilised approach is a causal factor in 40 % of all approach-and-landing accidents" and "In 75 % of the off-runway touchdown, tail strike or runway excursion/overrun accidents, the major cause was an unstable approach.". The same report stated that 66 % of unstabilised approaches are caused by either a "High and/or fast approach or Low and/or slow approach". From this point we have chosen the excess speed parameter as an error type, but this alone is not enough for a type of anomalies, so other events that are unforeseen by the pilot were added, namely a high possibility of a vacuum pump failure and a visual flight rules (VFR) approach rather than a instrument flight rules (IFR) approach was used for landing. Any pilot not taking into account the possibility of a vacuum pump failure and commencing into a high speed VFR landing, could face spatial disorientation in the event of a vacuum pump failure, this combined with such a high speed landing would reach an unwanted outcome. However if a system incorporates maintenance cycles into its model, then a pilot could be warned that one of the parts has a higher than normal chance of failure and with current high speed action, this could lead into an unwanted state. Therefore the high speed approach can be avoided and if the vacuum pump fails it will only be an inconvenience rather than an accident causing event. The same goes to the other two types of errors we chose. The flaps error basically is to take-off with the aeroplane flaps not set to the correct position, this normally can be fine, but when combined with any event disturbing the engines thrust, or/and any event affecting the climbing angle this could lead the aeroplane into something called an aerodynamic stall, which has caused a number of aeroplanes to crash after take-off. As for the landing gear error, it relates the event of an early deployment of a landing gear, in which case the landing gear will be deployed for a longer time. This could lead an unstable approach due to two scenarios, if the plane is in high altitude with certain temperature condition it will

cause the hydraulic system of the gear to be stiff on landing, and if the plane was in low altitude and low speed approach, deploying the landing gear a longer time before it should have, would cause an unnecessary drag because of the landing gear resistance to air, this will cause the aeroplane to descend much faster which could lead to touching down before the intended point.

After defining the anomalies we programmed the simulator to cause an effect on the related variables when these anomalies continued to occur, such as having a rough and bouncy landing when landing gears were kept extended for a longer time than they should before landing, which resemble realistic scenarios. The good point about these types of anomalies is that they are contextual anomalies, meaning that when they occur they can't be detected easily as they represent normal data instances at different circumstances. But at some specific circumstances they occur in, they are considered as anomalies.

The DBN model that we have built is a single layer DBN network. Which consists of two types of nodes: hidden nodes X_t^n representing immeasurable pilot actions which are annotated manually into the training data sets; and observable nodes Y_t^m which represent aeroplane instrumentation data recorded by our software. The observable variables are of a discrete type, including binary nodes with two possible values and nodes with multiple possibilities. You can find some of the variables that are represented by the observable nodes in Table 2. Due to the large number of available simulator variables in the experiment, we had to narrow down the numbers of variables. We have chosen variables which are essential and related to our experiment, check Table 2 to see a few of the flight variables that we have recorded. Fig. 7 illustrates the whole process of the proposed approach.

Table 2 Variables used in the experiments

See (Microsoft® 2008) for the parameter explanation
Aircraft engine data (4)
General engine throttle lever position ENG1
General engine throttle lever position ENG2
General engine throttle lever position ENG3
General engine throttle lever position ENG4
Aircraft position and speed data (6)
Ground velocity
Plane latitude
Plane longitude
Plane altitude
Plane pitch degrees
Plane bank degrees
Aircraft flight instrumentation data (1)
Vertical speed
Aircraft controls data (4)
Rudder position
Elevator position
Aileron position
Flaps handle index
Aircraft landing gear data (1)
Gear position
Aircraft environment data (5)
Ambient density
Ambient temperature
Ambient pressure
Ambient wind velocity
Ambient wind direction

Experiments and results

The model is trained under the consideration that the data is partially observable. Since not all training data sets contain anomalous data, the EM algorithm is used to train the DBN. In our training sets we have introduced three types of errors (excess speed error, landing gear error, flaps error), each one occurring 25 times, and the remaining 75 training sets have no errors. Each one of the errors introduced has its own effect. Our algorithm starts working on the slice where the effects appeared rather than the slice where the error began.

In the testing phase we record 99 new data sets with the same types of errors we have introduced, with each error type occurring in 33 different data sets. Note that these data sets do not contain annotated pilot actions, therefore when the algorithm begins data of the observable variables is fetched from the testing data set; whilst data of unobservable variables are entered manually through an annotation step done before running the algorithm.

we start by training the model gradually and at the same time testing the trained model with the testing data including all the three types of selected anomalies. We continue adding normal and anomalous training sets to the training phase and the detection successful rate increase gradually until algorithm reaches an overall detection rate of 90.9 %. The data sets were divided into two equal halves of normal and anomalous data sets when possible. The first learning experiment was on 15 data sets 8 of which were normal and 7 anomalous. The second was on 30 data sets divided into 15 normal and 15 anomalous. This process was repeated in increments of 15 data sets at each step, those 15 were always divided in half between normal and anomalous, either 8 normal and 7 anomalous or vice versa. steps are repeated until a total of 150 training data sets is reached.

Once the algorithm is trained on all 150 data sets, it was tested on all testing sets containing each error by turn. So

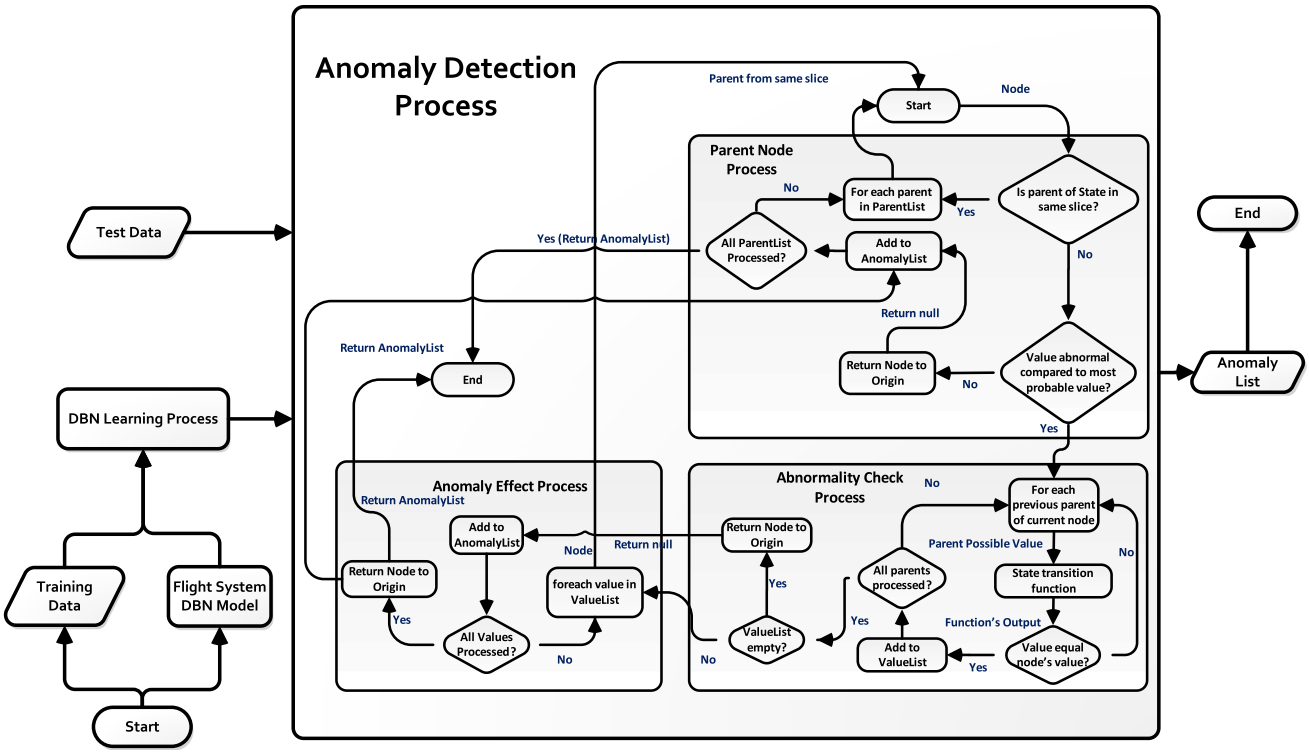


Fig. 7 Flow chart of the whole process

Table 3 Undetected anomalies for each error type (3 excess speed, 4 landing gear and 2 flaps), along with the percentage of how much was correctly identified from the path to the real anomaly

Excess speed error (%)	
1st undetected excess speed err anomaly	90.4
2nd undetected excess speed err anomaly	82.5
3rd undetected excess speed err anomaly	66.5
Landing gear error (%)	
1st undetected landing gear err anomaly	86.6
2nd undetected landing gear err anomaly	92.4
3rd undetected landing gear err anomaly	86.6
4th undetected landing gear err anomaly	89.6
Flaps error (%)	
1st undetected flaps err anomaly	84.3
2nd undetected flaps err anomaly	91.5

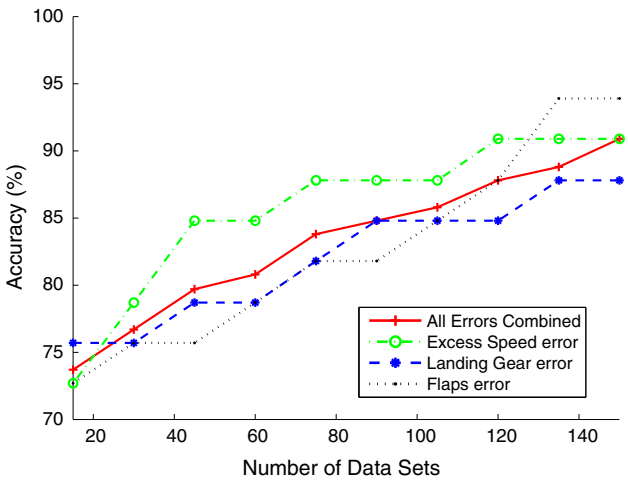


Fig. 8 Error results combined

we began with the (excess speed error), we run the algorithm on all 33 test data sets for this type of error, at the end the algorithm detects 30 out of 33 anomalies, which represent a 90.9 % detection rate. The same is repeated again for both landing gear and flaps errors, the algorithm detects 29 and 31 out of 33 respectively, which amounts to 87.9 and 93.9 % respectively. Table 3 shows the undetected anomalies (3 excess speed, 4 landing gear and 2 flaps). The algorithm stopped the detection process earlier at a wrong state in all of these cases. Which meant all of theses states were considered as the source of the anomaly, when in fact they were not. Therefore all of these detections were considered as false positives. Table 3 shows the percentage of the path to the real anomaly state that was identified correctly before the algorithm stopped at a wrong state. On most non-detections, the algorithm has recognised a large part of the anomaly path correctly.

The overall anomaly detection accuracy rate for the whole experiment is 90.9 % with a confidence range of ±3 %, as shown in Fig. 8.

Discussions and conclusions

The proposed anomaly algorithm is for a contextual type of pilot errors. In the following we analyse the advantages and disadvantages of our algorithm in comparison with other main types of anomaly detection algorithms, including classifications based algorithms, clustering based algorithms, nearest neighbour based algorithms and information theoretic based algorithms.

The outcome of the anomaly detection process comes with a probability which can give an indication of how certain the algorithm of the detection result, which most classification based, cluster based and nearest neighbour based algorithms cannot provide, they only classify the results into anomalous or normal without giving any other useful information.

Another main advantage is that there aren't many algorithms implemented for detecting anomalies with respect to the temporal dimension, which DBN can deal with.

The main disadvantage is like many other classification and statistical based algorithms, our algorithms needs a considerable amount of time for the training phase, but this does not affect its deployability in online scenarios since the testing phase is done in real time. Another accompanying disadvantage is that the training phase requires a large number of data instances.

Our approach is for a contextual type abnormality detection, which means it needs an unwanted flight variables to be present in order for the algorithm to start the search, unlike clustering algorithms which can work unsupervised.

In this paper we focus on detecting data anomalies in a DBN model. A novel algorithm to detect data anomalies has been proposed through backtracking steps of its effect on descendent states until a data anomaly is reached and detected. A DBN model have been built based on pilot actions and instrument data of a flight scenario. The experimental results show its robustness in detecting data anomalies that affect other future states in the model.

References

ACRO (2012) Aircraft crashes record office. http://www.baaa-acro.com/general-statistics/

Aggarwal CC, Yu PS (2001) Outlier detection for high dimensional data. In: Proceedings of the 2001 ACM SIGMOD international conference on Management of data, SIGMOD '01, ACM, New York, pp 37–46

AIB (2013) Preliminary report on accident involving associated airline embraer 120 aircraft registered 5n-bjy which occurred at mma on thursday 3rd october, 2013. Technical report, accident investigation Bureau, Nigeria. http://aib.gov.ng/reports/EMB120%20ACCIDENT%20update.docx-1.pdf.

Airbus (2006) Approach techniques—flying stabilized approaches. Technical report, AIRBUS. http://www.airbus.com/fileadmin/media_gallery/files/safety_library_items/AirbusSafetyLib_-FLT_OPS-APPR-SEQ01.pdf.

Babbar S, Chawla S (December 2010) On Bayesian Network and outlier detection. In: Proceedings of the 16th international conference on management of data, Nagpur, India

Bakar ZA, Mohemad R, Ahmad A, Deris MM (June 2006) A comparative study for outlier detection techniques in data mining. In: 2006 IEEE conference on cybernetics and intelligent systems, pp 1–6

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41:15:1–15:58

Das K, Schneider J (2007) Detecting anomalous records in categorical datasets. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '07, 2007. ACM, New York, pp 220–229

Ferdousi Z, Maeda A (2006) Unsupervised outlier detection in time series data. In: Proceedings of the 22nd international conference on data engineering workshops, April 2006, Atlanta, GA, USA, pp x121–x121

FSF (1998) Approach and landing accident reduction briefing note 7.1—stabilized approach. http://www.skybrary.aero/bookshelf/books/864.pdf

Ghahramani Z (1998) Learning dynamic Bayesian networks. In: Giles CL, Gori M (eds) Adaptive processing of sequences and data structures. Lecture notes in computer science, vol 1387. Springer, Berlin, pp 168–197

Han S-J, Kim K-J, Cho S-B (2004) Evolutionary learning program's behavior in neural networks for anomaly detection. In: Pal NR, Kasabov N, Mudi RK, Pal S, Parui SK (eds) Neural information processing. Lecture notes in computer science, vol 3316. Springer, Berlin, pp 236–241

Hawkins DM (1980) Identification of outliers. Chapman and Hall, New York

Heller KA, Svore KM, Keromytis AD, Stolfo SJ (2003) One class support vector machines for detecting anomalous windows registry accesses. In: Proceedings of the workshop on data mining for computer security, 2003

Hill DJ, Minsker BS, Amir E (July 2007) Real-time bayesian anomaly detection for environmental sensor data. In: Proceedings of the 32nd conference of the international association of hydraulic engineering and research, Venice, Italy

Hodge VJ, Austin J (2004) A survey of outlier detection methodologies. Artif Intell Rev 22:85–126

Kebabjian R (2013) Plane crash info website. http://planecrashinfo.com/cause.htm

Markos M, Singh S (2003a) Novelty detection: a review—part 1: statistical approaches. Signal Process 83(12):2481–2497

Markos M, Singh S (2003b) Novelty detection: a review—part 2: neural network based approaches. Signal Process 83(12):2499–2521

Microsoft® (2008) Microsoft esp 1.0—simulation variables. http://msdn.microsoft.com/en-us/library/cc526981.aspx

Murphy KP (2002a) Dynamic bayesian networks: representation, inference and learning. PhD thesis, University of California, Berkeley

Murphy KP (November 2002b) Dynamic bayesian networks. http://www.cs.ubc.ca/~murphyk/Papers/dbnchapter.pdf

Murphy KP (2012) Machine learning: a probabilistic perspective (adaptive computation and machine learning series). The MIT Press, New York, August 2012. ISBN: 0262018020

Noh S-K , Kim Y-M, Kim DK, Noh B-N (2006) Network anomaly detection based on clustering of sequence patterns. In: Computational science and its applications—ICCSA 2006, volume 3981 of lecture notes in computer science. Springer, Berlin, pp 349–358

Patcha A, Park J-M (2007) An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput Netw 51(12):3448–3470

Pearl J, Russell S (2003) Bayesian networks. In: Arbib MA (ed) The handbook of brain theory and neural networks, 2nd edn. MIT Press, New York

Shotwell MS, Slate EH (2011) Bayesian outlier detection with dirichlet process mixtures. Bayesian Anal 4:665–690

Tu K, Cooper DG, Siegelmann HT (2009) Memory reconsolidation for natural language processing. Cogn Neurodyn 3(4):365–372

Upadhyaya S, Singh K (2012) Classification based outlier detection techniques. Int J Comput Trends Technol 3(2):294–298

Zhang J-H, Peng X-D, Liu H, Raisch J, Wang R-B (2013a) Classifying human operator functional state based on electrophysiological and performance measures and fuzzy clustering method. Cogn Neurodyn 7(6):477–494

Zhang J-H, Qin P-P, Raisch J, Wang R-B (2013b) Predictive modeling of human operator cognitive state via sparse and robust support vector machines. Cogn Neurodyn 7(5): 395–407