GEMM-eMFIS (FRI/E): A Novel General Episodic Memory Mechanism For Fuzzy Neural Networks

Sheng Wei Pang
Sch. of Computer Science & Engg.
Nanyang Technological University
Singapore, Singapore 639798

Chai Quek
Sch. of Computer Science & Engg.
Nanyang Technological University
Singapore, Singapore 639798
ashcquek@ntu.edu.sg

Dilip K. Prasad
Dept. of Computer Science
UiT The Arctic University of Norway
Tromso, Norway 9006
dilip.prasad@uit.no

***Abstract*— In fields such as finance, medicine, engineering, and science, making real-time predictions during transient periods characterized by sudden and large changes is a hard challenge for machine learning. Humans keep memory of these transient events, abstractly learn the most relevant rules and reuse them when similar events occur, which stems from episodic memory that allows storage and recall of similar events. This paper proposes a novel online general episodic memory mechanism (GEMM) and demonstrates its integration into the Neuro-Fuzzy system (NFS) architecture called evolving Mamdani Fuzzy Inference System (eMFIS) with Fuzzy Rule Interpolation and Extrapolation (FRI/E). Our proposition, called GEMM-eMFIS(FRI/E), learns from past events by storing and retrieving them from an episodic memory cache during event-driven transient behavior, thereby boosting performance while using a few rules only. GEMM-eMFIS(FRI/E) further has several in-built mechanisms that enable it to learn effectively from continuous stream of online data. They include associative-dissociative learning theory to keep its rule base updated, 2-stage incremental clustering; (2-SIC) to determine cluster width, interpolation and extrapolation of rules to deal with concept shifts and drifts in the time-variant data, and rule pruning and merging to keep the rule base compact. GEMM-eMFIS (FRI/E) is benchmarked against other NFS' on various time-variant datasets such as stock index prices and rainfall runoff with 3%-5% improvement during transient period and shows strong forecasting performances with 4%-5% more interpretability with lesser rules.**

Keywords*— *Fuzzy Neural Networks, Episodic Memory, Inference Systems, Financial Forecasting

I. Introduction

One of the hardest challenges for machine learning of any predictive model is to make predictions during periods of sudden and larger than normal spike or drop in the values. This phenomenon is known as transient behavior, which is often defined to be a system's response to a change from a steady state [1] due to any event that affects the system's equilibrium. Take the example of a stock's price suddenly crashing upon its release of corporate events. Initially, we see that the price increases steadily, this is known as the steady state. Macro-economic events such as global recession or trade war as well as company related events such as corporate earnings release of adverse reaction of investors to breaking news can tip the equilibrium off and trigger a transient behavior. One type of predictive model that performs well in the domain of finance are Neuro-Fuzzy Systems (NFS). NFS combine the best traits of both architectures, i.e. accuracy of prediction from ANN and derivation of highly interpretable rule sets from FIS to result into interpretable and accurate learning models [3]. Most works on NFS cope with concept drifts [2] in the dynamic data using techniques like fuzzy rule interpolation-extrapolation and incremental clustering techniques. Concept drift occurs when the statistical characteristics of the features and target evolve over time in unexpected ways. However, they struggle to deal with sudden spikes or drops in the data because these incremental techniques assume that all the rule antecedents activated have the same significance, which results in inaccurate interpolation outcomes [4].

In this work, a novel approach of mimicking general episodic memory mechanism (GEMM) of human brain is used to detect transient behavior and perform prediction using suitable rules during the transient period in the event-driven financial domain. Episodic memory cache is implemented and integrated in FIS for the first time. It encodes, stores and retrieves the relevant 'events and rules' that the model has previously experienced, in order to simultaneously improve model accuracy while using a fewer number of rules. For this purpose, eMFIS(FRI/E) is chosen as the foundational FIS on which GEMM is integrated. Here, the choice of eMFIS(FRI/E) deserves an explanation. It preserves the interpretability of Mamdani type fuzzy membership functions even while providing accuracy similar to that supported by TSK type fuzzy functions. It adopts both compositional rule of inference (CRI) inference and the interpolation and extrapolation [3] from the n-nearest rules, which in known for good handling of concept drifts and shifts. In addition, eMFIS (FRI/E) also adopts a self-reorganizing network approach using BCM theory to self-reorganize the rules' potentials in the fuzzy rule base. Its fuzzy clusters have the ability to self-reorganize, ensuring the optimum representation of recent data distribution.The paper is organized in 4 sections. Section II presents GEMM - eMFIS(FRI/E). Section III presents several experiments for benchmarking, financial modeling and analysis of results. Section IV presents conclusion.

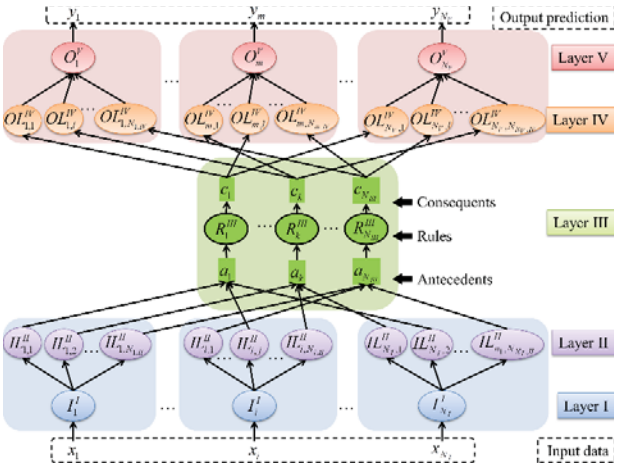


Fig. 1. Network architecture of GEMM-eMFIS (FRI/E) is derived from eMFIS(FRI/E).

II. Incorporating Episodic Memory Into eMFIS(FRIE)

A. Neuro-fuzzy architecture of GEMM-eMFIS(FRI/E)

The basic architecture is adapted from eMFIS(FRI/E)[3]. As an evolving and self-organizing network, GEMM-eMFIS (FRI/E) begins with no neurons or links initially. The neurons and links are incrementally created as knowledge is learnt from the arriving data. This mimics human cognition which incrementally creates schema via accommodation when existing schemas become obsolete in representing the new information. GEMM-eMFIS (FRIE) follows a five layer structure as described in Fig. 1. At the ith time step, the data is defined as the ordered pair (x_i, d_i) of input data values $x_i = [x_1, x_2, \dots, x_p]^T$ and the target data value $d_i = [d_1, d_2, \dots, d_p]^T$. Here, x_p and d_p are the pth attributes of the input and output, respectively. The mathematical details of the architecture can be found in [4] and are left here for brevity.

GEMM-eMFIS (FRIE-M) adopts BCM theory [5] which self-reorganizes its rule base to ensure that its fuzzy rules are representative of the current data. The best fit fuzzy labels are computed for every data value pair (x_p , d_p) which yields the highest membership value. Fig. 2 shows our episodic mechanism acting as a form of meta-learning on top of the inference mechanisms of eMFIS(FRIE). This means that all recalled rules undergo the same inference mechanism with rule pruning and merging. Moreover, it also implies that our episodic mechanism can be plugged into any online neuro-fuzzy model with a dynamic rule base.

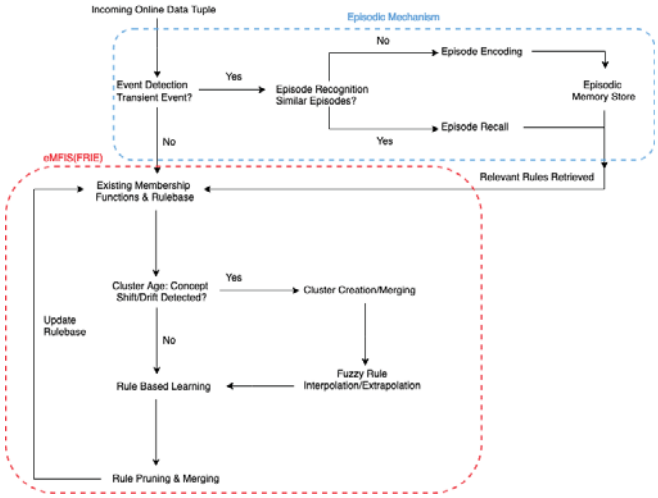


Fig. 2. Episodic mechanism integrated with eMFIS(FRIE)'s learning mechanisms.

The creation of a new cluster occurs when drift or shift is detected by the movement of data distribution to an unknown location. This is due to the second derivatives of the age curve becoming significant, implying that the clustering age is increasing at an increasing rate. Inspired by the categorical learning of infants and adults and adapted from [6], eMFIS(FRIE) adopts 2-SIC which uses both top-down divisive clustering approach [7], and bottom-up clustering approach [8]. It is able to overcome the issues of (1) over-smoothing masking the structure of the data distribution and (2) too many small clusters generated due to a smaller slope.

The interpolation and extrapolation approach, adapted from [9], works by building a new inference rule from its nearest n (n ≥ 2) rules, before applying scale and move transformations to derive the final results. The advantages of this approach allow it to involve multiple rules, each consisting of multiple antecedents, handle fuzzy sets with vertically sloped membership functions, and extrapolate the n-closest rules should they all lie on one side of given observation. This is also applied to the newly recalled rules, giving the system flexibility to interpolate on a larger rule base and likelihood of increasing accuracy. It involves choosing the closest n rules, constructing the intermediate rule and applying a scale and move transformation.

A pseudo-pruning mechanism is an online state-selection mechanism to put some prioritization on the fuzzy rules. Rules are set to off state if the weight falls below certain threshold. However, the rule is only removed when it has been set to off for a significant period of time, or it has a very small weight. Only the rule with higher weight is kept if the antecedents match the new rule, but consequents do not. In the case of duplicate rules, the duplicated rule is removed. The weights of the rules are normalized to the value of [0-1] in order to prevent newly created rule to always have highest weight, and results in overly biased toward new rules. The newly recalled rules are also subjected to removal should their weight fall below a certain threshold or are duplicates.

B. Fuzzy Rule Episodic Cache Mechanism

The GEMM comprises of a multi-store mechanism, event detection, episode encoding, episode recognition, and episode recall. Each of them is discussed below.

Multi-Store Model: The multi-store model consists of three types of memories, namely sensory, short term, and long-memory. Sensory memory is where any observed information is first registered. This is analogous to each input tuple of the data registered by the model. Short-term memory refers to active memory capable of holding small amounts of information for a brief-period of time. This is analogous to the dynamic rule base and membership functions, as rules get created and pruned over time. Long-term memory can store large quantity of information for a very long duration, analogous to the episodic cache introduced here. Let us represent them by the following notation:

Overall memory	$M_t = Z_t + S_t + L_t$	(2.1)
Sensory memory	$Z_t = [d_1, d_2, \dots d_i]$	(2.3)
Short-term memory	$S_t = [x_1, x_2, \dots x_j]$	(2.4)
Long-term memory	$L_t = [m_1, m_2, \dots m_k]$	(2.5)

where $Z_t$contains the vector d of inputs at time t, $S_t$contains the i-th memory x_i computed at time t, and L_t comprises of m_i, i.e. the i-th memory in a long-term memory store. It is noted that each memory type at time t is made up of a series of memories. Given our interests in predicting event-driven trading, we shall focus on exploring use of explicit long term memory to recall the events leading up to financial crises, which would be useful in predicting transient behavior. In the short-term memory, a series of real-time computations take place to make sense of the events and the associated contextual information that occurs over a short period of time. We may further represent the different types of information x_i stored short term memory as:

$e_i = [h_1, h_2, \dots h_m]$	(3.1)
$w_i = \{e_i, e_{i-1}, \dots e_{i-j}\}$	(3.2)
$v_i = (p_i - p_{i-1})/p_{i-1}$	(3.3)
$c_i = \{v_i, v_{i-1}, \dots v_{i-j-1}\}$	(3.4)
$b_t = \{r_1, r_2, \dots r_k\}$	(3.5)
$B_t = [b_t, b_{t-1}, \dots b_{t-j}]$	(3.6)

where e_i is the i-th event made up of a vector of inputs h with m dimensions, w_i is the i-th sliding window containing a series of j events, v_i is the price volatility difference between the i-th and (i - 1)th price, c_i is the i-th sliding window containing a series of j-1 price volatility differences, b_t is the rule base of the system at time t, and B is a set of rule bases at time t with length equal to window_size j.

Event Detection: Event detection refers to the process of detecting which events can be used to trigger episode encoding and recall [10]. The process of event detection for GEMM-FNN, D_e, is defined as:

$$D_e(w_t) = \begin{cases} 1, \ if\ T(w_t) > \theta_{tr} \\ 0\ , else\ otherwise \end{cases} \quad (3.7)$$

$$T(x) = (\max(x) - \min(x))/\min(x) \quad (3.8)$$

where D_e takes in an episode m_i and determines whether to trigger the episodic memory mechanism (1) or not (0), w_i is the i-th sliding window containing a series of j events , T(x) computes the maximum percentage difference between the highest and lowest values in a vector x, and θ_{tr} is volatility threshold parameter that is used to detect transient events.

Episode Encoding: Episode Encoding E_e is defined as the process of storing a set of events as an episode [10] in a long term memory structure L_m. We denote episodic memory m_i mathematically as a window of distinct events (e), that is triggered by a cue (c) as follows:

$$\bar{e}_{i,j} = \text{mean}(w_i) \quad (3.9)$$

$$\bar{b}_{i,j} = C(B) = \{r_1, r_2, \dots r_3\} \quad (3.10)$$

$$d_i = \{t, T(w_t), \bar{e}_{i,j}, \bar{b}_{i,j}\} \quad (3.11)$$

$$E_e(c_i, d_i) = \begin{cases} \text{mi} = \{ci : d\} \\ L_m = L_m + m_i, \ if\ len(I_e(c_i)) > 0 \\ Else, nothing\ otherwise \end{cases} \quad (3.12)$$

where c_i is a cue that contains a vector of price volatility differences $\{v_i, v_{i-1}, \dots v_{i-j-1}\}$, t is the time index of the current input tuple $\bar{e}_{i,j}$ is the average of j input events of the i-th episode, C(B) is a rule consolidation process that creates a set of unique rules aggregated from the i-th window rulebase B_i, $\bar{b}_{i,j}$ is a set of unique rules aggregated from the i-th window rulebase B_i, and m_i is the i-th episodic memory made up of cue c_i and d_i. The set of details d_i of the i-th memory m, consisting of a set of a cue, volatility, average input, and window_rulebase. $E_e(m_i)$ is a function that creates a memory from c_i and d_i and appends to the long-term memory store. Hence, the short-term memory stores a series of events made up of windows of past inputs, rulebases and volatilities. A transient event triggers episode encoding if no similar episode exists in the long term memory.

Episode recognition: Episode recognition refers to the identification of a stored episode in the episodic memory in response to a partial event sequence [10]. Suppose an event is detected, it should then be checked if there are any episodes with similar events associated with it. We may define episode recognition I_e as: given a new cue c_t at time t and a long-term memory store L_m containing a set of episodes, there exists a set of episodes with cues that are similar if any of them fall within a similarity threshold θ_s as defined below:

$$I_e(c_t) = \{m_1, \dots, m_{t-1}\} \subseteq L_m \text{ , where } S(c_t, c_{t-1}) < \theta_s \quad (3.13)$$

$$S(c_t, c_{t-1}) = ||c_t - c_{t-1}||2 = \sqrt{\sum(c_{t,i} - c_{t,j})^2} \quad (3.14)$$

$S(c_t, c_{t-1})$ computes the similarity of two episodes m_t and m_{t-1} using Euclidean distance between their cues c_t and c_{t-1}. If there exists a subset of stored episodes that have cues similar to the current event, then episode with the most similar cue is recalled. Otherwise, this event is encoded as a novel episode to be stored in the memory. The pseudocode to aggregate all rules within a window B containing a set of j rule bases for rule consolidation C(B) is presented in Algorithm 1.

Algortihm 1: Pseudocode to aggregate all rules within a window B containing a set of j rule bases for the function rule consolidation C(B).

```
rule_consolidation(B):
        all_rules = list()
        unique_rules  = list()
for rulebase in B:
                for rule in rulebase:
                        all_rules.append(rule)
        for rule in A:
                rule.relevance = mean(rule.weight)
                rule.rmse = mean(rule.rmse)
                new_rule = {rule.antecedent, rule.consequent,
rule.relevance, rule.rmse}
        unique_rules.append(new_rule)
return unique_rules
```

Episode recall: Episode recall comprises of the replaying of an episode when an external cue is presented [10]. We denote episode recall R_e as: given a set of similar episodes from RC_e, retrieve the most similar episode m_s and create new rules based on their relevance as follows.

$R_e(A, \text{max\_a}, \text{max\_c}) = \{r_1^k, r_2^k, \dots r_3^k\}$, where $r_i^k . U_r > \theta_r$

where r_i^k is a set of modified rules belonging to the closest episode , A is a set of similar recalled memories $I_e(ct) = \{m_1, \dots, m_{t-1}\}$. Hence, episode recall returns a unique set of relevant rules. The pseudocode that aggregates all rules within a window B containing a set of j rulebases for rule retrieval $R_e(A, \text{max\_a}, \text{max\_c})$ is presented in Algorithm 2.

Algorithm 2: Pseudocode to aggregate all rules within a window B containing a set of j rule bases for rule retrieval $R_e(A, \text{max\_a}, \text{max\_c})$.

```
R(A, max_a, max_c):
closest_episode = Min(A.diff)
rules = list()
for rule in closest_episode:
        w = rule.relevance
        if w > θr:
                new_rule.a = w . rule.a + (1- w) . max_a
                new_rule.c = w . rule.c + (1- w) . max_c
                new_rule.relevance = w
                new_rule.rmse = rule.rmse
                rules.append(new_rule)
return rules
```

Types of episodes in the financial domain: The significance of a financial event $C_{s,i}$ is indicated by its volatility, i.e. the fluctuation of price in the market. Based on the value of volatility V_m, we define three types of episodes, namely regular (m_R), déjà vu (m_D), and PTSD (m_P) as follows:

$$m = m_P \text{ , if } V_m \leq -25\%$$
$$m = m_D \text{ , if } -25\% < V_m < -15\% \text{ or } 15\% < V_m < 25\%$$
$$m = m_R \text{ , else otherwise} \quad (3.15)$$

PTSD episodes occur during the most negative of events, followed by Deja Vu and regular episodes.

Summary of episodic mechanisms: To summarize, the sensory memory actively takes in input targets and features, or the details of each event. We create a sliding window of fixed length w, each containing 1) A rule base containing all active rules 2) Input data for both the target and features. Next, for each window, we compute the overall volatility, volatility window, average inputs and relevance of each rule using the mean of their weights. These are stored in the short-

term/working memory. Using event detection, if the overall volatility crosses a certain threshold, the episodic mechanism is activated. Event encoding then converts an event into an episode. The long-term memory store is checked for similar episodes using episode recognition. If no previously similar episodes are found, the episode is cached in the long term memory store alongside its relevant rules. If a similar episode is found, the stored similar episode is retrieved, its rules are modified by computing a weighted average with the current max_set antecedent/consequent, and the modified rules are injected into the working memory of the system.

We note the five hyper-parameters of our episodic memory mechanism for financial data. They are the window size (W), transient volatility threshold (θ_v), volatility similarity threshold (θ_s), rule relevance threshold (θ_r), and error allowance threshold (θ_e). Further experiments have been conducted in [11] to tweak and understand their effect on the mechanism. The subsequent benchmarks and analysis use the optimized hyper-parameters.

III. Benchmark Experiments and Analysis

To evaluate the performance of the proposed GEMM-eMFIS (FRI/E), a series of benchmarks are performed, followed by investigation of other potential applications. Three types of benchmark experiments were performed:

A. Exchange traded funds (ETF) indices - DJIA and S&P 500
B. Individual stocks (e.g. Ford, Apple)
C. Rainfall Runoff [12]

The performance measures used in the experiments are root mean-square error (RMSE) and Pearson's product-moment correlation coefficient. The average number of rules created by each model during the prediction process is also included as an interpretability measure. Further, to evaluate the efficiency of the episodic memory cache, the number of transient events, episodes cached, successful recalls and percentage of relevant episodes are investigated. Finally, an execution time used for training and testing are also included as a measure of speed.

A. Using Global Macro Events To Predict ETF Indices

This experiment seeks to validate several key hypotheses that demonstrate the plausibility of predicting the impact of financial crisis' on ETFs through the implementation of an episodic memory mechanism on a Neuro-Fuzzy System. Throughout, the analysis will show that not only is the mechanism able to capture global financial macro events, it also provides better performance in accuracy and interpretability, that traders may use for profit.

Datasets: Two datasets, SNP500 and DJIA, were chosen to give a balanced representation of price movements of ETF. Extracted from Bloomberg Terminal [13], US Bureau of Economic Analysis [11] and US National Bureau of Economic Research [11], each dataset contain 54-years of daily prices of the respective stocks/indices, from December 1964 to February 2019. Rows of non-trading days were removed to prevent duplicate prices and to simulate a real trading scenario. The dataset consists of six features for presented in TABLE I. The features were selected to provide a good mix of technical indicators and macro-economic events.

TABLE I. Features for Macroeconomic Events Dataset

#	Feature	Description
1	Volume	A decimal number representing the total daily trading volume of the security.
2	Moving Average	A decimal number that computes a simple moving average of the securities' price for the past 15 days.
3	7 Days Ago Prices	A decimal number representing the price of the security 7 days ago from today.
4	21 Days Ago Prices	A decimal number representing the price of the security 21 days ago from today.
5	21 Days Ago Prices	A decimal number representing the price of the security 21 days ago from today.
6	Event GDP Affect	A negative percentage value between 0 to -100, that represents the decline in GDP of the US economy during a recession (peak-to-trough).

TABLE II. Experimental Results for the SNP500 Index Dataset

Architecture	RMSE	Correlation	Average # Rules	Run Time (s)
GEMM-eMFIS(FRIE)	25.7548	0.99932	19.01	3522.80
eMFIS(FRIE)[11]	26.5938	0.99928	19.35	1411
eMFIS[6]	27.0472	0.99926	22.9	1076
EFUNN[21]	26.4	-	3189	291.8
SAFIN[18]	98.7	-	1	312.32
ANFIS[20]	25.72	0.99500	3	44.32

TABLE III. Experimental Results for the DJIA Index Dataset

Architecture	RMSE	Correlation	Average # Rules	Run Time(s)
GEMM-eMFIS(FRIE)	190.02	0.99953	20.80	2965
eMFIS(FRIE)[11]	191.05	0.99954	21.54	1326
eMFIS[6]	196.56	0.99950	21.56	1407
EFUNN[21]	358.02	-	3028	121
SAFIN[18]	877.87	-	1	23.7
ANFIS[20]	192.55	0.99950	2	21.16

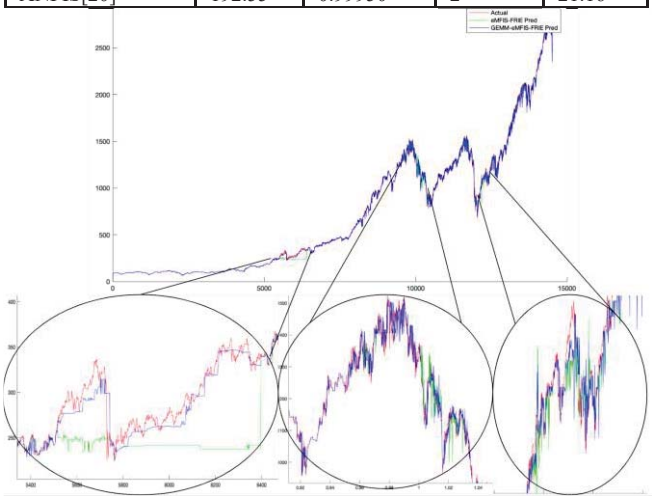


Fig. 3. Comparison of the real data (red) of SNP500 dataset and predictions by GEMM-eMFIS(FRIE) in blue and eMFIS(FRIE) in green.

Performance evaluation: The experimental results and comparison with several state-of-the-art methods for SNP 500 and DJIA are presented in TABLE II. and TABLE III. , respectively. We see that GEMM-eMFIS(FRIE) outperforms all other Mamdani systems as it is both more accurate and requires fewer rules. GEMM-eMFIS(FRIE) performs better overall compared to other Mamdani models and is even on par with TSK models, having not just better accuracy but also better interpretability as it has fewer rules. Fig. 3 presents the comparison of real data and trends predicted by GEMM-

eMFIS (FRI/E) and eMFIS (FRI/E). We see that eMFIS (FRI/E) struggles during periods of transient behavior as it is simply unable to derive accurate rules based on the existing rules. In contrast, we see the true power of the episodic memory mechanism coming into play here as GEMM-eMFIS (FRIE) (blue) is able to quickly adapt to the sudden drop in prices to achieve better accuracy. Using GEMM-eMFIS (FRIE) will allow traders to not only use macroeconomic features to predict sudden changes in price, but will also provide them with an episode base that they can reference in the future to understand the reasons for the swings in the market. As the model learns, its episode base grows and becomes more resilient to financial crises. Finally, traders can go one step further to employ GEMM-eMFIS(FRIE) for anticipatory trading by increasing the window size such that the model can learn the behavior of fluctuation of prices several time steps ahead of the sudden drop or spike, enabling them to perform long and short term planning on the market.

Analysis of episode recall: The main hypothesis of the episodic mechanism is that it learns, stores and recalls relevant episodes. We use the list of global financial crises since 1965 [14] as a reference on our timeline to identify the events being cached and recalled on the SNP500 dataset. Out of 300+ episodes cached, TABLE IV. shows some top ones. We see that episodes from previous events are indeed recalled, and that the model is able to correctly recognize similar events. For example, episode id 347 was detected and stored in the episodic memory cache in September 1990 close to the 1990s recession (A). It was retrieved during several later US financial downturns (B-F). This provides a new dimension of interpreting the model to stock market traders. By knowing which episodes trigger certain movements in the prices, they essentially now have 'inside information' as to how the market will likely react when a certain financial crisis is impending, potentially giving them competitive advantage over other models that do not have such episodic memory.

TABLE IV. LIST OF EPISODES RECALLED AT THEIR RESPECTIVE EVENTS AND DATES FOR SNP500 DATASET. THE EVENTS ARE REPRESENTED AS (INPUT TUPLE #, DATE, NAME)

Episode 53	**Cached**: 1933, 1973-05-17 Britain stock market crash
Recalled:	2174, 1974-05-01 OPEC Oil Crisis 3754, 1980-07-15 1980s recession 5587, 1987-04-20 Black Monday 10250, 2002-01-02 Recession by September 11 13737, 2015-11-06 Chinese stock market crash
Episode 104	**Cached**: 2182 1974-05-13 Opec oil crisis
Recalled:	3775, 1980-08-13 1980s recession 4043, 1981-08-11 Iranian Revolution 8752, 1997-11-07 Asian Financial Crisis 12408, 2010-07-29 Flash Crash
Episode 265	**Cached**: 4198, 1982-03-05 Iranian Revolution
Recalled:	6108, 1989-03-28 Friday the 13th Flash Crash 6554, 1990-09-28 1990s recession 6805, 1991-08-06 Japanese Asset Bubble 7134, 1992-10-01 Black Wednesday 11691, 2007-09-24 SSE Index Crash
Episode 347	**Cached**: 6549, 1990-09-28 Early 1990s Recession (A)
Recalled:	7136, 1992-10-05 Black Wednesday (B) 9127, 1998-07-21 Russian Financial Crisis (C) 10023, 2001-02-02 US Tech stock bubble crash (D) 11633, 2007-07-02 SSE Index Crash (E) 12607, 2011-05-12 US Bear Market (F)
Episode 369	**Cached**: 8643 1997-08-20 Asian Financial Crisis
Recalled:	10240, 2001-12-17 September 11 Attacks 12227, 2009-11-06 08-09 Financial Crisis' 13699, 2015-09-15 Global Stock Market Sell Off

TABLE V. TYPES OF EPISODIC MEMORIES BEING CACHED.

Episode ID	Date Cached	Volatility	Type
531 (A)	1987-10-26	-28.87%	PTSD
802 (B)	2008-10-10	-25.47%	PTSD
878 (D)	12334 2011-11-08	10.31%	Deja Vu
756 (B)	9992 2002-10-17	11.48%	Deja Vu
825 (C)	11653 2009-02-27	-11.09%	Deja Vu
699 (A)	9578 2001-09-19	-14.52%	Deja Vu
765	10159	5.05%	Regular
775	11304	5.55%	Regular
867	12285	-6.20%	Regular
917	13922	-7.19%	Regular

Types of episodes being recalled: Earlier we mentioned that there 3 types of memories (regular, DeJaVu and PTSD). Let us now investigate whether we are indeed able to classify memories into 3 distinct types based on the volatility of the price window. We explore the episodes created from running GEMM-eMFIS(FRIE) on the DJIA dataset and observe the types of episodes cached. The list of episodes and their characteristics were saved into a flat-file and an analysis was performed using pandas data frames to classify the various episode types based on their window volatility, V_m.

TABLE V. shows the caching of 2 PTSD type episodes during catastrophic events like the 2008 global financial crisis where there were sudden drop in prices greater than 25%. These represent traumatic incidents in the dataset and for the model, which occur rarely through the lifetime of the model. Déjà vus are events that seems familiar and with vivid photographic detail. This contrast with normal remembrances where certain sensory details have eroded over a long time. The memory can be both a positive or negative. TABLE V. shows the caching of 4 DeJa Vu type episodes during periods where there were either sudden rises or fall in prices between 15-25%. These events are not as severe as PTSD, but occur more occasionally and are still significant enough for the model to cache them. TABLE V. also shows the caching of 4 regular type episodes during periods where there were either sudden rises or fall in prices between 15 to -15%. These events are relatively more common than PTSD and DeJa Vu, and represent quarterly corporate sentiments in the dataset but are less significant in their price volatility.

B. Using Corporate Events To Predict Stock Prices

This experiment seeks to validate several key hypotheses that demonstrate the plausibility of predicting the impact of corporate events on stock prices using GEMM-eMFIS (FRI/E). The analysis will show that not only the mechanism is able to capture internal corporate events, but it also provides better performance in accuracy and interpretability that traders may potentially use to help improve profits.

Dataset: The individual stock prices of Ford and Microsoft were chosen because these companies have existed for at least 30 years and extensive data has been collected about them. Extracted from the NTU Bloomberg Terminal, each dataset contain at least 30-years of daily prices of the respective stocks until February 2019. However, Microsoft begins from March 1986 as it was incorporated only then while Ford begins from December 1964, as the oldest date available. Rows of non-trading days were removed to prevent duplicate prices and to simulate a real trading scenario. Each dataset consists of seven features listed in TABLE VI. The features were selected to provide a good mix of technical indicators and corporate actions, and have been tested to be leading indicators. While technical indicators are relatively straight forward, it is worth

it to mention that earnings surprise [11] is a powerful feature as future revenues are what analysts mainly use to decide on the valuation of share prices. Other sources of data they take into consideration include quarterly and annual reports, current economic conditions and the company's own revenue guidance [15]. Should the company experience a negatively unexpected earnings surprise due to poor performance that quarter, analysts will likely become bearish on the stock and reduce their future forecasts, lowering the company's valuation and hence stock price.

Performance evaluation: The experimental results and comparison with several state-of-the-art approaches for the two companies are given in TABLE VII. and TABLE VIII. We see that GEMM-eMFIS(FRIE) performs better compared to other Mamdani models and at par with TSK models, having not just better accuracy but also better interpretability as it has fewer rules. We also see that it makes better predictions specifically during transient events at t=3900 and t=7600. Fig. 4 presents the comparison of real data and trends predicted by GEMM-eMFIS (FRI/E) and eMFIS (FRI/E). Similar to the observation in Fig. 3, we note here also that GEMM-eMFIS(FRIE) (blue) is able to quickly adapt to the sudden drop in prices to achieve better accuracy than eMFIS (FRI/E).

Analysis of episode recall: The timeline events of Microsoft listed in [16] is used as a reference on our timeline to assess the recall of episodes. Out of 1000+ episodes cached, the episodes that have been recalled several times are presented in TABLE IX. It is interesting to note that episode id 468 was detected and stored in the episodic memory in December 1990, when MS Office was released (A). It was then retrieved later during several other product releases (B-G) such as Internet Gaming Zone, Windows XP and Microsoft Surface.

TABLE VI. LIST OF FEATURES FOR FORD AND MICROSOFT CORPORATE EVENT DATASETS

#	Feature	Description
1	Volume	Total daily trading volume of the security.
2	Moving Average	Simple moving average of the securities' price for the past 15 days.
3	Prices 7 Days Ago	The price of the security 7 days ago from today.
4	Prices 14 Days Ago	The price of the security 14 days ago from today.
5	Prices 21 Days Ago	The price of the security 21 days ago from today.
6	Earnings Surprise	A percentage value that signifies how well a company's earnings beat bloomberg analyst expectations for that quarter. A negative value means the company performed poorer than expected.
7	Description	An integer value of 1-3, categorizing the type of corporate event on a particular date, which could be an earnings release (1), a public conference/ announcement (2) or shareholder/ partnership meeting (3).

TABLE VII. EXPERIMENTAL RESULTS FOR FORD DATASET

Architecture	RMSE	Corre-lation	Average # Rules	Run Time (s)
GEMM-eMFIS(FRIE)	0.4138	0.9960	17.55	1475.5
eMFIS(FRIE)[11]	0.4210	0.9958	16.40	750.7
eMFIS[6]	0.4261	0.9957	16.45	749.0
EFuNN[21]	0.334	0.9996	4401	363.8
SAFIN[18]	0.6699	0.9928	1	291.96
ANFIS[20]	0.464	0.9973	7	36.4
DENFIS[19]	0.475	-	4	7.9

TABLE VIII. EXPERIMENTAL RESULTS FOR MICROSOFT DATASET

Architecture	RMSE	Corre-lation	Average # Rules	RunTime
GEMM-eMFIS(FRIE)	1.1381	0.9989	21.51	1685.8
eMFIS(FRIE)[11]	1.2809	0.9986	21.86	936.9
eMFIS[6]	1.2952	0.9985	21.86	706.0
EFuNN[21]	1.346	-	2416	263.8
SAFIN[18]	4.606	0.9622	1	167.91
ANFIS[20]	1.73	0.9947	3	12.49
DENFIS[19]	1.258	-	9	7.4

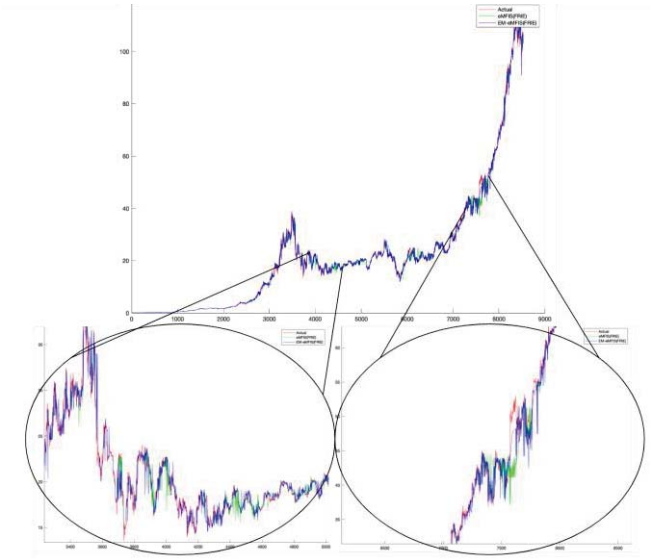


Fig. 4. Comparison of real data (in red) of Microsoft with the predictions of GEMM-eMFIS(FRI/E) in blue color and eMFIS *FRI/E) in green color.

TABLE IX. LIST OF EPISODES RECALLED AT THEIR RESPECTIVE EVENTS AND DATES FOR THE MICROSOFT DATASET. THE EVENTS ARE REPRESENTED AS (INPUT TUPLE #, DATE, NAME)

Episode 54	**Cached**: 117, 1986-08-28, Microsoft goes public (IPO)
Recalled:	670, 1988-11-03 Apple sues Microsoft 1044, 1990-04-30 Microsoft launches Windows 3.0 4043, 2002-03-20 .NET initiative was launched 6528, 2011-10-20 Skype Acquisition
Episode 103	**Cached**: 204, 1987-01-02, Microsoft announces OS/2
Recalled:	1010, 1990-03-12 Microsoft launches Windows 3.0 1440, 1991-11-20 Microsoft Research launches 4300, 2003-03-27 Windows Mobile Launches 7760, 2016-07-14 Microsoft acquires LinkedIn
Episode 206	**Cached**: 510, 1988-03-18, Apple sues Microsoft
Recalled:	2074, 1994-05-25 Windows NT 3.5 launches 2855, 1997-06-26 Apple Inc partners with Microsoft 5913, 2009-06-15 Microsoft Bing launches
Episode 468	**Cached**: 1209 1990-12-21, MS Office Launched (A)
Recalled:	1648, 1992-09-17 Microsoft releases Windows 3.1.(B) 2557, 1996-04-23 Internet Gaming Zone launches (C) 3864, 2001-06-27 Microsoft releases Windows XP (D) 4414, 2003-09-09 Windows Mobile launches (E) 5708, 2008-08-25 Bill Gates retires (F) 6618, 2012-02-22 Microsoft Surface,launches (G)
Episode 606	**Cached**: 1913, 1993-10-05, Windows NT 3.1 released
Recalled:	2145, Microsoft releases Windows NT 3.5. 2709, Microsoft launches Expedia 4909, Microsoft launches the Xbox 360. 6272, Microsoft announces Windows Phone

C. Experiment: Rain Runoff

This experiment demonstrates the feasibility of predicting rainfall runoff through the implementation of an episodic memory mechanism on a Neuro-Fuzzy System, as an example of non-financial application.

TABLE X. SUMMARY OF RECORDED RAINFALL EVENTS

		Rainfall		Bay 1		Bay 2	
Event	Duration	Peak Intensity	Average Intensity	Peak Flow	Average Flow	Peak Flow	Average Flow
	(min)	(mm/h)	(mm/h)	(L/s)	(L/s)	(L/s)	(L/s)
1	40	144.00	33.4	0.641	0.194	0.713	0.194
2	80	144.00	19.9	0.758	0.118	0.858	0.122
3	40	216.00	43.2	0.912	0.241	1.004	0.256
4	120	144.00	17.5	0.745	0.105	0.868	0.115
5	30	144.00	32.6	0.601	0.182	0.644	0.190
6	120	144.00	25.7	0.730	0.158	0.782	0.168
7	120	96.00	8.4	0.495	0.054	0.548	0.057
8	90	120.00	19.0	0.601	0.121	0.647	0.124
9	100	144.00	16.6	0.832	0.111	0.927	0.110
10	100	72.00	11.9	0.352	0.075	0.395	0.078

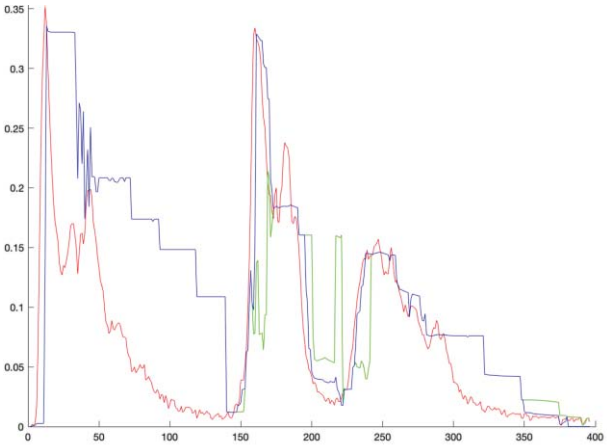


Fig. 5. GEMM-eMFIS(FRIE) rainfall runoff prediction

TABLE XI. EXPERIMENTAL BENCHMARK FOR BAY 1 RAINFALL DATASET

Event	Architecture	RMSE	Correlation	Average # Rules	Run Time
4	GEMM-eMFIS (FRIE)	0.0605	0.92971	3.6155	15.67
4	eMFIS(FRIE)	0.0653	0.91729	2.0273	8.16
6	GEMM-eMFIS (FRIE)	0.0839	0.84895	3.2222	17.4443
6	eMFIS(FRIE)	0.0851	0.84321	1.9419	8.7449
7	GEMM-eMFIS (FRIE)	0.0209	0.9721	1.4076	16.2434
7	eMFIS(FRIE)	0.0209	0.9721	1.3172	7.4354
9	GEMM-eMFIS (FRIE)	0.1109	0.83867	1.9495	15.2015
9	eMFIS(FRIE)	0.1110	0.83876	1.7323	6.4353
10	GEMM-eMFIS (FRIE)	0.0804	0.66813	2.7702	19.326
10	eMFIS(FRIE)	0.0886	0.50866	1.5328	6.6309

TABLE XII. CROSS VALIDATION RESULTS FOR RAINFALL DATASET

Train	Pred	RMSE	Correlation	Avg # Rules
4	6	0.0885	0.832	3.9
7	6	0.0868	0.835	3.56
9	6	0.0896	0.851	3.6641
10	6	0.087	0.83	2.9
4	7	0.0203	0.971	1.37
6	7	0.0205	0.972	1.3
9	7	0.0203	0.970	1.23
10	7	0.0202	0.971	1.5
4	9	0.0796	0.919	2.1
6	9	0.0806	0.916	2.02
7	9	0.1048	0.857	2.22
10	9	0.063	0.949	1.87

Dataset collection: The dataset consists of several rainfall events, each consisting of unit rainfall collected (1) using ground level sensors and (2) in a water catchment bay after run-off over different periods of time. The idea is to be able to forecast the amount of rainwater collected in the catchment bay during a rain event. The data have been obtained from an outdoor experimental plot set up at the Nanyang Technological University, Singapore. The experimental plot comprises four 25m long by 1m wide test sections. Further details of the experimental plot can be found in [17]. A summary of the recorded rainfall data is shown in TABLE X. Note that the discharge at the beginning of the rain for bay 1 is higher than that in bay 2, however, the discharge rate is much higher for bay 2 when the rain intensity increases as well. There is also some noise in the data as rainfall and environmental conditions are erratic. Out of all events, we use events 4, 6, 7, 9 and 10 as they last at least 100 mins and represent a good distribution of peak discharge and intensity:

- Event 4: High peak discharge and rain intensity.
- Event 6: High peak discharge with multiple peak events.
- Event 7: Low peak discharge.
- Event 9: High peak discharge, single peak event.
- Event 10: Low peak discharge.

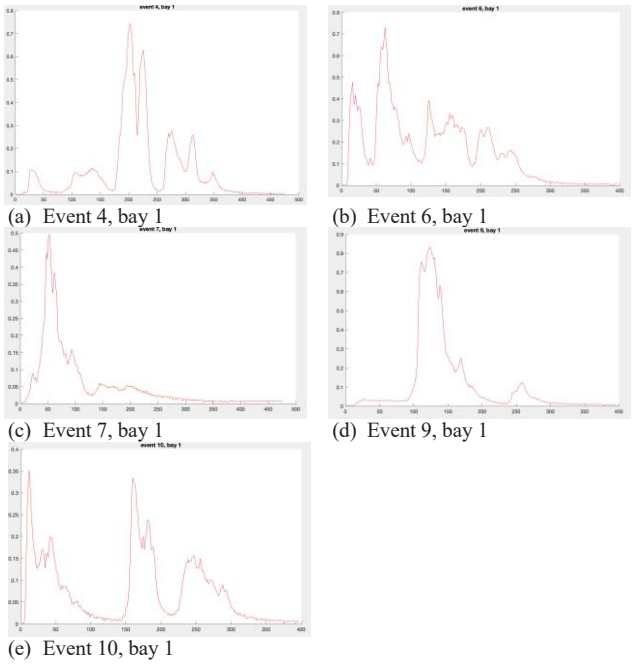
Intuitively, discharge rate depends on the amount of rainwater is in the collection bays and in turn the amount of rainwater depends on the intensity of the rain. Hence, we expect that the models with rainfall as input should give a fair accuracy of discharge. The features used for each event are the rainfall amount at time t-5 t-4, t-3, t-2, t-1 to predict the discharge at time t in Bay 1.

Results: Fig. 5 shows that the model is able to represent the rainfall data relatively well and is able to handle sharp transient spikes and drops in the dynamic data at t=120 and t=220, using its episodic memory cache. From TABLE XI. , we see that GEMM-eMFIS(FRI/E) outperforms EMFIS (FRI/E). Fig. 5 shows that interpolation/extrapolation methods of eMFIS(FRIE) (green) are unable to react quickly to transient changes, whereas GEMM-eMFIS(FRIE) is able to learn from the first spike episode and use it to react much more quickly and accurately during the next transient event. This is very important for governments monitoring rainfall data because the episode base built up can be potentially to predict weather crises' such as floods or hurricanes, that could save lives.

Results – cross validation: Next, we took our experiment further to investigate the extent to which the episodes from one rainfall event could help us predict another. This was done by training GEMM-eMFIS(FRIE) on a single dataset and then testing on another dataset, while still retaining the rules that the model learned during training. Ideally, we should expect better test performance on events where the data distribution of the training and testing are more similar. For this experiment, we shall only explore prediction of datasets 6, 7 and 9 in order to reduce complexity.

TABLE XII. shows that the model achieve relatively lower RMSE and higher correlation when training on datasets 4 and 10 before predicting on dataset 9. This makes sense as the three of them have large spikes around the 150-200 timestamp as seen in Fig. 6. It is suspected that event 7 performed much worse because the spike was much sharper and brief than in event 9. Next it also shows the best performance when training on dataset 10, before predicting on 7, because they both

experience similar sharp spikes at the beginning of the dataset. Finally, we see that relatively better performance is achieved when training on datasets 7 and 10 before predicting on 6. This is likely because they are not only both experience similar sharp spikes at the beginning of the dataset, but event 10 also has a relatively similar distribution compared to 6. This we can conclude to a large extent, that the cached episodic memories are able to improve accuracy when predicting similar events.



(a) Event 4, bay 1 (b) Event 6, bay 1

(c) Event 7, bay 1 (d) Event 9, bay 1

(e) Event 10, bay 1

Fig. 6. Rainfall Runoff Target Volumes over time for each event

IV. Conculsions and Remarks

This paper has proposed a novel and generic episodic memory mechanism that integrates with neuro-fuzzy system with the following features. First, an episodic mechanism based on human psychology, that can encode, store, recognize and recall similar transient events in online real-time data, has been devised. Second, the episodic mechanism has been integrated into a neuro-fuzzy model and is able to provide it superior accuracy in solving complex real-world problems, mainly in the domain of finance. Lastly, the model provides human-readable decision-making logic, which can be analyzed and understood by the decision makers. Uses the processes of Event Detection, Episode Encoding, Episode Recognition and Episode Recall to store and retrieve transient events. Relevant rules from most similarly recalled episode are injected into the system to deal with rapid changes in the target. This paper also provides benchmarking of the proposed GEMM-eMFIS(FRIE) against similar neuro-fuzzy models. It is demonstrated to provide both superior accuracy and interpretability when applied to financial and rainfall datasets. Results also show that the mechanism is able to correctly store and recall relevant events in the financial domain. Overall, this points towards the generalizability of the episodic mechanism on both model type and domain. Some limitations remain in the episodic mechanism and it can be further improved in several aspects. The first is that the episodic memory cache has only been demonstrated to work effectively on online fuzzy-neuro systems. In theory, it should be feasible to apply it to offline scenarios and other types of computationally intelligent models. Second, depending on the dataset and model, an external tuning algorithm may be required to optimize its hyper-parameters. These aspects will be explored in the future.

References

[1] G. Bin, H. S.-L. James, “Computing Transient Response of Dynamic Systems in the Frequency Domain”, Journal of Engineering Mechanics doi: 10.1061/(ASCE)EM.1943-7889.0001398, 2018.

[2] M. Ashrafi, “Online Neuro-Fuzzy Models For Real Time Flow Forecasting”, Nanyang Technological University, Singapore, 2017.

[3] Susanti, "The Evolving Mamdani Fuzzy Inference System with Fuzzy Rule Interpolation/ Extrapolation" Nanyang Technological University, School of Computing, Singapore, 2014.

[4] F. Li, Y. Li, C. Shang, Q. Shen, “Improving fuzzy rule interpolation performance with information gain-guided antecedent weighting.”, Soft Computing, 2017.

[5] J. Tan and C. Quek, "A BCM-theory of meta-plasticity for online self-reorganizing fuzzy-associative learning," IEEE Trans. Neural Netw., vol. 21, no. 6, pp. 985-1003, 2010.

[6] M. Hartanto, "The Evolving Mamdani Fuzzy Inference System (eMFIS)," Nanyang Technological University, School of Computer Engineering, Singapore, 2014.

[7] J. M. Mandler, P. Bauer and L. McDonough, "Separating the sheep from the goats: Di," Cognitive Psychology, vol. 23, pp. 263-298, 1991.

[8] T. J. Palmeri and M. A. Flanery, "Prototype abstraction in category learning?," in Proceedings of the 23rd Annual Meeting of the Cognitive Science Society, Edinburgh, 2001.

[9] Z. H. Huang and Q. Shen, “Fuzzy interpolation and extrapolation: a practical approach,” IEEE Trans. Fuzzy Syst., vol. 16, no. 1, pp. 13–28, Feb. 2008.

[10] W. Wang, “Neural Modeling of Multiple Memory Systems and Learning”, Nanyang Technological University, Singapore, 2015.

[11] S.W. Pang, “GEMM-eMFIS(FRIE): A General Episodic Memory Mechanism For Fuzzy Neural Networks”, FYP Report, Nanyang Technological University, Singapore, 2019.

[12] Y. L.-T. Larry, “Experimental Investigation into Benchmarking of Neural Networks in Rainfall-Runoff Prediction”, Nanyang Technological University, Singapore, 2008.

[13] Bloomberg L.P. Stock Price and News for SNP500 and DJIA indices Bloomberg terminal, 29-Jan-2019. .

[14] Wikipedia contributors, "List of stock market crashes.", Wikipedia, The Free Encyclopedia, date of access. 23 Jan. 2020, 2020.

[15] J.E. Pinto, E. Henry; T. R. Robinson; J. D. Stowe “Equity Asset Valuation (2nd ed.)”, John Wiley & Sons, 2010.

[16] Wikipedia contributors. "Timeline of Microsoft." Wikipedia, The Free Encyclopedia. Wikipedia, The Free Encyclopedia, 9 Feb. 2020. Web. 23 Mar. 2020.

[17] T. Kohonen, “Analysis of a Simple Self-Organizing Process.”, Biological Cybernetics. 44. 135-140, 1982.

[18] S. W. Tung, C. Quek, and C. Guan, “SaFIN: A Self-Adaptive Fuzzy Inference Network”, IEEE Transactions on Neural Networks, 2011.

[19] N. K. Kasabov and Q. Song, "DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction," IEEE Trans. Systs., Man, Cybern. B, vol. 10, 2002.

[20] J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," IEEE Trans. Systs, Man & Cyberns., vol. 23, no. 3, pp. 665-685, 1993.

[21] N. Kasabov, "Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning." IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 31, no. 6, 902-918, 2001.