# A Temporal Continual Learning Framework for Investment Decisions

City, University of London


Daniel G. Philps, CFA, B.Sc (Hons.)

January, 2020

**Abstract**

*Temporal continual learning* (TCL) is introduced in this thesis as an extension of *continual learning* (CL). While traditional CL has been applied to sequential tasks, extending CL to TCL aims to allow machines to accumulate specific knowledge of temporal states, to address *concept drift* (CD) problems. This approach is shown to hold considerable benefits in domains where non-stationary time-series are used for decision-making, particularly in finance.

A TCL framework called *continual learning augmentation* (CLA) is introduced, to drive long-term decision making in complex, multivariate, temporal problems. Moreover, CLA uses an external memory structure to store learner parameters from particular past temporal states for recall in the future. The contributions of this work are fourfold: First, a temporal, state-based, external memory structure is developed. Second, this is used to memory augment well-understood base-learners, such as LSTM, feed-forward neural networks (FFNN) and linear regression. Third, a remember-gate, based on residual-change, learns in an open-world fashion to define different states for which learner-parameters are stored along with a contextual reference of the state. Fourthly, a memory recall-gate is developed, based on various time-series similarity approaches, which can compare the current input space with the contextual references stored in memory, recalling the most appropriate learner parameters for use in the current period.

In testing, CLA is found to improve the performance of LSTM, FFNN, and linear regression learners applied to a complex, real-world finance task: stock selection in international and emerging equities investing. Several different similarity approaches are tested in CLA's remember-gate, with dynamic time warping (DTW) outperforming simple Euclidean distance (ED), while auto-encoder (AE) distance is found to both mitigate the resource overheads of DTW and provide better performance. A hybrid approach is also introduced, *warp-AE*, which performs well. In addition, a visualisation is introduced to allow CLA to be interpreted by domain experts in terms of which memory did what and when. A complex application is used to test TCL and a five-point statistical testing framework is introduced. This thesis elucidates the research of the last five years regarding TCL.

**Keywords**: Continual learning, time-series, memory, neural network.

# Dedication

Without the support of my wife, Ruth, this work would have been impossible, without my parents' unwavering patience improbable, and without the counsel of my dear friend Dr Bilgin Soylu impracticable. This work is dedicated to you.

# Declaration

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where stated otherwise by reference or acknowledgment, the work presented is entirely my own.

# Acknowledgements

My sincerest thanks to Prof Artur d'Avila Garcez and to Dr Tillman Weyde for all of their many contributions to this work and for their incredible multi-year mentorship of my journey, deep into the brave new world of artificial intelligence. Thanks to Prof Roy Batchelor for his contributions, support and advice.

Thanks also to Mr David Tilles for his belief in me and for his support of this research, without which it would not have been possible.

# List of Symbols

$a(h(\dots))$ Autoencoder functions; encoder $a(\dots)$ and decoder $h(\dots)$

$\hat{D_{ED}}$ Estimated Euclidean distance

$\hat{D_{DTW}}$ Estimated dynamic time warping distance

$\epsilon_t$ Residual value at time, $t$

$\boldsymbol{\epsilon_B}$ Vector containing the absolute error series of a learner, $B$

$\epsilon_{B,t}$ Absolute error value of a learner, $B$, observed at time, $t$

$g$ Recall-gate function

$j$ Remember-gate function

$k$ A distinct variable, generally a time series in a multivariate time series

$K$ Total number of time series in a multivariate time series

$m$ Discrete memory

$M$ Total number of discrete memories

$\boldsymbol{M}$ Memory structure, containing one or more memory columns

$\boldsymbol{M_t^m}$ A memory column, $m$, remembered at time, $t$

$N$ Number of instances (in much of the empirical testing, each $i$ is a financial security) in the inputs dataset $X$,

$N(\dots)$ Normal distribution

$i$ Instance of $N$ in the inputs dataset $X$,

$r_{n(D)}$ Random integer between 1 and $D$

$\phi$ A chosen learner, such as FFNN or LSTM

$t$ Discrete time step

$T$ Total number of time steps

$\theta$ Learner parameters

$\theta_B$ Base learner parameters

$\theta_m$ Memory, $m$, learner parameters

$\upsilon_{Uni}$ Perturbation test grid, using univariate synthetic testing

$\upsilon_{cs}$ Perturbation test grid, using cross-sectional synthetic testing

$\boldsymbol{X}$ Matrix containing an input space containing $N$ instances sampled over $T$ time periods and $K$ time series.

$\widetilde{\boldsymbol{X_m}}$ Matrix containing a representation of input/training data contained in memory, $m$, from a subset of $\boldsymbol{X}$

$\boldsymbol{X_m}$ Matrix containing a raw input/training data from $\boldsymbol{X}$

$\boldsymbol{X_t}$ Matrix containing a sliding window of input data up to time $t$. A subset of $\boldsymbol{M}$

$\hat{y_{t+1}}$ Regression forecast, (independent variable) at time, $t+1$

$y_t$ Independent variable, $y$, observed at time, $t$.

Domain specific:

$\beta_{MKT}$ Market beta (finance)

$\beta_{VAL}$ Value beta (finance)

$SR_p$ Sharpe ratio (finance)

$IR_p$ Information ratio (finance)

# List of Abbreviations

**1NN-DTW** DTW based nearest neighbour classifier

**ARIMA** Autoregressive integrated moving average

**AGI** Artificial general intelligence

**AI** Artificial intelligence

**CL** Continual Learning

**CLA** Continual Learning Augmentation

**TCL** Temporal continual learning

**CNN** Convolutional neural network

**AE** Autoencoder

**ANN** Artificial neural network

**DNC** Differential neural computer

**DNN** Deep neural network

**DTW** Dynamic time warping

**ED** Euclidean distance

**FC** Fully connected

**FFNN** Feed forward neural network

**GRU** Gated recurrent unit

*i.i.d.* Independent and identically distributed

**KB** Knowledge-base

**LSTM** Long-short term memory neural network

**LML** Lifelong-machine learning

**LWF** Learning without forgetting

**MTL** Multitask learning

**ME** Mixture of experts

**NLP** Natural language processing

**NTM** Neural Turing machine

**OLS** Ordinary least squares

**PIS** Past information store

**RNN** Recurrent neural network

**ReLU** Rectified linear unit

**SAE** Sparse autoencoder

**TSC** Time-series continual learning

**TDNN** Time delay neural net

**TSC** Time-series classification

$warp-AE$ Sparse autoencoder with dynamic time warping filter

**wrt** With respect to

Domain specific:

**TR** Total return

**SR** Sharpe ratio

**RR** Relative return

**IR** Information ratio

# Contents

# Chapter 1

# Introduction

*Temporal continual learning* (TCL) is introduced in this thesis to address two important problems in machine learning (ML) research: the *continual learning* (CL) [186] problem, where knowledge of tasks is retained to prevent forgetting in the learning process, and *concept drift* (CD) [235], where adapting to temporally changing states is needed for decision-making. In addition, TCL is an extension of traditional CL (also known as lifelong-ML (LML) [219]), for temporally changing states in temporal datasets with a large cross-sectional component. This work contributes to the body of knowledge addressing CL combined with *concept drift adaptation* (CDA), which is applied to time series and cross-sectional data in a finance context.

This thesis introduces, develops and tests a TCL framework, *continual learning augmentation* (CLA), which uses an external memory structure to store parameters of learners from past temporal states, for recall and application in the future when a state appears to be reoccurring. Moreover, CLA makes four key contributions to extending traditional-CL to TCL. First, a temporal, state-based, external memory structure is developed. Second, this is used to augment well-understood base learners using TCL, including LSTM, feed-forward neural networks (FFNN) and OLS regression; thirdly, a remember-gate based on residual-change allows learner parameters to be stored with a contextual reference relating to the input space where the learner may be best applied; fourthly, a recall-gate based on time series similarity compares the current input space with memory, contextual references to recall and balance memories for use in the current period.

The TCL problem is addressed by CLA using well-understood elements of ML and data-mining research, which are re-purposed, combined, extended, and applied and then tested on complex financial datasets. A multi-point performance benchmarking framework is introduced to evaluate a TCL approach applied to a financial man-

agement task. Additionally, memory addressing that allows potential interpretation through visualisation is also introduced.

Some of the sharpest criticisms of CL research have noted that it is essential "to embrace more complex datasets" [62], which this work directly addresses, making it philosophically different from traditional CL research. Much CL research tends to use overly simplified and highly stylised datasets, with little consideration of real-world complexities [166]. This has tended to be limiting for CL research, hobbling many conclusions, introducing many unrealistic assumptions and hampering (and generally preventing) real-world applications (see [166, 144]). In contrast, this research aims to gain deeper insight by conducting more in-depth testing and analysis on more complex datasets in a more complex real-world task. As a result, this research provides highly differentiated insight, contributing to addressing the TCL problem in the real-world.

This introduction is laid out as follows. Section 1.1 introduces and motivates TCL, Section 1.2 lists the hypotheses and research questions addressed in this thesis, providing the critical path taken to achieve a real-world TCL approach: CLA. Section 1.3 describes the contributions made in this thesis. Section 1.4 details the author's related publications and Section 1.5 describes the structure of this thesis.

## 1.1 Motivations

In this section the motivations for TCL are described. Section 1.1.1 describes the motivation for traditional CL, while Section 1.1.2 addresses the benefits of extending CL to temporal datasets in finance. Section 1.1.3 expresses the motivations for TCL's as a selective marriage of ML, CL, and CDA, which is elaborated on in Section 1.1.4, as TCL is further motivated by the shortcomings of these approaches in the context of the TCL problem.

### 1.1.1 Motivations for Traditional Continual Learning

The broad motivation for CL is that achieving artificial general intelligence (AGI) requires some form of open-world CL process. Therefore, achieving AGI would be hampered, perhaps impossible, without an answer to CL [209]. By accumulating knowledge of different tasks, CL aims to avoid *catastrophic forgetting* (CF) while improving learning outcomes. The concept of a task in CL is analogous to the concept of a state in time series data (more so when the cross-sectional component at each time-point is greater, see Section 2.3).

Tasks are defined as batches of generally isolated data, associated with discrete classes, groups of classes, domains, or areas of a problem space. Intra-task data

are generally considered to be independently and identically and distributed (*i.i.d.*) whereas inter-task data are not. Time series states are defined as a concept similar to tasks, but are assumed to be associated with the changing distribution of continuous data, which tends to come with a high degree of uncertainty in the number, definition, and separation of states.

The problem of time-evolving datasets has been addressed in the *adaptive learning* literature, specifically by CDA. CDA approaches tend to focus on adaptation to changing states of a stream of data to improve forecasting accuracy. However, while states in CDA appear somewhat analogous to tasks in CL, the differences are significant and have implications for interpretability, state-based memory addressing, and CF. CDA memory structures tend to be minimalist, based on ensembles and instance-based learners with a typical onus on stream processing speed and resource parsimony. Additionally, CDA tends to focus on classification rather than regression. The onus of CDA is on fast adaptation to streaming data more than committing machine resources to remembering representations of distinct past events, supporting interpretable outcomes, or addressing state-oriented CF.

While CDA has important ideas to contribute to TCL, the focus of CL on longer term knowledge is more relevant to TCL. The motivating themes for TCL are discussed.

### 1.1.2 Extending CL to Temporal Learning in Finance

Time series are omnipresent in modern human activity, and drawing inference from data with a time dimension is a vital area of research. Time series exist in many domains, including biology [14], geology [90], space exploration [96, 239], robotics [161], and human motion [222] and are ubiquitous in finance [69, 72, 149]. Research involving time series has involved different disciplines, including computer science, statistics and econometrics. Arguably, time series analysis has previously been limited by its formative research in the 1970s (principally [19]), but more recently, the testing of potentially far more powerful approaches has indicated that deeper inference and stronger modelling outcomes are possible [63]. This provides a strong motivation to investigate TCL, which is likely to be a sequential problem in the real world.

It has long been appreciated that an approach that has a very large number of sequential computational steps coupled with an effective learning algorithm would be powerful [212]. In fact the expressive power of a ML model is highly correlated with the number of sequential computational steps that it is possible to for it to learn [244]. However, very few if any CL approaches have been applied that attempt to continually learn repeating patterns and states in multivariate time series. This is

the application of the CL approaches researched in this study.

The use of time series has been of great importance in finance, for example, in capital allocation. Sequential learning of the information also has many benefits, with many learners being exposed to CF through down-weighting older data-points or using sliding windows. Therefore, successfully applying a CL approach to finance might bring several tangible benefits. First, the financial data-set is large and complex and provides a challenging proving ground for the framework introduced in this study, which might be applied to other areas of human endeavour. Secondly, the efficient operation of markets is critical for the meritocratic allocation of capital in an economy and therefore for ensuring the basis of social mobility that will provide security for our children and grandchildren. The approaches researched in this study may facilitate a more efficient method of allocating capital.

### 1.1.3 Motivation for TCL as a Selective Marriage of CDA and CL

The key motivations for this work are threefold: first, provide open-world learning in a real-world, temporal context where states can be remembered and recalled to improve outcomes; second, to augment well-known and understood base learners with these benefits; and third, the use of CL memory should be interpretable to determine which memory did what and when. These motivations appear common to traditional ML, CL, and selected CDA approaches; however, TCL contrasts with each of these three and the marriage of these approaches in TCL must be highly selective.

1. *Machine learning: closed-world learning*: ML has been shown to be a powerful tool but generally focuses on closed-world learning, where states (tasks and/or classes) are pre-defined in the training data. Prior knowledge of states tends to be a limiting assumption, especially in time-evolving datasets, a limitation associated with CF (discussed later in this thesis).

2. *CL: task definition, task dependency and classification*: CL seeks to address CF by providing the learning process with old and new information. In many cases, CL memory structures are used to do this, including parameter sharing, but generally with the limiting assumptions that tasks are well defined, have assumed dependencies, are labelled, and are of a limited and, perhaps, known number. These assumptions significantly limit the application of CL to real-world temporal data, which tend to have ill-defined states and varying dependencies and are rarely labelled. Moreover, almost all CL approaches are

classification based rather than regression based, which deeply limits time series analysis.

3. *CDA: minimalist memory*: CDA approaches have been developed to cope with unpredictable, changing states in temporal data. However, they are usually classification based and tend to focus on streams of data. This generally requires expediencies, such as instance-based base learners and minimalist, generally uninterpretable knowledge bases (KBs; introduced in Chapter 2). These limitations are not appropriate for addressing CF using state-oriented, interpretable memory augmentation. In addition, CL approaches have made far greater advances into memory augmentation than CDA approaches, many attempting to capture the interplay between both *episodic memory* (specific experience) and *semantic memory* (general structured knowledge). In contrast, CDA can generally be described as having a minimalist form of semantic memory.

While each approach has important elements that are needed for effective TCL, the limitations of each present major impediments. If these could be mitigated and the strengths of each approach combined, it would be highly advantageous: (1), harness the power of ML and address (2) the CF problem in temporally changing datasets, while coping with (3) *concept-drift*. However, the many draw-backs of the approaches that address these problems, in themselves, motivate the development of a TCL approach. These are described next.

### 1.1.4 Limitations of CL Approaches as a Motivator for TCL

#### Limiting Assumptions

While many CL approaches have been developed, very few have been practically applied to complex real-world problems. The reason is that, with many outstanding questions in CL, studies have tended to use over-simplified datasets to ease experimentation. This is viewed as so serious that some critics of CL research have noted that the stylised nature of test datasets may have resulted in illusory progress towards CL [62].

In summary, CL approaches are typically limited by one or more assumptions that tend to conflict with the need for a real-world capability: [144, 62, 166]:

1. Tasks are easily defined (and mostly labelled),

2. No task overlap is assumed,

3. Then number of tasks is known,

4. Task dependency is assumed,

5. Performance measures tend to be reported as an average over tasks.

These oversimplifying expediencies may have influenced CL architectures, limited experimentation and conclusions, and generally prevented real-world application. Oversimplifications associated with the relationship between tasks and time appear to be the most limiting for applications to datasets with temporally changing states.

Task definition and the number of tasks to learn are assumed by many CL approaches. Popular testing datasets are both labelled and stylised, such as MNIST [134], CIFAR-10 [128] (with or without perturbation) and others. In the real-world, tasks may not be so conveniently labelled, may overlap, may be difficult to define and separate, and their number may not be known [62]. These descriptions are particularly notable for states in financial time series.

Simplifying assumptions relating to task dependencies and the predictability of change is likely to have important limitations on CL in the real-world, and in a time-varying context. For instance, real-world tasks (or states) encountered over time, such as daily versus weekly weather conditions, tend to have inter-dependencies that can vary greatly, making many parameter-sharing CL approaches, for which task dependence tends to be a condition, questionable for application in real-world tasks.

Estimating the performance of CL approaches also tends to be problematic, where many CL researchers use an arithmetic average performance metric over a number of tasks, which limits the value of conclusions that can be drawn [144]. In finance, economics, and in many other real-world domains, performance is inherently geometric not arithmetic. For example, should a house price fall by -50% and rise by +50% the resulting change is not 0%, rather it is -25%. Therefore, something as simple as a flawed performance estimation can be expected to produce a deleterious impact on the sequential accuracy of a CL approach and perhaps different assumptions and even architectures. The philosophy of the testing in this thesis is different with empirical testing being conducted on a smaller selection of more complex real-world datasets on which deeper analysis is also conducted.

### Complexity

Generalisation of CL approaches is also a concern given that CL probably requires a higher level of complexity when compared to more traditional ML approaches. The risk is exemplified by one of the most well-known memory augmented approaches, differential neural computers (DNC) [87], which has 891,000 parameters [66]. Under

the bounds of classic statistical learning theory, the training instances to support a parameterisation of this complexity would be in excess of those available in many domains. However, complexity is not a straightforward consideration. Considerations such as *VC dimension* [225] have proved controversial in deep learning research and are likely to be inadequate [15, 83], but this does not mean that ignoring the principle of parsimony would be wise.

#### Interpretability

While a successful CL approach focused on time series may yield great potential benefits across many disparate and important domains, the interpretability of how CL knowledge is accumulated and used is also an important challenge. While the importance of forming knowledge of important past events is self-evident, it is also as important to be able to explain how this knowledge can be fairly, safely, and legally remembered, recalled, and used. A CL approach should ideally be able to express how, when and which memory is affected or used. This study introduces an approach to make CL memory usage more interpretable.

### 1.1.5 Limitations of CDA Approaches as a Motivator for TCL

#### Cross-sectional data

Most streaming approaches use the passage of time to construct an ensemble of diverse, generally instance-based learners from a narrow stream of data. However, temporal data with a large cross-sectional component presents the opportunity to examine discrete cross-sectional and temporal distributions. Most streaming approaches are not intended for application to these distributions. This recasts the CD problem, where a state is observed through CD in a data stream, as the temporal change in discrete cross-sectional states. In other words it is much more like a CL task but with a temporal dimension. The literature review did not identify any streaming approach that makes a clear distinction between cross-sectional and streaming data.

#### Limited Memory

Clear commonalities exist between TCL and the streaming problem; however, streaming approaches are potentially more resource-challenged given the onus on the speed of adaptation to large volumes of streaming data, when compared to the typical

datasets used in CL approaches. This generally necessitates expedient memory usage for streaming approaches and leads to generally minimalist memory concepts.

### Catastrophic Forgetting

Catastrophic forgetting (CF) is still an open problem for CL and streaming approaches alike; however, as streaming approaches tend to have less well-developed memory structures, CF is likely to be a bigger impediment for them. Streaming approaches attempt to respond to changes in streaming data with an onus on speed of adaptation and forecasting and a general aversion to resource hungry memory approaches. This makes the probability of CF greater in these types of approaches and therefore far less interesting for cross-sectional state based TCL. Overall, no ML approach found in the literature review directly addresses long-versus-short term dependencies in time series while considering CD. This indicates that CF is still very much an open question for CDA approaches.

## 1.2 Hypotheses and Research Questions

**The hypotheses are the following:**

**A real-world problem based on a state-varying time series with a large cross-sectional component can be addressed in a sequential, open-world fashion that addresses CF.**

**This can be addressed by remembering models of temporal states and recalling these models in future periods when the input space is similar to an associated past state.**

Several related research questions are also posited:

1. Open-world TCL approach : Can time series data with a large cross-sectional component, commonly found in finance and other domains, be separated into states sequentially?

2. Remembering using residual change-points: Can abruptly changing states be identified in noisy, multivariate time series with a large cross-sectional component used as remember cues in a state based memory structure?

3. Memory recall using similarity: Can similarity measures be used to identify repeating states in noisy time series and then be used to drive memory recall cues?

4. Temporal, state based memory addressing: Can a memory structure be populated with state-based memory concepts to allow potential interpretability of which state-based memory, from which time period did what and when?

## 1.3 Contributions

### 1.3.1 Major Contributions

1. Time series memory structure: A framework is researched to build, maintain and use a state based and addressed memory structure. This is different from traditional CL, in that memory is applied to time series states rather than tasks. It is also an extension of CD, as TCL memories are related to states and can be stored indefinitely.

2. Simple learner memory augmentation: Recurrent, FFNN and OLS regression learners can be memory augmented using a generalised, deep architecture.

3. Recall-gate : Data-mining approaches for time series similarity are repurposed for use in a TCL memory gate, an approach not known to be adopted in the CL literature. This is used to drive pattern recognition in time series input data to propose memories to recall.

4. Remember-gate based on residual change: The use of *residual change* is well known in the CDA literature but is not known to be used for memory gating in the explicit, state-based external memory structures that are researched in this study.

### 1.3.2 Minor Contributions

1. Open-world learning in the real-world : Real-world applied TCL approaches are reported in this thesis which are tested on real-world temporal data problems and are effective in tested contexts. A review of the literature indicates that this is one of only a few time series applied, open-world CL approaches. (While many CDA approaches have been tested in open-world time series data, approaches with long-term memory structures that are not associated with an adaptive ensemble are not known).

2. Potentially Interpretable memory: It is possible to extract memory remember and recall information from the memory structure introduced in this work. A

visualisation to explain which memory, did what and when is developed to allow interpretation by domain experts.

## 1.4 Publications

### 1.4.1 Published

- Daniel Philps, Artur d'Avila Garcez, Tillman Weyde. **"Making Good on LSTMs Unfulfilled Promise"**. 2019. NeurIPS 2019 Workshop on Robust AI in Financial Services: Data, Fairness, Explainability, Trustworthiness, and Privacy. 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

- Daniel Philps. **"The Next Generation of Artificial Intelligence"**. 2019. Foresight: the International Journal of Applied Forecasting.

- Daniel Philps. **"Continual Learning: Will AI Give the Gray Hair a Pay cut?"**. 2019. CFA Institute: Enterprising Investor. January 2019.

- Daniel Philps, Tillman Weyde, Artur d'Avila Garcez, Roy Batchelor. **"Continual Learning Augmented Investment Decisions"**. 2018. NeurIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy, Montreal, Canada.

### 1.4.2 Working Papers

- Daniel Philps, Tillman Weyde, Artur d'Avila Garcez, Roy Batchelor. **"Continuous Learning Augmentation for Stock Selection"**. 2019. Working paper.

- Daniel Philps, Tillman Weyde, Artur d'Avila Garcez, Roy Batchelor. **"Continual Learning Slides and Warps in the real-world"**. 2019. Working Paper.

### 1.4.3 Conference Presentations

- **NeurIPS Spotlight Presentation**: Making Good on LSTMs Unfulfilled Promise. 13th December 2019. NIPSRAIFS2019, NeurIPS 2019, Vancouver, Canada.

- **City University of London, Data Bites**: AI Fund Managers: Research and Opportunities. 21st November 2019. City, London University.

- **Cambridge University: AI/ML in Finance: Is it just another bubble**: Generating alpha - AI and ML - uses in Investment Management. 8th October 2019. The Royal Society, London, UK. Cambridge University, Judge Business School.

## 1.5 Thesis Layout

The remainder of this thesis is organized as follows.

Chapter 2 reviews literature relating to CL, time series learners, and CDA. Chapter 3 addresses literature relevant to developing TCL memory addressing concepts: time series similarity and change-points.

Chapter 4 introduces TCL memory-gating, bridging the fields of ML, CL, and CDA to develop TCL remember and recall gates.

Chapter 5 introduces the CLA approach, describing how similarity driven recall and change driven remember gates are integrated into a TCL framework.

Chapter 6 reports empirical testing of CLA on complex financial management tasks.

Finally, Chapter 7 summarises the qualitative and quantitative conclusions of this work and suggests directions for future research.

# Chapter 2

# Literature Review: Continual Learning and Time series

This chapter discusses literature relating to CF, CL, and CDA as it relates to TCL.

Section 2.1 covers CF. Section 2.2 discusses CL, focusing on memory augmentation. Section 2.3 discusses time series learning paradigms, including adaptive learning (AL) and CDA. Finally, Section 2.4 bridges the themes of CL and CD to describe the TCL problem.

## 2.1 Catastrophic Forgetting

Catastrophic forgetting (CF) affects learners that are applied to sequentially evolving datasets. Furthermore, CF describes where past learned information is lost after attempting to learn newer information and has been found to have a strongly deleterious effect on artificial neural networks (ANNs) [67, 151]. This is a problem, because ANNs have so far provided the most successful applications of ML and artificial intelligence (AI) technology. Research into CL has addressed the CF problem.

While CF remains an open question, many techniques have been developed to address it including gated neural networks [95], explicit memory structures [233], prototypical addressing [211], weight adaptation [93, 213], task rehearsal [210], concept-drift based [81], generative memory orientated approaches [165] and encoder based lifelong learning [220] to name a few. This varied research area has tended to come under the unifying heading, CL, which has been described as having two main subdivisions: regularisation and memory augmentation. This study focuses on multi-column memory architectures, which are included in the above subdivisions.

## 2.2 Continual Learning

Following an increase in interest in CL, particularly over the past five years, a number of priorities have emerged, clearly distinguishing CL from other forms of ML, which traditionally have not been designed to cope with sequentially changing problems:

1. Open-world learning: learning new states (or tasks) in a dynamic environment,
2. Knowledge remembering: remembering a representation of the past state that can be used for knowledge transfer and/or strengthening future outcomes,
3. Knowledge recall: recognising the re-occurrence of a task (or state), recalling knowledge associated with similar tasks, and applying this knowledge, and,
4. Memory management: consolidating memories and forgetting memories where appropriate.

More recently CL has also started to encompass the following:

5. Forward knowledge transfer: using knowledge to help future learning,
6. Backward knowledge transfer: using new knowledge to enhance past knowledge,
7. Interpretability: interpretable learning and use of knowledge.

As researchers have addressed these initial challenges of CL, other problems have emerged, such as the overhead of external memory structures [176], problems with weight saturation [123] and transfer learning [144, 64], and the drawbacks of outright complexity [245]. While most CL approaches aim to learn sequentially, only a fraction of CL approaches have been focused on time series [111, 86, 140, 65]. How effective these approaches would be in dealing with long term CL of noisy, non-stationary time series is unclear, particularly those commonly found in finance. However, perhaps the most challenging problem for CL is stability-plasticity [91].

The stability-plasticity dilemma describes whether a system should tend towards stable, more fixed parameters over time or tend towards plasticity with more easily adapted parameters. A system with too much stability becomes unable to adapt to a changing distribution and therefore is exposed to CF, whereas a system that is too adaptable to new information could corrupt older knowledge. Stability-plasticity is a problem suffered by both artificial and human neural systems and is an open question, with much research focused on this area (eg [123]).

Task dependency assumptions have become perhaps the most important driver of design choice for CL approaches with instance-based memory chosen for expediency in streaming applications [70] and parameter-sharing-based memory in tasks with higher task dependencies [123]. If tasks are likely to be highly dependent or if tasks are easy to tell apart, parameter-sharing or a fully connected (FC) architecture may make sense because it may allow transfer learning, consolidation, and attention features. This describes many pet applications used by CL researchers (for example [191, 221]). However, if tasks are more independent or if they are more difficult to tell apart, the benefits of parameter-sharing and traditional FC architecture start to make less sense. This situation tends to describe noisy time series applications, and as we observe, an architecture that allows parameter ring-fencing (as opposed to parameter-sharing) is likely to be more stable in this environment.

### CL Families

An array of CL categories have emerged in recent years as researchers have sought to differentiate and classify their own work. First, a commonly used classification for CL approaches is as follows [47]:

1. Regularisation (see Section 2.2 below),
2. Replay based (see Section 2.2.1 below),
3. Parameter isolation (see Section 2.2.2 below).

Parameter isolation describes the work in this study, where distinct parameterisations are stored for future use. Second, a probabilistic classification is sometimes used [62]:

1. Prior-focused (see Section 2.2 below)
2. Likelihood focused (see pseudo-rehearsal in Section 2.2.1 below)

This thesis addresses a likelihood approach, determining the implicit probability of which memory to recall and when. In this thesis, the following classifications are considered to allow a detailed review of memory augmentation in relation to TCL. The remainder of this section will be laid out in this manner:

1. Regularisation (Section 2.2 below),
2. Memory augmentation
    (a) Simple memory structures (Section 2.2 below),

(b) Task rehearsal and pseudo rehearsal (Section 2.2.1 below),

(c) Memory augmented neural nets (Section 2.2.1 below),

(d) Multi-column memory architectures (Section 2.2.2 below).

Where the work in this study is focused on memory augmentation using multi-column architecture, that is discussed below.

**Regularisation**

A somewhat different approach is that of parameter-sharing, or regularisation. Forming generalisations over the same parameters comes with the potential benefit of lower resource usage along with the possibility of being able to share knowledge between tasks. Most research in this category has roots in multitask-learning (MTL) [26]. These approaches aim to jointly learn tasks using common parameters. The aims of parameter-sharing are threefold:

1. Consolidation: Consolidate knowledge of multiple tasks in a single neural architecture,

2. Multitask-learning: Inductive transfer is performed by learning multiple tasks simultaneously,

3. Transfer-learning: Use past knowledge to better learn new knowledge (and vice versa).

Using shared parameters, MTL aims to learn multiple tasks at the same time. The regularisation this induces by requiring a learner to perform well on multiple related tasks is an alternative to standard regularisation. This is known as inductive transfer, and it improves generalisation using the domain information contained in the training signals of related tasks as an inductive bias. It does this by learning tasks in parallel while using a shared representation; what is learned for each task can help other tasks be learned better [26].

Transfer learning and MTL can be achieved by fine-tuning and consolidation with examples including [46, 221]. Generally, a new task is learned while older task parameters might be used to fine-tune the learning process. This can be as simple as using older task parameters as initialisation weights for the new learner [167]. However, the motivation to transfer knowledge and consolidate it has led to deeper research into regularisation CL.

One highly influential thread is elastic weight consolidation (EWC) [123], which includes a regularisation term that forces learner parameters, when learning a new

task, to remain close to the parameters of the network trained on previous tasks. In addition, EWC has been used as the basis for a number of CL approaches [135, 221, 142, 249]. The appeal is that consolidating knowledge is likely to be more scaleable than more resource-hungry approaches, such as task rehearsal. However, while addressing scalability concerns, EWC suffers from weight saturation, leading to the phenomenon of *blackout catastrophe*. Noted in standard Hopfield networks [41], a blackout catastrophe is when network capacity is saturated, resulting in the inability to store new information effectively, with the practical implication that older knowledge is corrupted. This remains an open question for regularisation approaches. Some researchers have attempted to mitigate this issue by segregating memories before regularisation. *Gradient of episodic memory* (GEM) is such an approach [144], focusing on forward and backward transfer of knowledge. However backwards transfer of knowledge, while a principle aim of CL, risks corrupting older knowledge, particularly if the implicit assumption of parameter scaling is violated. Thus tasks are less interdependent than expected. It is unclear whether this approach to backward transfer is sensible in a noisy time series environment where the underlying function is not necessarily known, even after the event.

*Learning without forgetting* (LwF) [136] regularises predictions rather than weights using the consolidation introduced in [123]. This is carried out using a convolutional neural network (CNN), with shared weights between tasks, in which only the last classification layer in task specific, similar to Siamese nets. Unfortunately, LwF performance tends to drop when exposed to a sequence of tasks drawn from different distributions [220]. The reason is that LwF only outperforms fine-tuning when the two tasks are sufficiently related. This can be addressed through a thresholding approach, which defers memory addressing to a form of similarity gate [221]. This is likely to make the approach unsuited to non-stationary or multi-modal time series inputs.

Overall, shared parameter approaches offer huge potential for inductive transfer and the possibility of greater scalability through consolidation. Unfortunately, the assumption of task dependency is not likely to be appropriate for noisy time series domains and is therefore of questionable immediate relevance to a real-world temporal CL approach. This problem is likely to be exacerbated for domains with limited data points, where researchers have adopted more highly paramaterized, FC architectures in the belief that parameter sharing approaches would benefit from forming a generalised, differentiable form for feature extraction, fine-tuning and joint training [79].

### 2.2.1 Memory Augmentation

The CL approaches that use external memory to address CF require an appropriate memory addressing mechanism (a way of storing and recalling a memory). Memory addressing is generally based on a similarity measure, such as cosine similarity [85, 87, 168] kernel weighting [228], linear models [211], or instance-based similarities, many using nearest neighbours [112, 213, 185] and more recently, autoencoders (AE) [5, 220]. However, no single approach has been widely adopted as a generic solution. Simple memory structures tend to store examples and tend to be more resource intensive, whereas more complex CL memory structures tend to make limiting assumptions about task dependency and the discernibility of tasks. Conventional CL memory approaches are not obviously well suited to assessing similarity in noisy multivariate time series. The primary challenge for TCL memory addressing is that, although the input data are not expected to be *i.i.d.* in any CL approach, the vast majority of CL approaches expect recognisably distinct tasks to be presented sequentially. Clear delineation of states cannot be assumed in many real-world time series, particularly in finance.

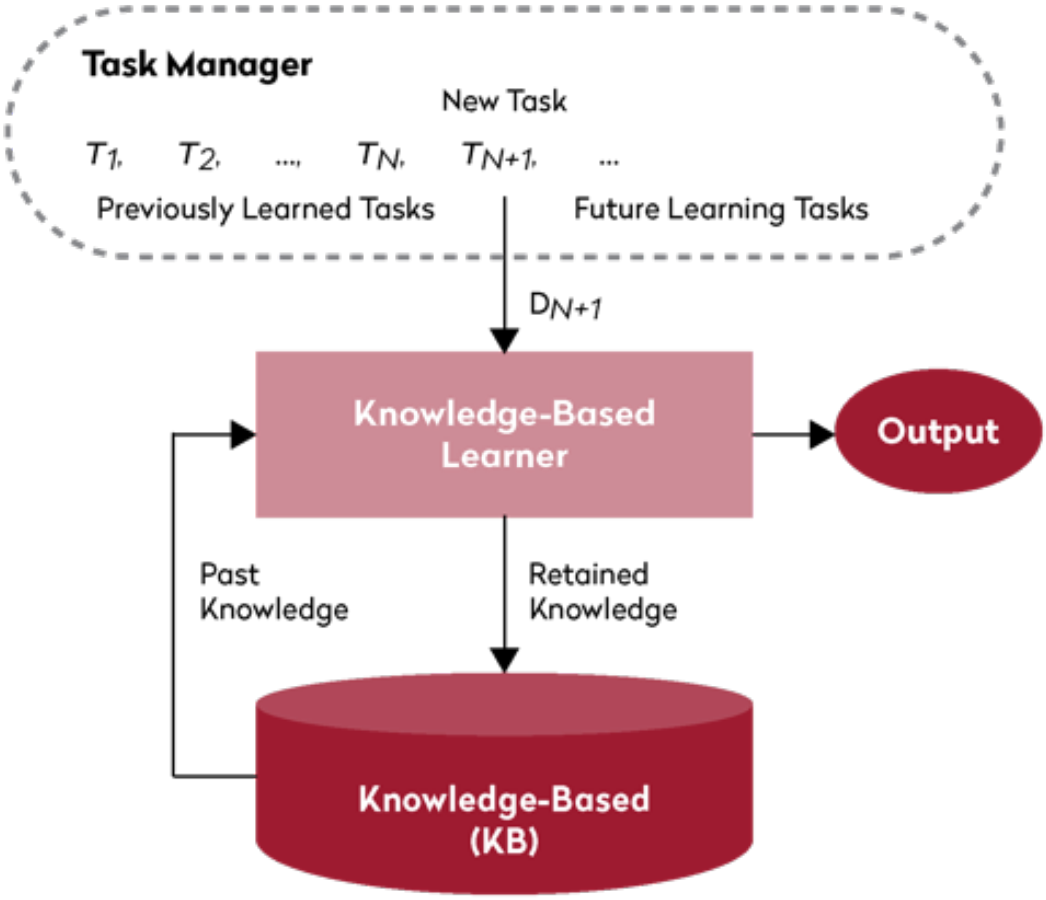
In contrast, data-mining researchers have extensively researched noise invariant time series distance measures [27], to determine similarity for time series classification (TSC). Hundreds of TSC approaches have been proposed over the last five years alone [9]. In the next section, TSC and time series similarity are reviewed and memory augmented approaches are covered. Specific to deep learning for TSC has only been extensively studied over the past two to three years and has not been thoroughly explored.

#### Knowledge Bases

ML memory structures belong to a broader super-set referred to as knowledge bases (KB). A KB refers to any knowledge storing element of a ML approach and, by definition, is an attribute of an approach that provides knowledge to enhance learning outcomes (Figure 2.1). In their most simple form, KBs are instance-based, such as k-nearest neighbours [7, 203], but have become significantly more capable in recent years, becoming stores of latent parameters [192], model parameters [32], exemplars [209], and more. This section provides an overview of the most notable KB approach; memory.

While many forms of KBs have been used, one of the more versatile and potentially interpretable approaches is known as a *past information store* (PIS) [31], referred to as simply *memory* in this thesis. The KB's of this nature contain information relating to past learning, including past results, training data, intermediate results or final

Figure 2.1: CL System: Reproduced from Chen et al. (2018) [31]




Note: A simple CL system for learning $T$ tasks, using a knowledge base

models. However, PIS have not been used in temporal CL. Other KBs are possible, where PIS data are further abstracted into meta-information. A subset of PIS KBs is referred to as memory approaches, because they aim to emulate the associative memory exhibited by humans and tend to consist of structures or parameters for storing past representations, for recall in the future. This study focuses on memory approaches for use in TCL.

### Simple Memory Structures

The first serious research into, what has now become memory augmentation, was conducted by Hopfield in the 1980s [97]. Hopfield introduced a simple, associative memory architecture designed to store patterns in low-energy states. This was followed by Boltzmann machines [2] and sparse distributed memory [114]. However these approaches suffered from a limited capacity and bottlenecks in reading and writing to memory.

Research and development of memory has tended to originate from applications for language and writing recognition which is a very different application from multivariate time series datasets. In the 1990s, researchers investigated recurrent, neural architectures with Hochreiter's development of LSTM [95], which were much later simplified to result in gated recurrent units (GRU) [35]. These approaches performed well on analysing writing and similar tasks but it is unclear whether the hard gated structures within these approaches are necessary or desirable in the context of TCL, which is examined later in this thesis.

### Task Rehearsal and Pseudo Rehearsal

Task rehearsal approaches retain samples, or representations of samples in a KB, from past tasks that are replayed as a new task is learned [210]. The intention is to attain joint training of more than one task so that CF does not occur. *Rehearsal* approaches use stored exemplars for joint training [185] while *pseudo rehearsal* [188, 189] approaches approximate past samples. However, the overhead for stored exemplars increases linearly with the number of tasks stored, and the resource and complexity overhead for pseudo rehearsal can be significant. In this sense, rehearsal approaches are also memory approaches, carrying a KB comprising samples or representations. More recently, generative approaches have been introduced [82], which have been employed in rehearsal to represent the inter-task data generating distribution. These approaches are known as *generative replay* [206, 227, 247, 190, 101, 131, 165] and allow joint training on many tasks through a process of generating samples. While these approaches have selectively been found to outperform other CL

approaches on more simple tasks [165], generative approaches have more complexity and resource overhead and additional problems, which is exemplified by the *mode collapse* problem. However, as with other forms of CL, to date, pseudo-rehearsal methods have only been evaluated using relatively low complexity datasets [166], which is likely to have resulted in limiting conclusions. Much has been written on generative approaches. Thus we direct the reader to excellent reviews and research (including [165]).

### Differential Neural Computers

One of the most technically accomplished memory augmented neural network (MANN) approaches is the neural Turing machine (NTM), developed by Alex Graves et al. [85] and later, differentiable neural computers (DNC) [87]. DNC can learn algorithmic tasks, navigation problems, question answering tasks and also learn relational data. With this flexibility come costs and constraining assumptions that are not appropriate for TCL. In this section, NTM and DNC are briefly reviewed, advantages and disadvantages are covered, with finally a summary of lessons learned that should influence the development of TCL approaches.

The NTM and DNC consists of a controller, such as a feed-forward network or LSTM, which interacts with an external memory module using a number of read and write heads [85]. Memory encoding and retrieval in a NTM external memory structure is fast, with representations via vectors being placed into or taken out of memory potentially at every sequential-step. This feature allows NTM to support meta-learning and low-shot prediction, as it is capable of both long-term storage via slow updates of its weights, and short-term storage via its external memory structure.

In terms of architectures these approaches are FC and have fully differentiable memory addressing with the ability to crystallise explicit, slot-based memories in a memory matrix and then recall a distribution over these memories. This smooth memory recall distribution (rather than using a sharp, single memory recall distribution) is advantageous because tasks (and states) are unlikely to perfectly repeat. However, the expense of memory addressing is a problem for DNCs, which requires a search over a large portion of its memory matrix, and therefore needs a large portion of the matrix to be held in machine memory. Subsequent research has attempted to address this issue [176], but DNCs and related approaches still suffer from several, important limiting assumptions in a TCL context. Firstly, DNC's assume sequential task dependency, where in the case of TCL, dependencies may vary and must also be considered in the cross-sectional distribution (this point is expanded on in 3.1). It is unclear how DNC's would deal with non-stationarities for example. Secondly, DNC's

assume absolute task dependency, where a smooth distribution of memories will be recalled to address the current task at hand. While this would also be advantageous for TCL, the training requirements for a fully differentiable architecture might exceed the data points available at a given time. Thirdly, task delineation is assumed to be equally possible for all tasks and associated memory references. In a TCL context this is rarely if ever the case where states in financial time-series are ill delineated, may overlap or may present false indications of delineation (known as $virtualCD$). These are all questionable assumptions in a real-world time series context. Therefore, it may not be a coincidence that few if any practical applications have been found for DNCs [176], and it is unlikely that a fully differentiable model of this complexity would scale up to large problems well [244].

In spite of this, DNC's remain one of the most impressive developments in machine learning research of the past 10 years. Many ideas behind DNCs will be relevant for TCL and should be used to enhance TCL approaches in the future, however as we will come on to see, the disadvantages of DNCs make it unlikely they will even be directly applied to TCL. The advantages of DNCs:

1. Fully differentiable
2. Explicit memory
3. Implicit memory (controller)
4. Sequential patterns remembered
5. Smooths sharp functions
6. CL capable: Where DNCs have been shown to learn over longer time periods than , say, LSTMs [99]

However, several important draw backs conflict with the aims of TCL and make it unlikely DNCs would provide a good template for such an approach:

1. Complexity: DNC has approximately 891,000 parameters, as applied, [66] and in many real-world data sets (and even with an aggressive interpretation of VC dimension) it is highly unlikely that there are enough data points to support this level of parameterisation. This would make DNC an inappropriate basis for application to a real-world TCL.
2. Prone to over fit: DNCs are prone to over-fitting and suffer instability throughout the learning process [99]. In a noisy real-world time series context, overfit is a

dangerous possibility and this draw back makes DNC inappropriate for use in TCL.

3. Interpretability: While Graves et al produced powerful visualisations of the smooth memory recall of the DNC, it is difficult to see how these might actually be used in more than just an indicative capacity, for example for definitive regulatory scrutiny of modelling outcomes. It would be far more appropriate to be able to identify a principle TCL memory driving the outcome in question, which could then be scrutinised, rather than to display a wide and sparse distribution. This is something that TCL aims to achieve.

4. Memory sizing has been found to be an important consideration with DNCs [157] where bad choices over memory sizing can then lead to sub-optimal performance. It seems more appropriate for TCL approaches to have a more simple and perhaps a sparser memory addressing approach which would be less likely to be as sensitive to resource or sizing issues.

5. Weak parallelism: In the same fashion as RNNs, the sequential learning of a DNC is challenging to parallelise. An alternative, multi-column memory structure, such as [191], would more easily support parallel processing and the speed efficiencies this enjoys and would thus be more ideal for a TCL approach.

6. Access of full memory required: A full memory read is required by DNC which can be expensive. It would be preferable for scalability and efficiency to be able to store and select relevant memories should the memory structure exceed machine memory resources (ie as virtual memory or as an I/O function from disk) through a more simple balancing process, before loading them and acting upon them in memory.

7. Rigid architecture: The fully differentiable nature of DNC precludes easy incorporation of domain (eg business) logic into modelling outcomes. Typically, business logic (for example) is sacrosanct and would ideally be setup as rules to guide a learning process. In DNC's case these rules would most likely need to be installed as part of a pipeline which would potentially be far less efficient. More effective would be a CL approach that could rely on a domain specific base learner, in which domain logic might be incorporated and on which memory actions might be executed.

This said, DNC offers several important guides for TCL researchers, in many ways reading like a reiteration of the motivations of TCL in this thesis:

1. Parsimony: A TCL approach should be as parsimonious as possible, at least until the initial open questions have been adequately addressed, aiming for a minimalist parameterisation.

2. Avoid variance prone approaches: A noisy real-world time series context, where the signal to noise ratio might be low, avoiding variance prone approaches is a priority.

3. Interpretability: Memory use should ideally be explainable and interpretable for domain experts.

4. Memory sizing, an important consideration for DNCs [157], has an important bearing on interpreting which, in turn, may hinder the effective sizing memory and lead to sub-optimal performance. It seems more appropriate for TCL approaches to have a more simple memory addressing approach that is less likely to be as memory hungry or as sensitive to resources.

5. Parallelism: Scalability of a TCL approach would be beneficial and an easily parallelised architecture would provide processing scaleability.

6. Flexibility of learners: To allow a TCL approach to be agnostic to the base learner used, would be highly beneficial.

7. Learning process: In a domain with limited data points, the parameters to learn should be as parsimonious as possible. This assumption might be relaxed after initial questions around the TCL problem are resolved.

In summary, as powerful as DNCs are, there are many assumptions and costs that are either not appropriate or necessary for a TCL approach. These lessons will be expanded upon in the later chapters 4 and 5 and describes the TCL approach introduced in this thesis.

### 2.2.2 Multi-column Memory Architectures

Multi column architectures have been effectively used in CL [191] and have many commonalities with mixture of experts (ME) approaches [104]. While CL essentially seeks to learn a time and task generalisation, ME attempts to find a specific selection of *specialist* models, each relating to different regions of an input space before selecting the most appropriate models for the current task [242]. This gives ME three distinctive aims that each have relevance to CL:

•**Specialisation:** individual experts specialise on smaller parts of a larger problem,

•**Partitions:** it allows the input space to be partitioned,

•**Learner flexibility:** Many different learner types can be applied.

Another interesting benefit of ME approaches is that columnar architecture naturally has a parallel structure, with obvious benefits for processing a highly parallelised architecture in a real-world setting [202]. This creates a series of questions that CL and ME share: How can one divide the input space into regions? How to specialise models in each region? How can one combine the output of each model? These problems are remarkably similar to the challenges of applying memory augmentation and cause ME approaches to use a gating function to select the correct expert. Contemporaneous research using multi-column architecture is most similar to the work in this study but with generally remaining concerns about traditional CL approaches (as detailed in 1.1.1.4)

Many different gating approaches have been used in ME approaches, but most seek to relate the error of the learner to the nature of the input space. Gaussian mixture models [183], softmax of gaussian processes [236], Dirichlet distribution [59], Dirichlet process [152], ANNs [108], max/min networks [57], and probit function [77] have all been tested. Furthermore, ME has also been applied to huge conditional computation problems, incorporating billions of parameters [202] (although this complexity is self-defeating for an approach that principally intends to subdivide a problem for the sake of parsimony). Most notably, Aljundi et al. has come closest to bringing ME and CL together [221], using the *similarity* of the input space as a method of indexing experts, using an $expert - gate$. However, the generally *sharp* model selection of ME and the lack of a time-dimension or memory concept (i.e. remembering, forgetting, recalling) has limited this thread of research in its application to problem spaces that change and repeat over time.

*Progressive neural networks* [191] are one example of a recent columnar CL approach, proposed to store pre-trained learner parameters in a columnar architecture. This columnar architecture initially allows distinct parameterisations between learners but includes lateral connections for transfer learning between tasks or columns. However, this distinct memory architecture would not be advantageous in tasks where establishing dependencies between memories has a higher error, meaning that transferring or consolidating knowledge would have a higher probability of corrupting older knowledge. In TCL problems, where the signal-to-noise levels of a real-world time series make determining dependencies difficult in some cases, the limiting assumptions of task dependency and discernibility could be problematic.

Next, this study examines the different CL architectures for their potential benefits and costs to a noisy, real-world, applied TCL approach. In this case, a multi-column CL memory architecture resulting in parameter ring-fencing, rather than sharing using lateral connections, is likely to be more appropriate.

## 2.3 Time series Learning Paradigms

Time series are ubiquitous in modern human activity and much intellectual capital has been invested in better understanding the methods of associating, classifying, and modelling this form of data. They are extensively found in domains ranging from biological taxonomy and video streaming to social media time lines and finance.

Time series are used for forecasting, classification, transformation or interpretative modelling to describe a temporal process. Three major questions need to be addressed by researchers of any discipline when modelling time series:

- How can one deal with discrete versus continuous time? [160]
- How can one train a time series approach: sequential or batch training?
- How can one ascertain and represent temporal dependencies?

Each of these elements is discussed in turn.

While a rule of thumb for time series sampling from continuous series has been long known [160], many time series are not sampled at the appropriate frequency, owing to costs and practicability. An example is gross domestic product (GDP), where periodicity is lower than might be desirable owing to the cost and practicability constraints on data collection. This section reviews time series approaches, defining their functional form and discussing how the shape of a time series can influence the choice of learner: sequential or sliding window, and whether cross-sectional modelling is more appropriate than time sequential modelling.

### Time series Functional Form

Multivariate time series, which are commonly found in databases, follow the general form:

$$(x_{t=0,k,i}, x_{t=-1,k,i}, \ldots, x_{t=N,k,i}) = \mathbf{X}, \tag{2.1}$$

where $t$ is a discrete time-step of $T$ total steps, $k$ is a single time series of $K$, where $K = 1$ for univariate time series and $K > 1$ for multivariate time series. $i$ is an instance of $N$ instances, which may be sampled at each time-step and for each time series $K$.

A time series of $T$ time-steps, $K$ series, and $N$ instances can also be represented using matrix notation, $\boldsymbol{X}$, the notation used in this thesis. An example time series is a univariate arrhythmia series, where $K = 1$, and $T$ is the number of discrete samples taken over time for a patient $i$. In addition, $T$ is considered the *time* dimension, stretching over time. The number of patients ($N$) that may be sampled at each time slice, is considered the *cross sectional* dimension of a time series. As we will observe, the relative size of $T$ wrt $I$ wrt $K$ will influence the choice of learner and training approach.

Time series are also generally assumed to be generated by the process, $P$, of a generally hidden, underlying function:

$$P \rightarrow \boldsymbol{X} \tag{2.2}$$

where $P$ is generally assumed not to be a random process, where the main aim of time series analysis tends to be to functionally approximate $P$ in some way. This leads to the question of which learner one might use to achieve this aim, which is most heavily influenced by the shape and availability of the time series data.

### Sequential versus Cross-sectional Data

While the nature of a time series process influences the choice of learner and training approach, data availability is, in practice, almost as important. Time series data defined sequentially through time (ie wrt $T$), in terms of the number of time series involved (ie wrt $K$), and cross-sectionally wrt $N$. Generally, if $T \ggg N$, the onus is placed on modelling temporal dependencies before cross-sectional relationships. This would prioritise finding temporal dependencies before relationships shared across instances. Where $T \lll N$, the onus might change to finding relationships across instances rather than prioritising temporal dependencies. These two extremes might be represented by a sun-spot time series, where $T \ggg N$ when compared with financial credit-scoring, where $T \lll N$. In the first case, there is only one sun, and while different observations may be collated, it is assumed only one accurate and comprehensive sun-spot record exists. In this case, assuming $K = 1$, all inference must be made temporally. Where $N = 1$ by choice or by necessity (as in this example), it is called stochastic training. However, in the case of credit-scoring, where $K > 1$ and $N > 1$, batch training can be used, where many instances are used in training. This is an important distinction, as the characteristics of borrowers in a single time slice could be more important than the temporal dependencies of an individual borrower. This can be described as the subtle difference between a CL problem, where cross-sectional past states should be remembered, and a simple time series

problem, where time-sequential dependencies should be addressed. Depending on whether the informational content is cross-sectional versus time-sequential data, this would influence our choice of learner, as discussed in the next section.

### 2.3.1 Time series Learners

This far, ML researchers have proposed many approaches for time series modelling including neural networks [159],expectation-maximization [78], support vector regression [158], Gaussian process regression [231], kernel regression [16], Gaussian mixture models [120], kernel PCA [180], recurrent approaches [13, 122, 24, 6] and others. Most of these approaches forecast a single point ahead, use a sliding window of input data, and do not fully address long-versus short-term temporal dependencies. Additionally, in many cases, it is unclear whether these approaches can effectively deal with changing states in a time series, also known as CD[196]. Furthermore, a family of approaches called *adaptive learning* that have been developed to cope with this. These approaches aim to update predictive models as they step forward through time.

Time series approaches can separated into four modelling paradigms, each with many learners.

- Traditional time series models: Such models tend to assume a certain distribution of the data (eg linear regression),
- Sliding window: Dividing a sequence into a series of discrete steps allows the application of a range of learners (eg FFNN),
- Recurrent: The learners attempt to model a sequence as a sequential process (eg LSTM),
- On-line: Generally these are stochastic and batch approaches to learning.

However, some major common challenges for all time series and sequential learning problems [49]: the modelling of long-versus short-term temporal dependencies, including in cross-sectional data. Although each paradigm has provided very powerful learners for a range of problems, no single approach adequately deals with this distinction between long and short-term dependencies. This is a manifestation of CF specific to time series modelling. In this section, an overview of time series learning paradigms is presented.

### Traditional Time series Models

Traditional time series approaches tend to assume that time series conform to a specific distribution, most commonly a normal distribution, and seek to model an underlying time series process. Probably the most popular time series modelling approaches are OLS regression (e.g. [61]) and autoregressive integrated moving average (ARIMA) [20]. The popularity of these models is owed to their ease of use, flexibility, and transparency, and the availability of an extensive statistical toolkit, such as the box-Jenkins methodology, for model selection. However, with transparency and flexibility comes limitations. For almost all approaches, including ARIMA and OLS regression, the assumption of linearity becomes unrealistic in many practical situations. To overcome this drawback, nonlinear models have been proposed [250] but these are more complex to implement and interpret [3], and in this sense, they might present a poor compromise.

Also available are online learners, referred to as stochastic learners, when they operate on only a single example at each iteration [42]. These learners tend to be faster to train but tend not to support cross-sectional data and are inferior to batch learning approaches [200]. As a result of these drawbacks, this study does not focus on these approaches.

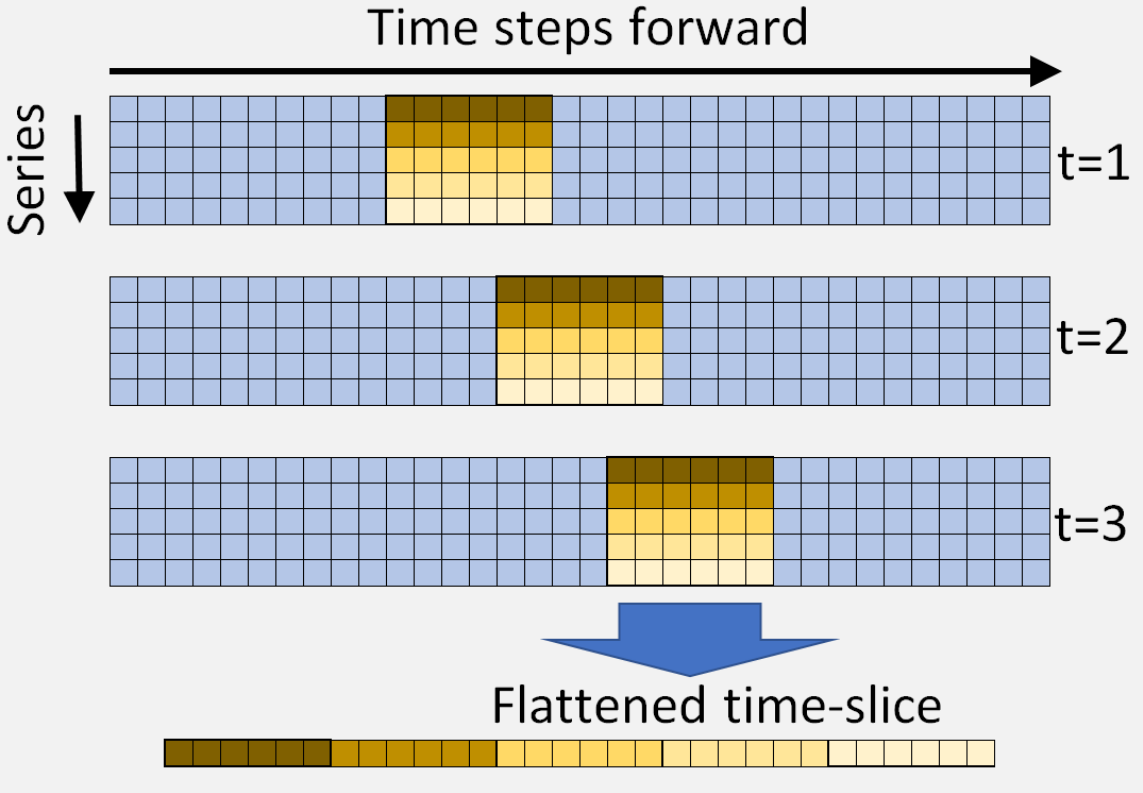
More highly parameterized ML approaches have also been applied to time series modelling. These approaches present the opportunity to move away from traditional, parametric time series approaches and to lever the higher complexity of ML approaches to draw deeper inferences from time series data than may be possible with traditional approaches.

### Sliding Window

Sliding window describes a dynamic, step-forward process where, generally, data with a fixed temporal range are used as the input for a learner, for example an FFNN, and where every input position represents a fixed time lag from a specific time series (Figure 2.2). This window steps forward in discrete time with the learner, generally training and forecasting at every step. Sliding window approaches have the advantage that they can use one of many well-understood and well-researched learners and are relatively straight-forward to apply and interpret. The key disadvantage is that a fixed time window must be selected.

Sliding windows can be of a) a fixed size, which generally limits temporal inference to within the window, b) an expanding size, where the start period is fixed while the window continues to expand, which can come with a significant resource overhead, or c) a variable-sized sliding window, sometimes used in adaptive ML. The major

Figure 2.2: Time Series Paradigms: Sliding Window




Note: A sliding window steps forward over five time series in discrete time. The window is flattened to into an input vector where each input position relates specifically to a series and lag combination.

shortcoming is that an arbitrary window length must be selected and, over time, all information that moves out of this sliding window is forgotten. Serious questions have also been raised about some applications [137].

Many different learners have been applied to sliding windows in sequential learning tasks, such as OLS regression, FFNN, [172, 171], and support vector machines (SVM) [218], and more time series specialised approaches, such as time delay neural networks (TDNN) [230], CNN [256, 12].

Although specialised learners, such as TDNN and CNN, seem appealing, they have greater constraints than generalist learners.

Moreover, TDNN and CNN approaches use max-pooling to learn latent representations of subsequence features. However, the down-sampling required by such

approaches imposes additional temporal constraints requiring further setup choices to be made. This is likely to be a serious problem because, in the case of CNNs, if the down-sampling reduces the dimensions of the input space too much, longer term features might be missed and if the dimensions are not sufficiently reduced, over-fit might result. In the case of TDNNs, time dependencies generally need to be chosen, so that the input data can factor in temporal dependencies as the sliding window passes over the input data. Clearly, imposing these dependencies is far from ideal, as a time series approach should principally be learning these for itself.

### Recurrent Architectures

Recurrent neural network (RNN) approaches, such as LSTMs, address the arbitrary window size concern of sliding windows but have to model potentially complex cross-sectional and short and long-term temporal relationships sequentially. Additionally, whereas a simple FFNN has complexity relating to its *feedforward depth*, RNNs have two additional dimensions of complexity; *recurrent depth* and *recurrent skip coefficient*, giving RNNs three degrees of complexity [251]:

- **Feedforward depth:** Similar to FFNN, this represents the complexity of the non-linear input-output transformation. In an RNN, input-output transformation has to be dealt with in combination with two other degrees of complexity.
- **Recurrent depth:** The average maximum number of nonlinear transformations per time step.
- **Recurrent skip:** Skip connections across multiple time steps may help improve the performance on long-term dependency problems.

Although RNNs are Turing equivalent, performance in long-term memory tasks has generally been poor [126], indicating that despite their promise, RNNs are no panacea for time series modelling. Early research has reported that RNNs were able to learn short-term patterns but had difficulty capturing global behaviour [154] because the fraction of the gradient due to information from $n$ time steps ago approaches zero as $n$ becomes large [18]. Specific adaptations have been required to allow RNNs to perform well over long and short-term dependencies [29]. This has been addressed by some researchers, for example LSTNet adds a recurrent-skip layer [130], but most add additional challenges (in this case the skip length of the recurrent-skip layer must be manually tuned in order to match the period of the data).

Furthermore, LSTMs are probably the most popular form of RNN. However, the performance of LSTMs verses that of sliding window approaches has been mixed.

On the one hand, researchers have shown that LSTMs are able to solve time series problems that sliding window approaches, such as FFNNs, have not. On the other hand, sliding window approaches have outperformed LSTMs on seemingly more simple time series problems [76]. In addition, RNNs have been found to be poor at representing long term dependencies in time-series [34], which is a major shortcoming when applied to the TCL problem.

Another major draw back of RNNs is their interpretability. Explaining how an RNN achieves temporal and cross-sectional functional approximation, in terms of the parameterisations is challenging. Specific approaches have been developed to attempt to make RNNs more interpetable [89]. Moreover, recurrent approaches are exposed to exploding gradients [94].

### Time series Data Mining in the Context of Memory

Considering time series data-mining techniques in the context of memory augmentation and CL is relatively novel. It is also not clear that any current thread of research associated with memory augmentation and modelling can be appropriately applied to multivariate time series modelling in a financial context. Some fall short in real-world challenges [176], some in their application to complex multivariate time series, and some when considering CF [67, 151], which can result from an approach without an explicit memory of a past event and its outcome (motif discovery and data mining). While memory augmented models are well researched, development has tended to originate from applications to language and writing recognition, which almost certainly have a far more regular distribution of outcomes than many multivariate datasets (eg financial time series data). Secondly, many sequential learners have been repurposed as time-series model, such as recurrent learners, convolutional architectures and attention mechanisms, as we will come on to see this comes with disadvantages.

Approaches such as LSTM [95], and GRU [36] perform well on handwriting and similar associated tasks, but whether the "hard" gated structures within these approaches are necessary or desirable for time series modelling is unclear. Memory augmented modelling, after the NTM and DNC [85, 87], is a closer cousin of TCL, with a fully differentiable architecture and the ability to crystallise explicit memories and then to retrieve a distribution over these memories. One of the disadvantages of this and associated approaches, which researchers have attempted to address in subsequent research [55], is the requirement to search over a large portion of the systems's "memory matrix", requiring a large portion of the matrix to be held in "machine memory". Notably, few if any practical applications have been found for neural Turing machines at time of writing [55] and it is unlikely that a fully

differentiable approach of this complexity would scale up to large problems well [244]. Although not a memory modelling approach, and therefore exposed to CF [67, 151], ME [104] provide a blueprint to combine the outputs of different models in a highly parallelisable architecture, which can (and has been) deployed in a real-world setting [202]. Contemporaneous research, using "multi-column architecture" is most similar to the work in this study [191] with the problem of parsimony remaining: when should memory be efficiently crystalised? When should one parsimoniously retrieve memories? How can one balance these memories? This study seeks to address these three key questions.

While simple Euclidean distance (ED) offers a rudimentary approach, it has a high sensitivity to the timing of data-points, which has been addressed by dynamic time warping (DTW) [193]. However, DTW requires normalised data and is computationally expensive, although some mitigating measures have been developed [254].

### 2.3.2 Deep Neural Networks as Time series Models

While deep neural networks (DNNs) have been very successfully applied to such challenges as computer vision and other snapshot-based challenges, considerably fewer studies have applied DNNs to time series forecasting. While DNNs promise the capture of non-linear interdependencies in time series, their successes have not been easy to replicate in time series modelling [217, 63]. It has become clear that fairly discrete ML approaches are required to deal with time series.

In addition, DNNs can be affected by perturbations in input data [217], which a human viewing a visualisation of the data would hardly notice, but which can affect forecasting outcomes. This strongly indicates that a naive application of DNNs to noisy (ie perturbed) time series would be misguided. Rather than functionally approximating an underlying time series process, a more straight forward application of DNNs is TSC, an important field for research that is relevant for this study and is covered in the next chapter.

It has long been appreciated that a learner with a very large number of sequential computational steps would be immensely powerful [212]. Put another way, the expressive power of a learner is highly correlated with the number of sequential computational steps that it is possible to for it to learn. In this important regard, ML approaches have greater potential than traditional time series models, as the deeper a neural network is, the greater the number of sequential computational steps [244]. However, it is not a straightforward task to apply deep learning to a noisy, perturbed dataset [197]. Now reviewed are some selected deep learning approaches

that have shown promise as time series learners.

### ES-RNN

One of the most successful recent ML approaches for univariate time series modelling is *exponential smoothing-recurrent neural networks* (ES-RNNs),developed by Slawek Smyl et al. at Uber, which won the M4 forecasting competition in 2018. The ES-RNN is a hybrid approach using traditional exponential smoothing coupled with RNNs. However, the M4 competition is applied to generally univariate time series forecasting problems, with few if any cross-sectional components in the test datasets. Nonetheless, augmenting an approach such as ES-RNN with an external memory structure would be an interesting thread for TCL research.

### CNNs

Convolutional networks (CNN) [133] have been re-purposed for CL, developed for sequence learning and applied to time-series data. CL approaches, such as LwF and derived approaches (see 2.2) use CNNs highly selectively in their architecture. CNNs have also been extensively applied to sequence modelling [198, 92]. This has included use in speech recognition [230], part-of-speech tagging [39], semantic role labelling [195], sentence classification [113] and document classification [253]. Convolutional architectures have also been found to reach state-of-the-art accuracy in sequential modelling, when compared to RNNs [11]. More recently researchers have aimed to combine the sequential learning of RNNs with the encodings of CNNs [204, 21] and this approach has also bee applied to time series [23]. These approaches generally replaces the FC layers in a LSTM with convolutional layers, to gain the benefits of convolutions in the process of recurrent learning. These and similar approaches have produced good sequential learning results but retain many of the disadvantages of both recurrent and convolutional approaches, which makes them less appropriate for an application to TCL in a noisy real-world context. Specifically problematic are the three dimensions of complexity recurrent learning entails that lead to issues regarding explainability, problematic for TCL and which are enumerated in 2.3.1 but CNNs come with their own set backs also.

CNNs have been extensively used for TSC [44, 232, 132, 139], with encouraging performance. However, beyond TSC, temporal learning in a noisy problem space is unlikely to be as appropriate using CNNs for two main reasons:

1. Tuning of hyperpraamters is non-trivial: CNNs require three hyperparameterrs to be tuned, in addition to conventional ANNs: filter size, pooling-size and

training epochs. The choice of these hyperparameter values can have a substantial impact on performance and thus the noisier and more temporally variable the time-series subsequences being considered the greater the possibility of overfit or bias.

2. Max pooling: Use max-pooling to learn latent representations of subsequence and recurrent features, requires down-sampling which in turn imposes additional temporal constraints. This is likely to be a problem for TCL, because in the case of CNNs, if the down-sampling reduces the dimensions of the input space too much, longer term features might be missed and if the dimensions are not sufficiently reduced, over-fit might result.

In a noisy real-wold time series context where data points are limited, temporal dependencies are indistinct and false-alarms are probable, it is likely that CNNs would prove challenged. Some of the shortcomings of CNNs have been addressed by attention which is discussed next.

**Attention**

Attention mechanisms were developed [10] to draw inference from hidden units in existing machine learning approaches and have been developed on natural language processing (NLP) problems but have also been applied to TSC [130].

RNNs [205] and CNNs [248] have both had attention mechanisms applied. Unfortunately, for reasons discussed in this chapter, both RNNs and CNNs are less appropriate architectures for TCL but an appropriate application of attention is likely to be beneficial, and should be investigated after TCL's initial research questions have been addressed.

Attention as a process is divided into four steps [10], 1) encoding, 2) learning attention weights, 3) creating context vector, 4) decoding. The aim is to determine a weight for different elements of, generally, a hidden layer and thus learn a latent representation of how these hidden units should be attended to (ie how much weight is given to them in the final modelling outcome). In this sense attention is a broad concept and there are clear applications for it in TCL, but current implementations have major draw backs that make their application to TCL problematic.

One important example of an attention based approach is the transformer network [226], which avoids convolutions and recurrence. This inspired a number of progenitor approaches which have been applied to NLP, such as BERT, [48], GPT [175] and XLNet [240]. Of these BERT, for example, is described as a self attention approach (after [33]) where an attention mechanism is conducted on the inputs. Attention

has also been widely used with RNNs for sequence learning tasks. These generally aim to use attention mechanisms to find dependencies in sequences over the long or short term [10]. Attention offers three distinct advantages over recurrent learners: 1) arguably better learning of long and short term dependencies; 2) parallelisable architecture, and; 3) explainability using, for example, attention heat maps.

However, attention applied to time series has been complicated owing to: 1) the limitations of the input of the base learner (usually an RNN), 2) these approaches have generally been used for classification tasks rather than time series forecasting.

Several approaches have attempted to apply attention mechanisms specifically to time series forecasting, again, generally using RNNs. One example is DA-RNN [174] which has a two stage approach, first selecting input series in the encoder, while learning a temporal attention mechanism in the decoder.

While results have been impressive using attention mechanisms, the use of recurrent learners to apply attention mechanisms to time series problems has major draw backs for TCL. In short, attention mechanisms are only as good as the learner they are attending to and in the case of RNNs, the numerous disadvantages of these learners may outweigh the benefits of attention in an application to TCL.

Specific disadvantages in applying attention to RNNs follow:

1. Several parameters require tuning: The number of time steps to use, the size of hidden states for the encoder and the size of hidden states for the decoder.

2. Time series noise: Simple attention approaches with multiple variables in each time step may fails to ignore variables which are noisy in terms of forecasting utility [205].

3. Averaging across time-steps: typical attention mechanisms average information across multiple time steps which might result in a failure to detect temporal patterns useful for forecasting.

In spite of the draw backs of attention, the dividing line between success and failure of an approach is likely to relate the choice of base model and the application. It is therefore likely that a selective application of attention to a TCL approach would prove useful and should be investigated, but not before the more immediate problems of TCL, posed in this thesis, have been addressed.

### 2.3.3 Concept Drift Adaptation

In the literature, CDA has been studied in different areas of ML and data-mining research including pattern mining, data stream mining, information retrieval, and

Table 2.1: Adaptive Learning Strategies

| | Step change | Evolving change |
|---|---|---|
| Single model | Detectors | Forgetting |
| Ensemble | Contextual | Dynamic ensembles |

recommender systems [258]. Moreover, CDA is an advanced time series learning paradigm, developed to adapt a temporal learning process to CD. It has many commonalities with CL but different priorities:

1. Incremental learning: learn from training data, streamed continuously.
2. Detecting change: determine changes in the underlying data process behind this stream of data, generally referred to as CD.
3. Responding to change: the onus is placed on the speed of adaptation and forecasting.
4. Diversity: models should be preserved for future use.

These priorities have generally resulted in approaches that are dependent on ensemble effects to retain and combine models for incremental learning. Some approaches have touched on the more committed knowledge management problem of CL but rather than learning individual tasks or states, CDA can avoid this issue and generally uses the passage of time to construct an ensemble of diverse, normally instance-based parameters. In addition, CDA approaches are associated with detecting and responding to change in streaming data, with many having the onus on speed of adaptation and forecasting. In this sense, CDA does not directly address the CF problem that CL seeks to address.

The following are categories of adaptive learning strategies based on [258]:

Many different AL approaches been developed, including those based on decision trees [224], clustering [43], association rules [177], support vector machines [125, 246], random forests [1], neural networks [238] and many others. These approaches are focused on dealing with CD, but notably only a small minority of approaches use the idea of a reoccurring concept (eg 10% of approaches reviewed in [102]), a minority are based on neural networks (eg 8% of approaches reviewed in [102]) and almost all are classification based [102]. In addition, simple memory-based CDA approaches have also been researched [73, 215], but all reviewed approaches fall well short of state-distinct memory to allow the use of reoccurring concepts for regression tasks.

### Adaptive Learning

Aiming to optimise the expense of computational resources against predictive accuracy, AL is an advanced form of incremental learning [74], generally supervised and typically designed for higher-speed processing of data streams. In addition, CDA approaches are the most advanced form of AL and are sensitive to changes between dependent and independent variables over time, such as CD [196, 235].

Furthermore, AL procedures can be conducted as an online process or a batch process following the general form below:

1. Forecast: New input data arrives, $X_t$, and a forecast is made, $\hat{y_{t+1}}$, by a learner, $\phi_t$.

2. Assess: Over time, once $y_{t+1}$ becomes observable the error of $\phi_t$ can be assessed.

3. Adapt: A function of the error is used to update $\phi_t$ is updated to $\phi_{t+1}$.

Incremental or online data-mining methods, such as [223, 73], continuously refine a model as new data arrives, attempting to achieve the performance of a batch learning but with streaming data.

However, these approaches tend to focus on accuracy at the current time point, rather than on a motivation for deeper inference from past states or episodes. For example [52] uses an incremental decision tree algorithm for streams comprising discrete data with this focus in mind. Again, this results in a focus away from the TCL's priorities, of distinct state based memory concepts that can be explained.

### Concept Drift

The term CD describes how a stream of data can change (or appear to change) over time owing to non-stationaries, multi-modalities, or similar aspects (known as regime change or state change in econometrics and finance). This describes *real* CD [234], where an underlying function undergoes a change, $P(X_t) \neq P(X_{t-1})$. However, CD can appear to be occurring owing to more spurious effects, such as sampling, weak model generalisation or changes in the local feature space. This is known as *virtual* CD [194] and occurs where a function, $f$, a functional approximator for the true underlying distribution, undergoes a change, where $f(X_t) \neq P(X_t)$.

### Concept Drift Adaptation

By addressing CD, CDA improves on the idea of AL [196]. Approaches have been developed to deal with CD which have been applied to smart grids [153], email

classification [25], industrial processes [169], finance [216], and more. Moreover, CDA aims to identify changes in the input state to then illicit a response. This is different from the aim of TCL, which aims to learn a state (or task) to avoid CF. The resulting approaches are therefore quite different in priority and design.

Simple CDA approaches use sliding windows to assess change, being extended to an adaptable window size [235, 147, 124], even using support vector machines to adjust size [125]. Memory concepts are limited in the CDA literature and extend to *gradual forgetting*, where input samples are retained after the sliding window has passed over them and are generally decay-weighted (ie down-weighted based on their age). Linear and exponential decay have been used [127, 124]. Generally, KBs consist of either using example instances, exemplars that are stored and used in training, or a sliding-window approach where a moving window of data is used to train and adapt the approach. These approaches tend to be limited in scope and power because accommodating large volumes of streaming data in machine memory may not be feasible in some areas of application.

### 2.3.4 CDA Memory

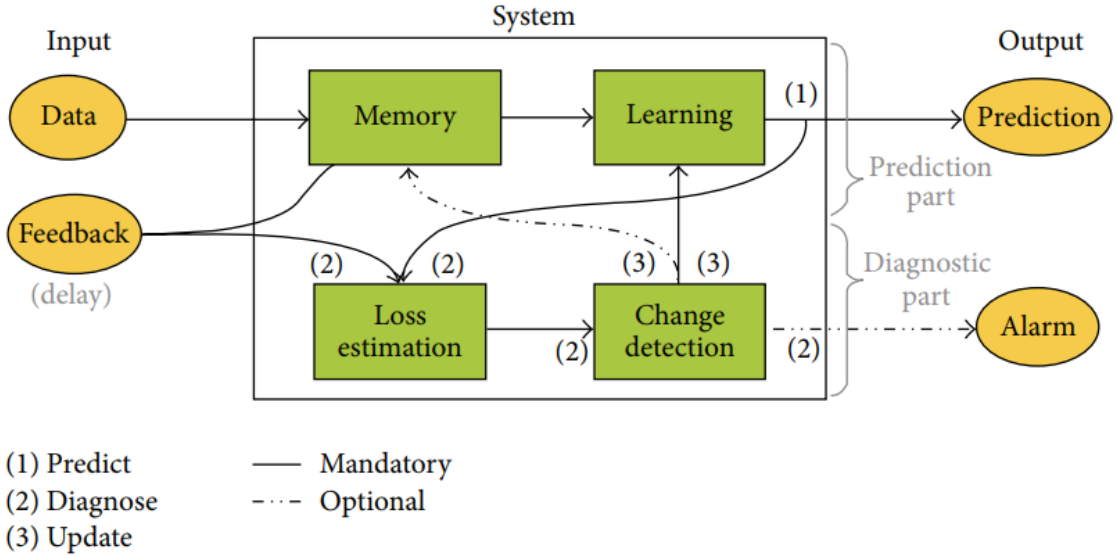
While many CDA approaches are based on a single learner and tend not to have memory structures, because these approaches seek to adapt the current model while older parameterisations are discarded, more advanced forms have KBs. Using exemplars and instances [70], CDA memory is generally ensemble based, aiming to adapt to change quickly. Model-repositories have also been used [81], where parameterisations are saved. In addition, CDA memory is discussed in this section regarding how it differs from CL memory in purpose and function.

While there are similarities to CD and TCL problems, the use of memory in either system makes a key distinction. Regarding CD, first, separating the influence of past knowledge on modelling outcomes is likely to be difficult in a typical CDA ensemble. Second, this means interpretability is likely to be affected. For example, which parameter, remembered during which period influenced which modelling outcome? Third, as these systems tend to focus on the current forecasting challenge, the retained knowledge is less about understanding the temporal interaction between past and current states and more about adapting to the current environment.

#### Ensemble Memory

Generally, CDA ensemble memory is motivated by both accuracy and diversity among the ensemble of learners [53, 129], a slightly different motivation compared to CL memory. This generally results in CDA memory structures being far more simple

Figure 2.3: CDA Memory Reproduced from Gama et al. 2014




Note: Reproduced from [70]

than CL simply because of the economies that are required in processing streams of data.

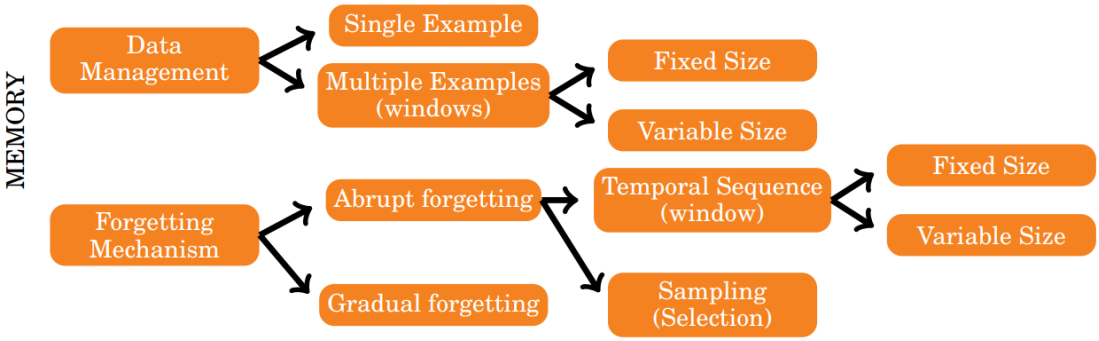
On the one hand, AL has resulted in simple approaches to adapt the learning process, such as self adjusting memory k-nearest neighbours (SAMkNN). This example simply weights a short-term learner versus a longer term learner based on the current period performance [145].

Most approaches are purely instance-based, evolving a CDA ensemble approach, such as changing rules or adding exemplars, over time. Research that is contemporaneous with this study provides a good example of how these systems work [215] and why the imperative is different for TCL. In this instance-based system, the balancing decision and decision to add a decision tree to the ensemble is based on *diversity* using the Yules Q-statistic [243]. This has proved to be very effective in certain tasks but has several problems.

### Model Repository

One close comparison of the aims of TCL with CDA is provided by *model-repository* approaches (eg [81]). While different forms of model-repository memory exist in the

Figure 2.4: Idealised CDA Architecture Reproduced from Gama et al 2014




Note: Reproduced from [70].

CDA literature, most are, again, for simple models to aid forecasting more than to understand and interpret the association between the state they represent and the period in which they are applied. As one example, [81] uses naive Bayes learners, which are stored in a model repository and reused based on a contextual cue.

There are clear commonalities between the TCL and CDA however, CDA memory approaches are focused on adapting a learner for drift in a stream of data, which is considerably more resource challenged than many CL approaches. Therefore, CDA memory approaches generally necessitate expedient memory usage.

### Architectures Compared: CL versus CDA

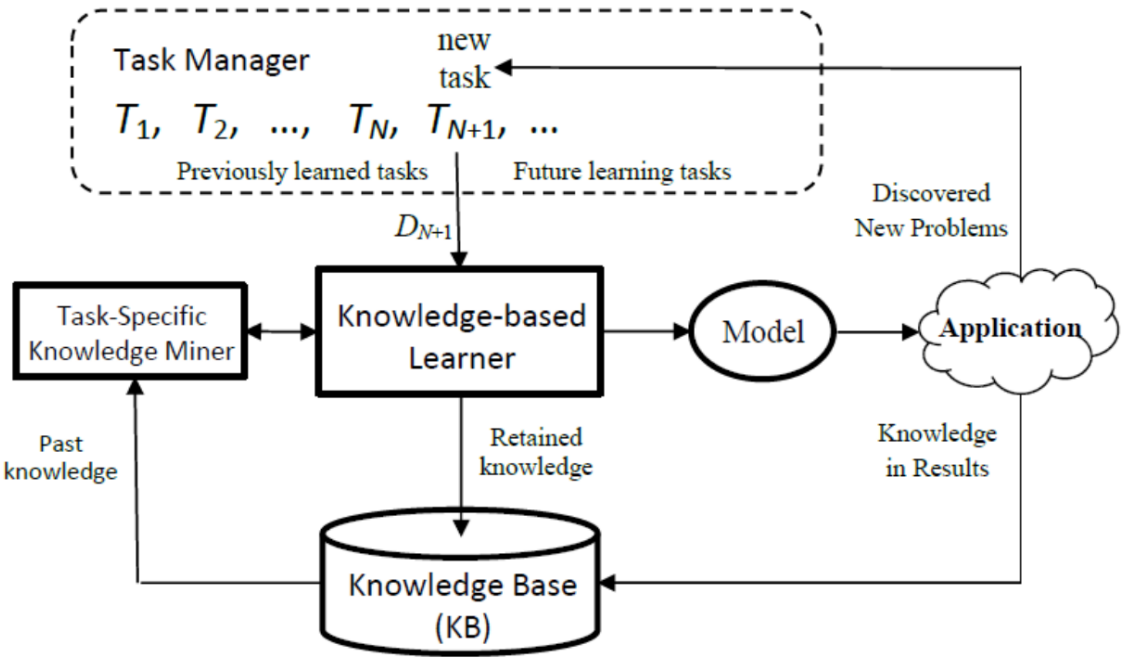
On a trivial level, CDA and CL architectures look similar. Figures 2.4 and 2.5 show generaslied CDA and CL architectures respectively. Both architectures show a memory/KB and both show a learning element associated with memory, influencing outcomes. This is where the main commonalities end:

1) CL KB learner: While CDA favours AL, CL places an onus on learning tasks and identifying previously learned tasks to assist outcomes and learning.

2) Change detection: This is a critical part of CDA architecture. In addition, CL generally does not place the same onus on understanding task discernibility or dependency, owing to the limiting assumptions made by many CL approaches.

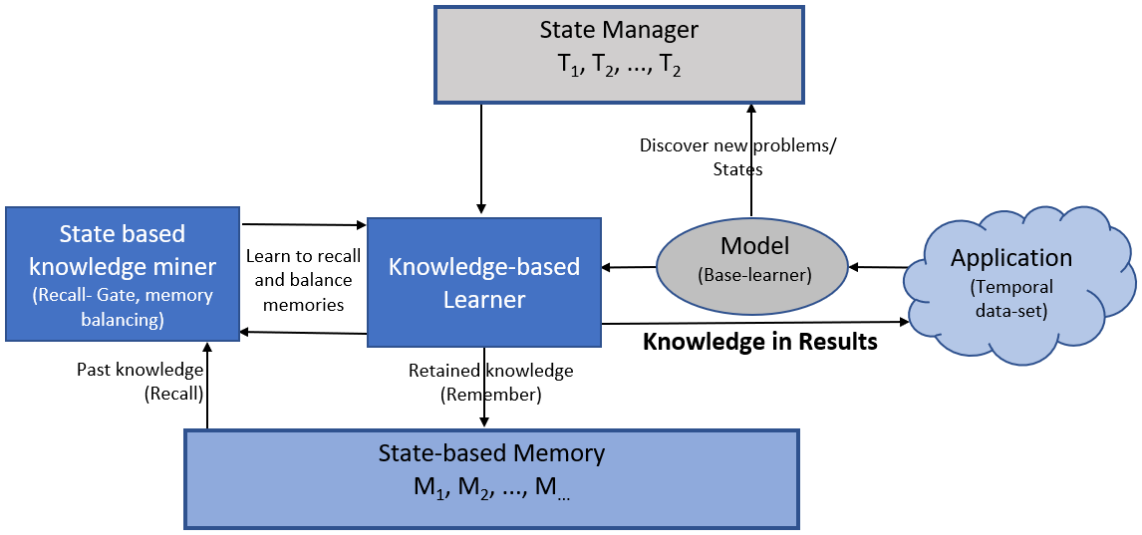
3) Task manager: A CL system identifies new tasks or repeats older tasks, generally using task-based memory. However, CDA tends to balance knowledge to adapt outcomes, with generally less distinction between states and less understanding of individual states.

Figure 2.5: Idealised CL Architecture Reproduced from Chen and Liu 2016




Note: Reproduced from [31].

Figure 2.6: Idealised TCL Architecture




Note: TCL incorporates elements from both CL and CDA approaches. The blue elements are expanded upon in this thesis.

## 2.4 Marrying Continual Learning and Concept Drift

Temporal continual learning combines the concepts of CL's task manager and CDA's change detection to identify previously learned states and new states. The KB learner of CL is generally associated with understanding distinct tasks, while CDA has a more simple and less resource-expensive memory concept. Additionally, TCL seeks to adopt the distinct memory concept of states from CL, while avoiding the limiting assumptions generally made by CL, to allow open-world learning of these states as they occur in a time-sequential manner, similar to CDA. Figure 2.6 shows the comparative form of TCL, borrowing from both CL and CDA architectures in three main ways:

1. State manager: CDA aims to identify ill-defined, changing states in sequential data but lacks a state manager to deal with a distinct state concept within the system. For CL, a distinct task concept is generally central to the approach. Moreover, TCL attempts to combine the beneficial elements of both CDA and CL by learning changing states in sequential data, similar to CDA, but combining this with a distinct state concept, which is used for memory addressing, similar to CL. This aims to allow TCL to build state-based memories over time in an

open-world manner.

2. State-based memory: CDA and CL have different memory needs. Generally, CL assumes tasks have interdependencies and can be discerned and stored as distinct task concepts in memory. In addition, CDA generally does not have a distinct concept for states but instead tends to assume elements of dependency, usually defined by instances. This allows CL to develop a more advanced concept of a task that can be more interpretable, whereas CDA can only develop a more simple concept that is generally less interpretable. Furthermore, TCL draws from CL in having distinct concepts held in memory and using these concepts to form a context for future input, while being similar to CDA in that these concepts are states rather than tasks.

3. The base learner is distinct from memory: Both CL and CDA approaches have been developed using a wide range of sequential learning approaches but the chosen approach tends to be integral to the system. In addition, TCL can be designed to use sequential learners that are architecturally distinct from a memory structure and the knowledge-based learner. This might allow the interchangeable use of well-understood base learners, which would add to the interpretability of outcomes and memory management.

# Chapter 3

# Literature Review: Memory Addressing

CL memory approaches using external memory structures require an appropriate memory addressing mechanism: a method of storing and recalling memory. This chapter combines selective areas of CL, ML, time series research, and CD to build two concepts that are useful for driving TCL recall gates and remember gates: time series similarity and change points.

Section 3.1 describes how similarity has been used in CL and CDA and introduces how it can be used to drive a TCL recall gate by comparing contextual cues stored with state-based memories with the current input space. Section 3.2 discusses residual change to drive a remember gate capable of open-world learning and of supporting state-based memory augmentation.

## 3.1 Time series Similarity

Time series similarity, particularly cross-sectional similarity, is a key concept for TCL. Identifying repeating patterns or distributions and judging similarity between subsequences, past and present, can form the basis for contextual memory addressing, specifically recall cues. This concept is covered in this section with a focus on the defacto benchmark approach for TSC: DTW [193].

### 3.1.1 Time series Classification

Time series similarity research has primarily been motivated by the need to classify time series data in the real-world. Moreover, TSC has been conducted from the *data*

*mining* thread of computer science research, which has remained fairly distinct from other ML approaches, perhaps owing to the real-world constraints imposed by the practical matter of the task: minimising classification errors using computationally inexpensive approaches that can sift through massive, noisy, real-world datasets. Hundreds of separate approaches for TSC have been developed over the past decade, divided into three categories [9]:

1. Whole series: Comparison of two series,
2. Subsequences: Select and compare partial sequences of a time series, and
3. Motif based: Identify and compare short patterns.

Additionally, the following variations are sometimes considered:

- Dictionary: Frequency of recurring patterns,
- Combinations: Combines shapelet and dictionary approaches, and
- Model based: Model fitting and comparison.

While time series similarity has predominantly been associated with pattern repeats in large databases, TCL systems should be more concerned with detecting and responding to changing tasks or states presented to the system. In a time series context, this means gauging whether the distribution of the input series has changed or, better still, whether a signature is available to identify a given task or state should it approximately reoccur in the future. The most obviously relevant family of data-mining approaches for this are whole series, but motif-based approaches may be useful, and these areas are expanded upon in this paper.

Two main considerations are required for TSC: representation and similarity. Representation refers to how the data are transformed before a distance calculation to determine similarity can be applied. Many transformations exist, such as the discrete Fourier transformation [60], discrete wavelet transform [28], symbolic approximation [117], and perceptually important point [68]. However, a greater focus has been on the choice of similarity metric, which is where this study is focused. Various measures have been used, including ED [59], DTW [193], hidden Markov models [162], ARMA [237], compression-based dissimilarity measure [118], spatial assembling distance [30], and others [4]. The different categories of TSC, exemplified by DTW, are discussed next.

### Whole Series and Subsequences

The research in *whole series* and *subsequence* TSC has shown that, despite the multitude of algorithms applied to the challenge, a simple nearest neighbour (1NN) classifier has proved exceptionally difficult to beat, with the best similarity measure being specific to the domain of application [17]. Although hundreds of alternative similarity measures have been developed over the last 10 years, numerous authors have reported that variants of DTW, most commonly 1NN-DTW, are the best measure in most domains [178]. In fact, DTW has become the defacto benchmark for TSC problems [9]. Moroever, DTW has the flexibility to compare to sequences of different lengths (although the benefit of this feature has been debated [184]).

However, DTW has a quadratic expense and is analytically intractable, prompting the development of many variants that are generally focused on speed-up measures [150, 252, 119]. The deeper question for TCL is whether a similarity measure, such as DTW, would be effective for memory addressing, and further, whether DTW is effective or whether a less resource-costly alternative would be as good.

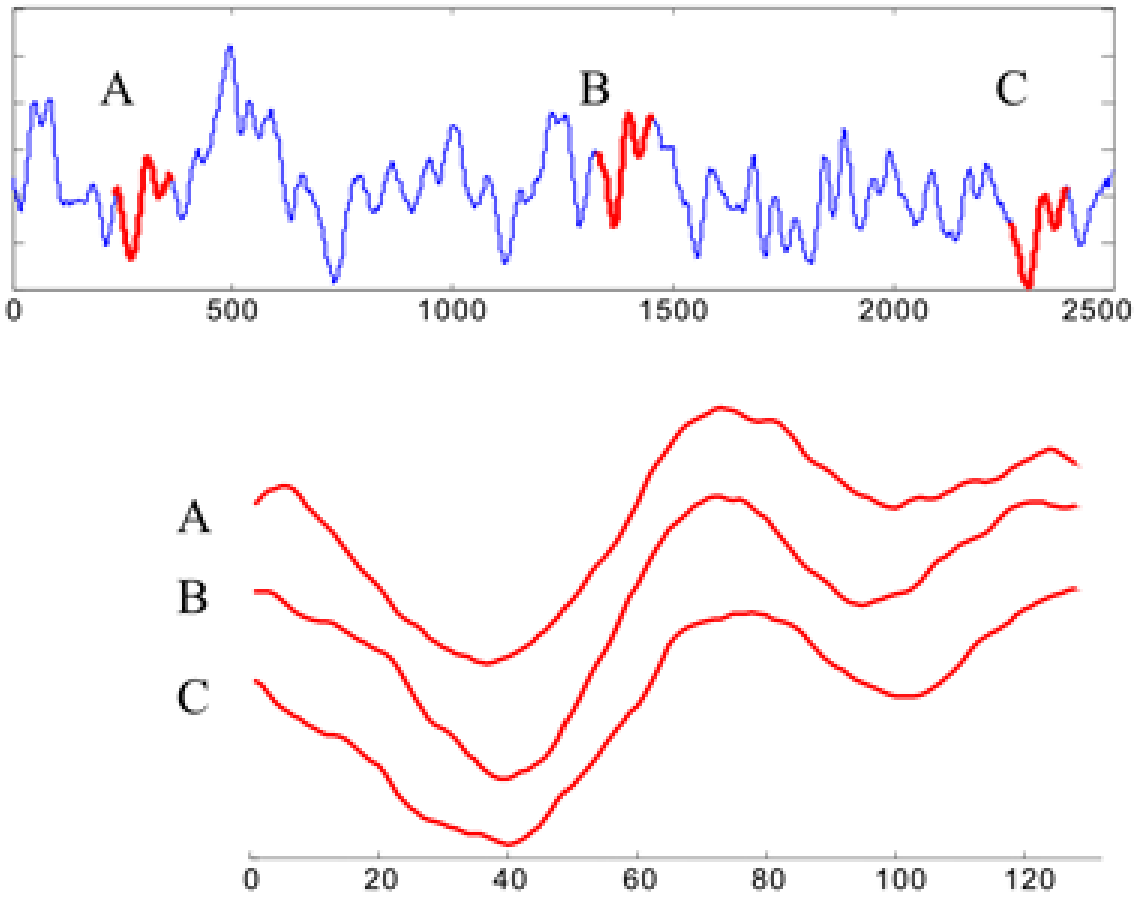
### Time series Motifs

Time series motifs [138] are pairs of individual time series or subsequences of longer time series that are similar to each other. (These have also been described as primitive shapes [45] or frequent patterns [98]). Again, DTW can be applied as an approach to find motifs. Motifs represent repeating patterns in noisy and long time series data. Figure 3.1 shows an example of a time series motif, where a repeating pattern is hidden in a seemingly noisy time series (seen at A, B, and C). In addition to using the changing distribution of input data to determine context, repeating patterns of this nature could also, in principle, be used to determine contextual memory cues in complex noisy data.

Lin et al. (2002) [138] noted the truism that the discovery of associated rules in time series first requires the discovery of motifs. It is posited here that those rules could be used to drive memory gates in a TCL framework. However, the challenges to achieving this are significant.

Time series motifs (or shapelets) are typically used in unsupervised data mining or for analytical functions. Considerable research has focused on time series motif discovery across a variety of domains. Perhaps the most challenging aspect of motif discovery is that motifs are generally of a variable and unknown length, and it has been suggested that exact algorithms to determine motif length [156] are likely to be intractable because of the high expense of computation involved [155]. This challenge can alternatively be described as attempting to find generalised distance

Figure 3.1: Motif example in astronomical data.




Note: Motif example in astronomical data: A, B, and C represent a very similar (as seen in the lower chart) repeating pattern or "motif".

measures that are invariant to noise. Identifying what is and what is not "noise" represents a key challenge. However, despite the practical challenges, intuitively, the idea behind motifs applies to the real-world, including finance. For example, it is generally accepted that stocks with growing earnings are likely to make good investments because stocks or currency pairs that have a strong price momentum tend to continue to appreciate. To make the point in a multivariate context, consider the consequences of the identification of the following hypothetical motif: a company with fast-growing earnings but with a falling price. It intuitively follows that motif discovery in financial data could be extremely interesting. However, the noisy nature of financial time series makes applying current techniques a huge challenge. One example within a finance dataset is of dividend payments, which can be seen in Figure 3.2. A repeating three-dimensional (3D) motif occurs (highlighted) in 2013, 2014, and 2015.

Statistics have also addressed time series similarity to some extent, mainly wrt the goodness of fit. This work generally attempts to establish statistical tests for comparing two probability distributions, the most notable being the non-parametric Kolmogorov-Smirnov two-sample test. An empirical distribution function is compared to an assumed distribution to determine whether it was drawn from the assumed distribution. However, where $K > 2$, this presents problems [110]. Nonetheless, the basic idea of the ordering of the cumulative distribution functions of two distributions and cross-comparing is applicable for similarity associated with cross-sectional data.

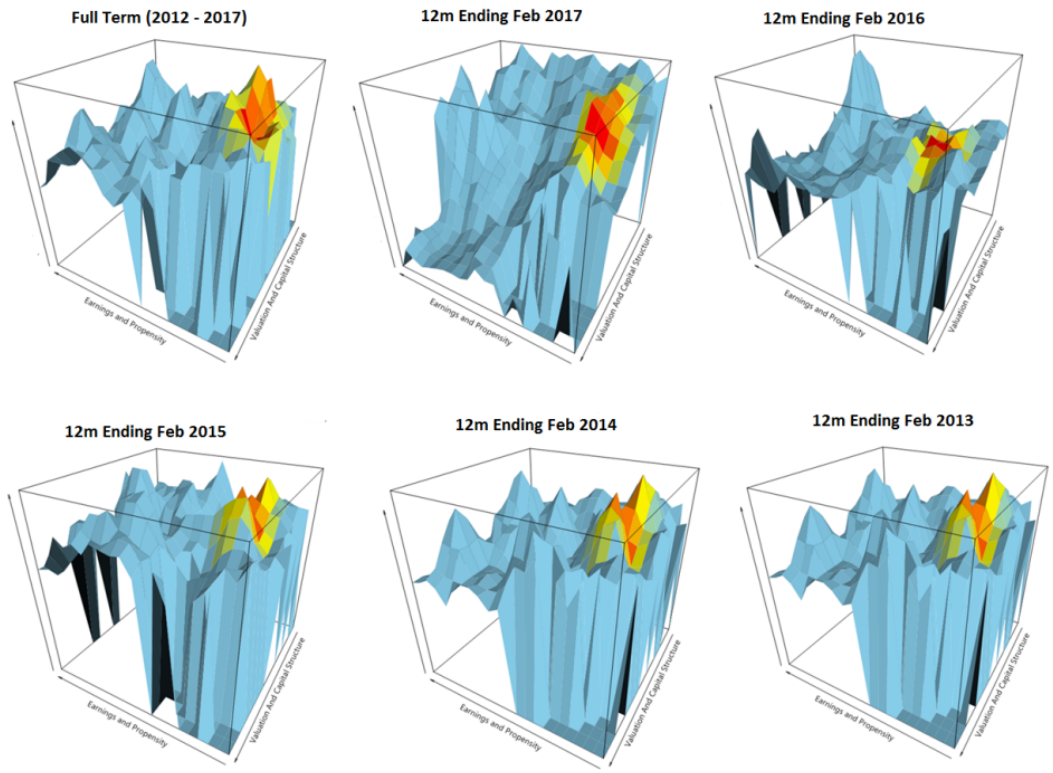
### 3.1.2 Dynamic Time Warping Distance

As DTW can identify phase-invariant repeating patterns, it might also be used as a measure of contextual similarity and therefore provide cues for a memory recall gate. Variations of DTW may also be appropriate in this application, such as time warp edit [150], using an elastic distance metric, weighted DTW [252], which adds a penalty based on the warping distance, and derivative DTW [119], which uses the (estimated) local derivatives of the data.

Common problems exist among DTW variants: the expense of multivariate time series [199] (although this has been contested [184]) and the need for direct comparison of raw instances within the dataset $\boldsymbol{X_t}$ to judge similarity with a subsequence from a different temporal context, $\boldsymbol{X_m}$. However, efficiency measures have been introduced, such as adding constraints to the warping path [193, 179]. A plain-vanilla DTW approach is described below, with details of the specific DTW customisation used described in Chapter 5).

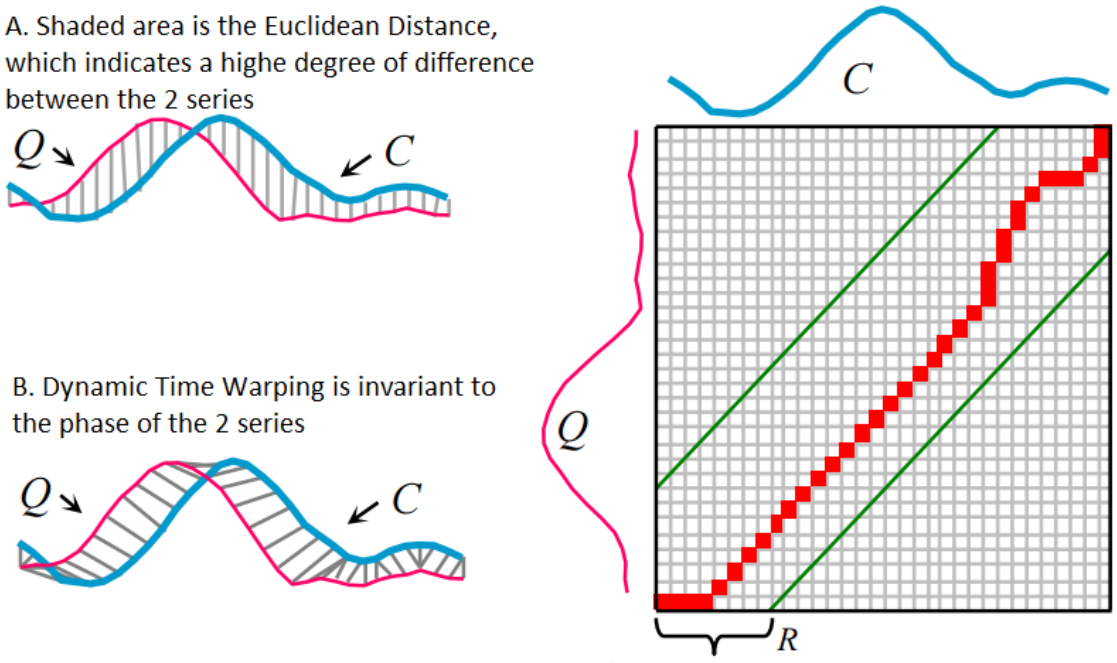
Mathematically, DTW requires two time series to compare:

Figure 3.2: Three-dimensional (3D) motifs.



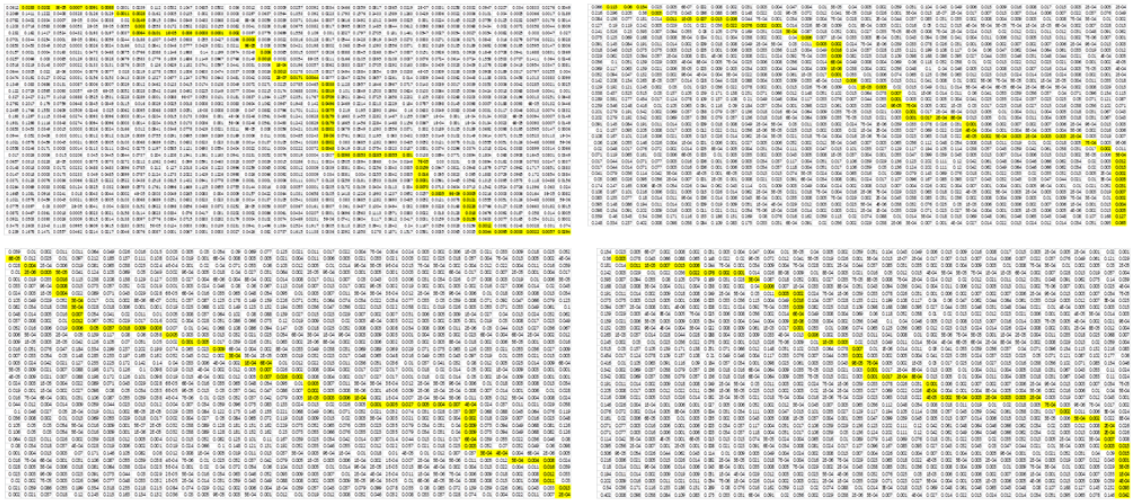

Note: Motifs can also be higher dimensional. This is an example of a three-dimensional (3D) motif in a finance time series relating to company accounting and stock valuation data. See the repeating 3D shape (red/yellow), in 2015, 2014, and 2013, which represents the level of dividend payments (when considered in terms of the hand-picked features: *earnings and propensity* and *valuation and capital structure*).

Figure 3.3: Dynamic time warping versus Euclidean distance.




Note: Motif example in astronomical data: A, B, and C represent a very similar (as seen in the lower chart) repeating pattern or "motif".

Figure 3.4: Dynamic time warping, multivariate warping paths of financial time series.



Note: An example of warping paths showing two instances (left and right), each with four associated time series. The diagonal yellow line represents the warping path between one instance time series and another.

$$Q(q_1, q_2, \ldots, q_n) and C(c_1, c_2, \ldots, c_n), \tag{3.1}$$

with the length of $n$ and $m$, respectively. The $n$ x $m$ similarity matrix, $\boldsymbol{M}$, is then defined to represent the phase-invariant "mapping" of $\boldsymbol{Q}$ onto $\boldsymbol{C}$, where the element $M_{DTW:i,j}$ indicates the distance $d(q_i, c_j)$ between the data points $q_i$ and $c_j$. The point-to-point mapping between $\boldsymbol{Q}$ and $\boldsymbol{C}$ is represented by $\boldsymbol{W}$, and the path of least resistance is through $M$, or the "time warping path". An example of the warping paths this creates is shown in Figure 3.4, where two instances (securities) are sampled at random from $\boldsymbol{X_m}$ and $\boldsymbol{X_t}$ before their four associated time series have warping paths generated and are both evaluated for DTW distance.

The warping path has the following characteristics: $W = \langle w_1, w_2, \ldots, w_k \rangle$, $max(m, n) \leq K \leq m + n - 1$, where the element $w_k = (i, j)$ indicates the probable alignment and matching relationship between $x_i$ and $y_j$. If a path is the lowest-cost path between two series, the corresponding DTW distance is required to meet the following:

$$DTW(\boldsymbol{Q}, \boldsymbol{C}) = \min_{w} \left\{ \sum_{k=1}^{K} d_k, W\langle w_1, w_2, \ldots, w_k \rangle \right\}, \tag{3.2}$$

where $d_(k) = d(q_i, c_j)$ represents contiguous elements in the matrix, $\boldsymbol{M_{DTW}}$, and each one is represented as $w_k = (i, j)$ on path $\boldsymbol{W}$.

While the warping path can express an extremely interesting mapping between two series, the final DTW distance measure of two time series can be calculated using dynamic programming on the warping path accumulated through $\boldsymbol{M}$. The following represents this, where the warping path leads to the current cell $r(i, j)$ from the minimum distance adjacent cells:

$$r(i, j) = d(x_i, y_j) + min\{r(i-1, j), r(i, j-1), r(i-1, j-1)\}, \tag{3.3}$$

where $r(i, j)$ is accumulated as the dynamic programming determines the warping path and where $i = I$ and $j = J$, it represents the time warping distance between series $Q_{1:i}$ and $C_{1:j}$. Series with high similarity can be effectively identified because the best alignment and matching relationship between two series are defined by the dynamic time distance. Many variants and tweaks have been proposed for DTW to exploit the phase-invariant distance while attempting to mitigate the resource cost of the approach.

### 3.1.3 Autoencoder Distance

Autoencoders have been used by CL researchers to gauge task similarity in the context of MTL [5] and for memory consolidation [220, 257]. (A separate application of

AEs in CL has been for generative replay [165, 206] in pseudo-task rehearsal, using *variational autoencoders* and generative adversarial approaches after [84]). In this context, AEs are useful to gauge the similarity of tasks using *reconstruction loss*, the distance between input data passed into an AE and the reconstruction returned. However, assessing the similarity between more stylised CL tasks is likely to be more straightforward than assessing the similarity between the noisy multivariate time series in a TCL approach.

Research into AEs, outside of CL, has introduced different AE variants, such as *masked AEs* for distribution estimation [75], *stacked AEs* [13], and others. However, research into SAEs, designed to benefit data representation, [182, 181, 51, 163] appears most relevant to TCL. By forcing a sparsity condition, it is possible to represent larger amounts of data using fewer latent variables, giving the benefits of compression, dimensional reduction, some invariance to noise, and easier classification [182]. The noise-invariance properties of AE sparsity are particularly interesting in a CL and TCL context, but AEs have only been narrowly investigated [5, 220]. As one of the few examples of AE use in a CL approach at the time of writing, Aljundi et al. [5] found that AEs with a degree of sparse representation using rectified linear units (ReLU) as activation functions in the encoder gave beneficial sparsity to AE representation for task identification and selection. Taking this idea further, it is possible to introduce additional sparsity conditions to AEs, such as sparsity-based regularisation (and traditional regularisation) [164]. The AEs that employ this are sometimes referred to as SAEs. Achieving more sparsity in AEs using sparse activation functions and the addition of sparse regularisation is highly likely to have a promising application in time series similarity.

## 3.2 Change Points

Open-world learning of different time series and cross-sectional states might also be described as a time series change-point detection problem. If states are to be defined and learned and their re-occurrence detected, detecting changes between states is necessary. This has been investigated in the CD literature to some extent but at a generally minimalist level; therefore, an overview of the different change approaches is conducted here to determine the most appropriate approach for the state-based memory addressing of TCLs. A wide range of change-point detection approaches have been proposed with many similar ideas being given different names in different fields of research. This has created a confusing array of overlapping nomenclatures and ideas, each with their own terminology and related concepts in different fields, for example:

1. Econometrics: Regime change,

2. Statistics: Change points, and

3. Adaptive learning/streaming: CD and co-variance shift.

All change approaches have significant drawbacks, and at least some of these drawbacks can be mitigated by taking a minimalist approach for the purpose required. Given the aim of TCL for open-world learning, state-based memory augmentation, applicability to multivariate time series, and the aim to generically support time series learners, *sequential change* using a simple *residual change* approach is highly appropriate.

Subsection 3.2.1 presents an overview of change approaches. Section 3.2.2 describes an example of states in a financial time series. Section 3.2.3 describes popular examples of change approaches. Section 3.2.4 explains in more detail the important concept of residual change, and Section 3.2.5 describes the use of change as a memory concept in CDA approaches while contrasting the needs of TCL.

### 3.2.1 Change Concepts

Many alternative approaches have been used for change-point detection, including non-parametric approaches [80], relative density [141, 115], and AL algorithms [70] extending to a number of ML approaches [100]. Several types of change have been identified, including the following:

- Sequential change,

- Statistical process change, and

- Change between distributions.

Change-point problems, in many cases, have tended to be framed as univariate. However, many real-world problems are multivariate, which further complicates assessing change. One solution to this has been to use the sequential change in the residual multivariate time series learner. This can turn a multivariate problem into a much simpler univariate problem: *residual change*. More recent problems, given the modern high generation of data, need to deal with change in multivariate time series data that have a large cross-sectional component. This type of dataset is relevant to the TCL problem, and *residual change* is proposed as the appropriate change-point approach. First, we describe the variation in research across econometrics, statistics, and AL (and CDA, more specifically).

**Change in Econometrics** In econometrics, the problem of change points is sometimes referred to as regime change. Regime switching models [121] and change-point detection [170] provide a simplified answer to identifying changing states in time series with the major disadvantage that change points between states (or regimes) are notoriously difficult to identify out of sample [58], and existing econometric approaches are limited by long-term, generally parametric assumptions in their attempts [56, 255, 208]. The study of change points in statistics is discussed next.

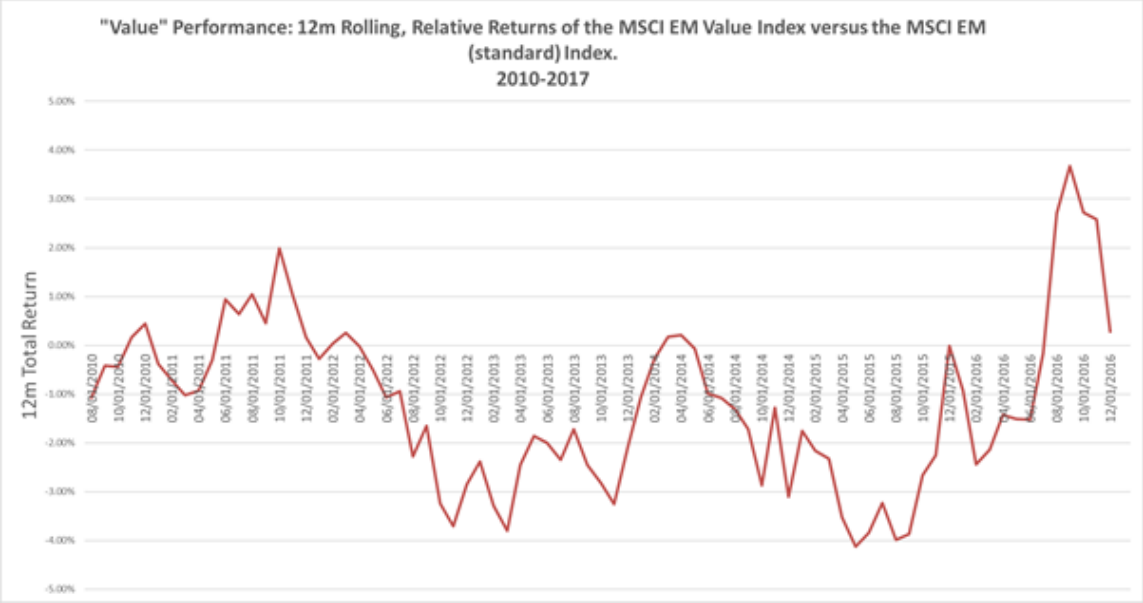
**Change in Statistics** Researchers have attempted to address the change-point problem using several approaches, many based on the sequential probability ratio test [173], cumulative sum (CUSUM) [54], and exponentially weighted moving average (EWMA) [187]. These approaches have traditionally been applied to input data but have also been used on the error of time series predictors [70], residual change, which is a much more interesting area of application for TCL, as we observe herein.

**Change in Adaptive Learning/Streaming** Change in sequential distribution has also been used in change-point detection, for example, using the Hellinger distance to detect changes in bias between training and test data distributions [37] and the drift detection method used in AL [109]. This idea has also been used for change detection in sequential batch learning [50]. Additionally, more complex change detection has also been proposed by monitoring multiple analytics, such as performance indicators, accuracy, recall, and precision [125]. In addition, the use of residual change [71] and change to drive memory cues, which is covered later, were addressed in the CD literature.

### 3.2.2 Example of States in Financial Time Series

Anecdotally, an example of a well-known regime shift that occurs in financial markets is that between value (i.e. tending to be in older industries, such as banks, telecommunications, and utilities) and growth stocks (such as technology). Figure 3.5 illustrates that stocks that are considered to be value have periods of out-performance and under-performance. Considerable debate exists regarding the driving causes and leading indicators of this, but an approach that could successfully learn these change points would make a successful investment strategy.

Figure 3.5: Changing States: Value versus Growth.




Note: An example of regime shifts is arguably the shift between value and growth stocks, shown here in emerging markets. Source: MSCI.

### 3.2.3 Traditional Change

Both developed in the 1950s, CUSUM and EWMA are still among the most popular change approaches [107, 8, 207, 214]. Both are parametric approaches, requiring a critical value to be specified that would represent a change. For example, CUSUM, as the name suggests, sequentially sums data points in a time series, whereas EWMA calculates the exponentially weighted average. In either case, when the derived value exceeds a determined critical level, this is assumed to represent change. Determining this critical value is the crux of these approaches (sometimes known as *online thresholding*). Moreover, CUSUM provides a threshold representing a decision rule for a sequential, generally univariate input series:

$$g_t = max(0, g_{t-1} + (x_t - \delta))(g_0 = 0), \tag{3.4}$$

where $x_t$ is the latest value in a time series, $\delta$ is the change that is permitted, and $j$ is a user-defined threshold to give the decision rule $g_t > j$. This means the CUSUM test rests on the selection of the values of $\delta$ and $j$. This can result in more Type I errors if these values are too low (and more Type II errors if the values are too high). Statistical tests have been developed for CUSUM to determine the values of these parameters, including the related Page-Hinkley (PH) [54], Shiryaev and Roberts method, and Shiryaev's Bayesian test.

Parametric implementations of approaches such as CUSUM and EWMA are likely to be disadvantageous in a world potentially exposed to non-stationarities. A non-parametric approach is likely to be advantageous. However, thresholding approaches are appealing for their simplicity and because threshold values might be sequentially learned over time in a TCL framework. An additional consideration is the need to support multivariate time series and allow for a generic application to time series learners, which is made possible using *residual change*, as discussed in the next section.

### 3.2.4 Residual Change

The choice of the series in which to judge change is a critical choice with many change approaches focusing on a change in the input data. However, no guarantee exists that a change point in a time series represents a significant change in the accuracy of an applied model, a far more useful perspective for learning different states in TCL. An alternative approach is to focus on *residual change*, the change in the absolute error of a learner, aiming to capture as much information as possible regarding changes in the relation between independent and dependent variables. The main advantage of

this approach is that it can be applied to any learner in principle. Different forms of residual change have been developed in the past [22, 106, 105, 146, 103]. However, many change-point detection approaches assume a single or known number of change points in a series and are less applicable to a priori change points or multivariate series [241]. As we observe, residual change in the CDA literature [71] has been used to drive decisions, such as forcing the training of a new model when a change was detected in an absolute error series. However, driving decisions from change points has the disadvantage that false alarms can be triggered if a change is inaccurately detected, for example, when an out-of-sample residual increases though over-fitting rather than from a change in state. However, if it was possible to sequentially learn a critical change threshold sequentially over time for a given learner, the risk of false alarms might be mitigated.

### 3.2.5 Change as a Memory Concept

The concept of residual change has also been extensively researched in the CD literature, where it is known as *concept drift* (CD). Discussed earlier in this thesis, CD has been used to drive simple memory cues in CDA. The idea is that different forms of drift should logically illicit different memory actions, which can be harnessed and expanded for TCL.

*Concept drift* is analogous to residual change in the CD literature, and relates to a change in the relationship between the independent and dependent variables of a given time series function. This is commonly used to infer whether an apparent change in state has occurred, perhaps where inferred through a change in the input distribution and/or through a change in the efficacy of the learner applied to it. For instance, an apparent change may result from a sampling effect or perhaps from model over-fit, which are both examples of $virtual\ drift$ [234]. However, a genuine change may occur in the underlying state of the input distribution, known as $real\ CD$ [194]. Both types of drift have been used to provide cues to CDA approaches. First, when *real* drift occurs, adaptation should follow [194]. Second, when *virtual* drift occurs, model generalisation might be improved, or the data could be resampled [234] [70] before training, which might be beneficial. Both cues have been used in CDA but can also be used in TCL to act as a cue for richer, more interpretable state-oriented memory concepts, rather than the minimalist and outcome-oriented responses that are typical in CDA. However, driving decisions from change points has the disadvantage that false alarms can be triggered, for example, in the case of virtual drift where an out-of-sample absolute error increases though over-fitting rather than from a change in state. This might be mitigated by sequentially learning a critical change

Table 3.1: Adaptive Learning Based on Concept Drift

| | | **State Changed?** Yes | **State Changed?** No |
|---|---|---|---|
| **Learner Absolute Error Increased?** | Yes | *Real concept drift* (Remember & Train) | *Virtual concept drift* (Forget & Tune) |
| | No | *Recurrent concept drift* (No Change) | *No concept drift* (No Change) |

This table represents changes in the true state of a temporal series (top) as interpreted using changes in absolute learner error (residual change, left). These concept drift cues can be translated into memory cues for the continual learning problem to form state-based memory cues for temporal continual learning.

threshold over longer periods for a given learner, which TCL, with its richer memory, should ideally be able to support. If false alarms can be reduced through sequential learning of thresholds, it may be possible to learn how the degree of change (*real* CD) influences state-based memory addressing to make state-based remembering more accurate and more interpretable. In additional to this, if the number of thresholds needed can be minimised, be learned over the very long term, and be non-parametric, this is clearly advantageous for TCL. Additionally, interpreting *recurrent* CD (Figure 3.1) might also be translated to the TCL problem, where attention and balancing between state-based memories might be introduced to improve state-based memory recall.

The concept of real drift, referred to as residual change in this study, as a memory cue is expanded on in Chapter 5.

# Chapter 4

# TCL Remember and Recall Gates

This chapter introduces TCL remember gate and recall gates. These provide the basis of the CLA framework, which is fully introduced in the next chapter. Section 4.2 discusses memory addressing concepts for recall, whereas Section 4.1 covers remembering. Section 4.1.1 sets out the empirical case for a memory-recall gate applied to noisy temporal datasets based on time series similarity. Section 4.2.1 presents the statistical case for a memory-remember gate based on residual change.

## 4.1 Memory Recall Cues

Time series similarity, particularly cross-sectional similarity, is a key concept for TCL. Identifying repeating patterns or distributions, or *motifs*, and judging similarity between subsequences, past versus present, can form the basis for contextual memory addressing, specifically recall cues. This section examines several distance measures, from which (dis)similarity is established in empirical testing. Tests are reported using synthetic datasets to examine time-sequential similarity combined with cross-sectional similarity.

### 4.1.1 Univariate Time series Similarity

Simple, univariate time series, time-sequential similarity is likely to be more important for datasets with a smaller cross-sectional component (perhaps $N = T$). This describes relatively simple time series problems where a less complex similarity approach might be more appropriate. However, in the real world, many samples might exist at each time point, adding a cross-sectional component to the problem that should be accounted for. For TCL, it would be of varying importance to examine time-sequential

and cross-sectional similarity, determined by the shape of the considered dataset. For domains with fewer samples at each time-point (perhaps $N = T$), time-sequential similarity might be more important. For domains where many more instances exist at each time point (i.e. $N >> T$), cross-sectional similarity might be more important. Real-world datasets are noisy, and understanding how both time-sequential and cross-sectional similarity are affected by noise is therefore important. To allow these perspectives to be tested, a stylised test dataset was created and used to assess how different similarity approaches perform on controlled perturbations (which has been noted to be very important [83]). In the next sections, testing is reported first on time-sequential similarity and, second, on both cross-sectional and time-sequential similarity.

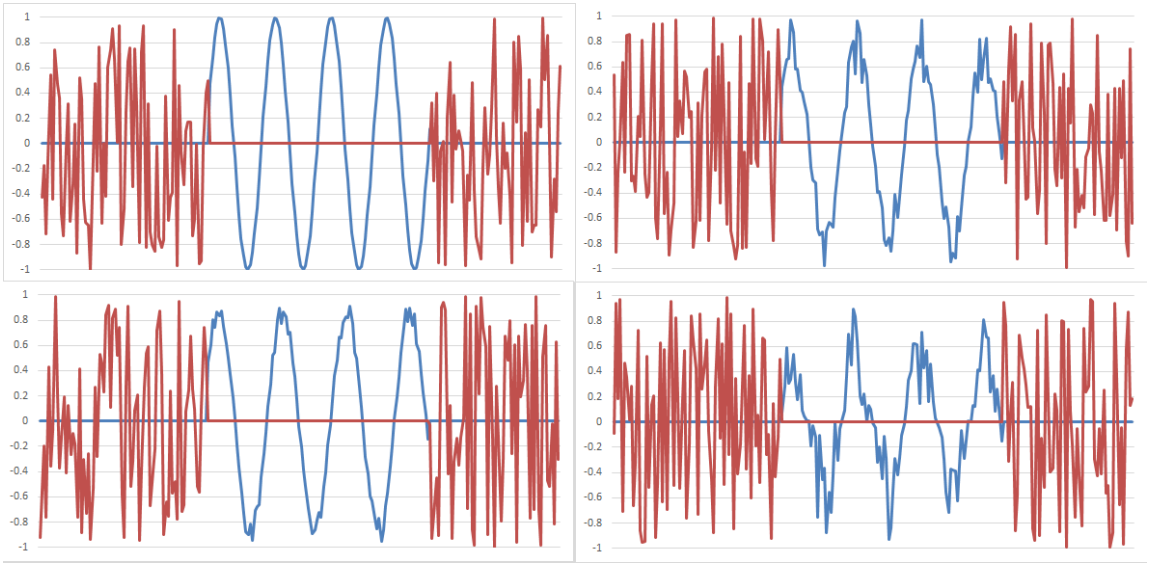
### Univariate Experimental Setup

Stylised tests were designed to establish the effectiveness of different univariate time series similarity approaches in dealing with noise while trying to identify time-sequential pattern repeats (*motifs*). This involved observing the successful identification of pattern repeats while avoiding spurious identification of patterns when no pattern was present.

A simple sine wave was taken as a *motif* to identify and was repeated 20 times at equal distances apart and was interspersed with normally distributed random values between -1 and 1. This formed a time series of 2,000 data points in which the motif was hidden. The 20 sine waves in the series were perturbed in a step-wise fashion over 10 steps of increasing perturbation. Firstly, each wave was perturbed by phase, stretching the sine waves by an increasing amount. Second, each wave was perturbed by noise by introducing an increasing amount of random distortion to the waves at each step. This was done for 10 steps in each dimension, creating a 10x10 grid of increasing perturbations. Cell $\{1, 1\}$ represented a test on the data with no perturbation. Cell $\{10, 1\}$ was heavily perturbed by phase with no perturbation by cross-section, while cell $\{1, 10\}$ had no perturbation by phase but was heavily perturbed by cross-section. Each cell was used to test similarity approaches.

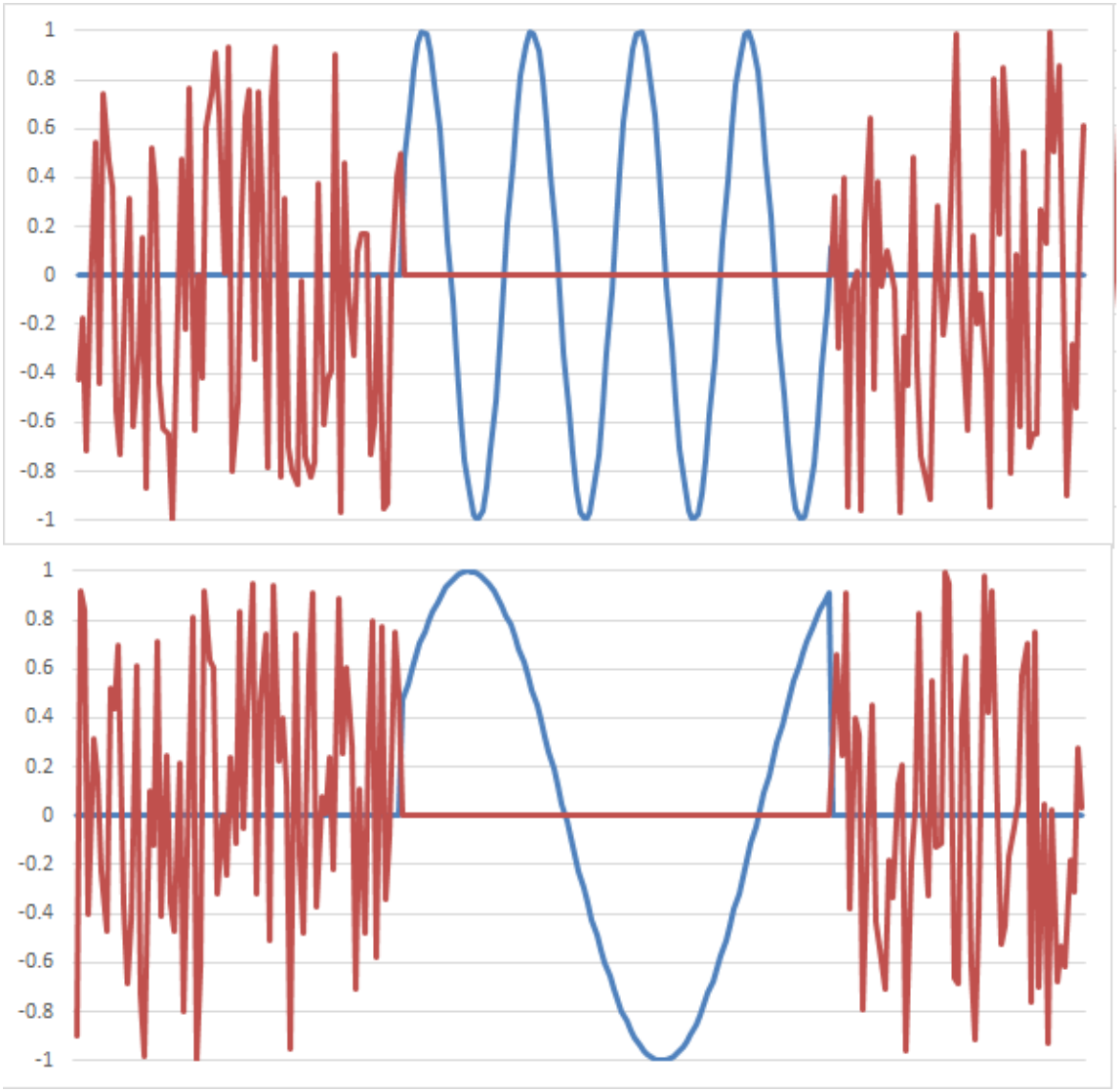
Each similarity approach was tested 100 times relating to each cell in the 10x10 perturbation grid. By taking the motif and calculating the distance between this and the 20 perturbed sine waves in the series and then calculating the distance between random numbers interspersing the perturbed sine waves in the series, two average distance calculations were found for each measure at each point in the matrix: the average distance between the motif and perturbed sine waves in the series, $\bar{D_{true}}$, and

Figure 4.1: Time-sequential Similarity Testing: Noise



Note: Perturbing by phase and noise. Time-sequential similarity tests, using a sine wave as a motif, examined how the effectiveness of the approach varied with phase. The blue line indicates the motif, and red indicates a random variable. Top left: Unperturbed sine wave (blue) interspersed with random variables (red). Bottom left: Sine wave mildly perturbed by noise (blue). Top right: Sine wave moderately perturbed by noise (blue). Bottom right: Sine wave heavily perturbed by noise (blue)

Figure 4.2: Univariate Similarity Testing: Phase




Note: Perturbing by phase. Top: Unperturbed sine wave. Bottom: Sine wave heavily perturbed by phase.

the average distance between the motif and interspersing random numbers, $\bar{D_{false}}$:

$$D = \|\bar{D_{true}} - \bar{D_{false}}\|, \tag{4.1}$$

where $D$ is a measure of the performance of the tested distance measure, $\sigma_D$ is the full series standard deviation of $D$, and $\bar{D}/(\sigma_D/\sqrt{n})$ is a test statistic. Statistical testing was conducted to test for the difference of means (using a $t$-test). Ideally, an effective distance measure should show a low difference between the motif and perturbed waves, whereas the distance between the motif and noise should be fairly consistently high. However, as the perturbations of the sine wave increased, the distance was expected to also increase, ideally in a strictly increasing fashion.

While many similarity approaches exist, ED was tested as a baseline with DTW as a common benchmark approach, while the AE distance was examined following the work by [221] but as a means of gauging distance between cross-sectional distributions.

### Univariate Distance Measures

Four distance measures were used to calculate univariate distance. First, ED was tested:

$$\hat{D_{ED}} = 1/N \sum_{i=0}^{N} ED(X_{m,i}, X_{t,i}), \tag{4.2}$$

where $\boldsymbol{X_m}$ and $\boldsymbol{X_t}$ are time series subsequences, $\hat{D}$ is the dissimilarity, and $N$ is the number of samples to take. Second, the DTW distance is as follows:

$$\hat{D_{DTW}} = 1/N \sum_{i=0}^{N} DTW(X_{m,i}, X_{t,i}). \tag{4.3}$$

Third, AE reconstruction was used to calculate distance, similar to [221], and ReLU were used in the encoder. The AE reconstruction loss function results in a form of AE, sometimes known as a SAE, as follows:

$$\mathcal{L} = 1/N \sum_{n=1}^{N} \sum_{k=1}^{K} (x_{k,n} - \hat{x_{k,n}})^2 + \lambda \Omega_{weights} + \beta \Omega_{Sparsity}, \tag{4.4}$$

where $\lambda$ is the coefficient for the L2 regularisation term, and $\beta$ is the coefficient for the sparsity regularisation term based on Kullback–Leibler divergence [164]. The AE distance calculation is as follows:

$$\hat{D_{AE}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} ED(X_t, a(h(X_t))), \tag{4.5}$$

where $ED(X_t, a(h(X_t)))$ is the reconstruction loss of the current input, $X_t$ is calculated as the ED, and $a$ and $h$ are the encoder and decoder functions, respectively.

Fourth, a DTW filtered AE distance, *warp-AE*, is introduced, which is intended to result in a phase-invariant AE reconstruction. This was expected to result in two beneficial effects: gaining the benefits of lower resource usage of AE compared to DTW while benefiting from the phase-invariant loss of DTW:

$$\hat{D_{wAE}}(\boldsymbol{X_m}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} DTW(X_t, a(h(X_t))). \tag{4.6}$$

By adding the DTW filter before the reconstruction loss is calculated, it may be possible to add a degree of phase invariance to the AE reconstruction loss. This is tested later in this thesis.

#### Univariate Results and Discussion

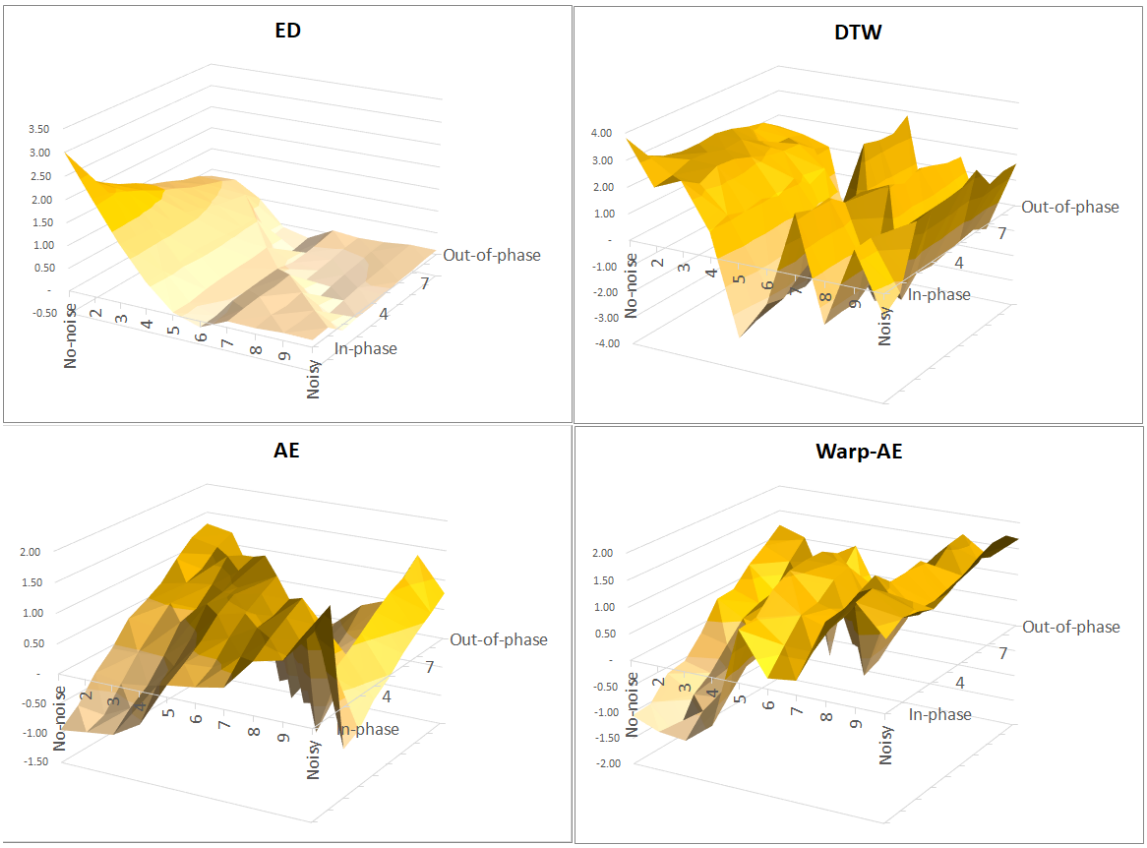
The ED tests showed a fairly linear deterioration in similarity with both phase and noise, as shown in Figure 4.3. This indicated a relatively stable relationship between perturbation and distance in the univariate tests. DTW showed a slower deterioration as the noise was introduced, indicating a greater invariance to these perturbations. This was stable up to the fourth noise step, but from the fifth, it became unstable, showing random values to be more similar to the target pattern (see the values that become negative at Step 5). The time dimension showed similar results with better performance than ED up until the fourth step but with instability from the fifth step onward. This indicated that DTW was relatively phase-invariant versus ED but only up to a point.

Both AE and warp-AE distance performed poorly on all univariate tests. The motif was, on average, not discerned from the noisy subsequences. As AE is used to reproduce a distribution, its poor performance in terms of discerning the univariate time-sequential pattern (as opposed to the cross-sectional distribution) was noted. It further implies that, for univariate pattern recognition, AE is a poor choice of similarity measure for driving TCL memory gating.

### 4.1.2 Cross-sectional Similarity

Next, the tests were designed and conducted to gauge the effectiveness of different distance approaches when applied to time series cross-sectional distributions rather than just univariate time series. This involved synthesising distributions with both time-sequential and cross-sectional perturbations for skew. Specifically, these were

Figure 4.3: Univariate Similarity Testing: Results.




Note: The chart shows the results from all cells in $\upsilon_{Uni}$. The y-axis is the distance between the perturbed motif and the true motif minus the distance been the random values and the true motif stated as an average. The higher the value, the greater the effectiveness of the distance measure. Note that ED and DTW are identical when matching the perfect pattern repeats. The x-axis shows 10 steps of noise perturbation, and the z-axis shows 10 steps of phase perturbation.

tests to determine the similarity between the distribution of instances cross-sectionally and time-sequentially. This is important because most real-world time series learning problems have a more significant cross-sectional dimension (i.e. many instances at each time point). For example, stock-related data in financial datasets, credit-scoring data, and datasets such as the UCI EEG database dataset, have at least 122 instances at each time point. In this case, where cross-sectional data are more plentiful than time-sequential data, it is likely to be more interesting to gauge a changing distribution across the cross-sectional distributions at each time step than to gauge sequential motifs. A stylised test dataset was generated to assess a number of time series distance measures.

### Cross-sectional Experimental Setup

For each distance measure, 100 different tests were conducted, this time on time series cross-sectional distributions that were perturbed using skew. The motif was a 500x500 matrix, $\boldsymbol{X}$, simulating a dataset with 500 time steps with each time step having a 500-sample cross-sectional distribution. Using a function $N$, a pseudo-random, normally distributed variable $y_t$ was sampled for each time step, and in each time step, this value was used as the mean from which to seed the 500 cross-sectional samples. This same procedure was conducted to form the perturbed data $\boldsymbol{X_m}$, except that, for the cross-sectional distribution at each time step, a pseudo-random, normally distributed variable was sampled with a certain amount of skew imposed, $\gamma_{Long}$, in each test:

$$\boldsymbol{X_m} = \{y_t = N(\mu, \sigma, \gamma_{Long}), \} \tag{4.7}$$

where the mean is $\mu = 1$, and the standard deviation is $\sigma = 0.2$. Cross-sectional samples were skewed by another amount, $\gamma_{Cross}$), as follows:

$$\boldsymbol{X_t} = \{x_{t,i} = N(y_i, \sigma, \gamma_{Cross}\}. \tag{4.8}$$

For each test, the parameters $\gamma_{Long}$ and $\gamma_{Cross}$ were increased with 10 steps of 0.1 each, in a range from 0 to 0.9, which resulted in a 10x10 grid, $\upsilon_{cs}$, of test results of different combinations of temporal and cross-sectional skew.

For each column in the grid, $\gamma_{Cross}$ was increased by 0.1, skewing the random variables drawn for the time-sequential distribution by that value on which testing would be conducted. For each row, $\gamma_{Cross}$ was increased by 0.1, skewing the random variables drawn for the cross-sectional distribution by that value. Element $\upsilon_{cs:1,1}$ therefore represented an unperturbed distribution, whereas element $\upsilon_{cs:1,10}$ represented a cross-sectionally unperturbed distribution (skew = 0) and a heavily skewed (i.e.

perturbed) temporal distribution (skew = 1). In addition, $\upsilon_{cs:10,10}$ represented a heavily perturbed cross-sectional (skew = 1) and temporal distribution (skew = 1). This created the example distributions for which the probability density functions (*pdf*s) are shown in Figure 4.4.

### Cross-sectional Distance Measures

As before, each similarity approach was tested, 100 times relating to each cell in $\upsilon_{cs}$. However, in these tests differences in distribution were estimated. The increases in distance between motif and test distribution that were expected with each step of perturbation were tested using the Mann-Kendall test [148, 116] to determine if statistically significant monotonicity from increasing distance existed. Additionally, the differences between motif and perturbed distributions were independently examined (using a Kolmogorov-Smirnov (KS) test), to determine if the chosen distance measures faced a trivial or non trivial task in calculating these distances.

Similarity approaches were adjusted to cope with the cross-sectional distribution. This involved using a sampling based approach for ED and DTW.

**Euclidean distance** Sampling-based ED was used to reduce processing time, only selecting a subset of $N$ randomly sampled instances from $\boldsymbol{X_m}$ and $\boldsymbol{X_t}$, sampling over rows, each of which represent different securities in the dataset, before comparing all time series associated with each pairing:

$$\hat{D_{ED}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} ED(\widetilde{X_{m,r_1(D)}}, X_{t,r_2(D)}), \tag{4.9}$$

where $\hat{D}$ is the dissimilarity, $N$ is the number of samples to take and $r_1(D), r_2(D)$ are random integers between 1 and $D$.

**Dynamic Time Warping** Efficiencies were also introduced to DTW to reduce some of the computational expense: applying traditional constraints to the warping path [193, 179] and using a sampling-based implementation. Sampling-based DTW was also used, only applied to a subset of $N$ randomly sampled instances from $\boldsymbol{X_m}$ and $\boldsymbol{X_t}$, sampling over rows, each of which represent different securities in the dataset, before comparing all time series associated with each pairing:

$$\hat{D}(\boldsymbol{X_m}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} DTW(X_{m,r_1(D)}, X_{t,r_2(D)}), \tag{4.10}$$

where $\hat{D}$ is the expected distance, $N$ is the number of samples, and $r_1(D), r_2(D)$ are random integers between 1 and $D$. Note that this is a multivariate variant of DTW; therefore, all related time series of every instance sampled in $\boldsymbol{X_m}$ are compared to those sampled from $\boldsymbol{X_t}$. The mean sampled distance is used to determine the final DTW similarity.

A DTW setup was used that applied traditional constraints to the warping path:

$$\hat{D_{DTW}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} DTW(\widetilde{X_{m,i}}, X_{t,i}), \tag{4.11}$$

where $\hat{D}$ is the expected distance and $N$ is the number of samples in the sliding window used.

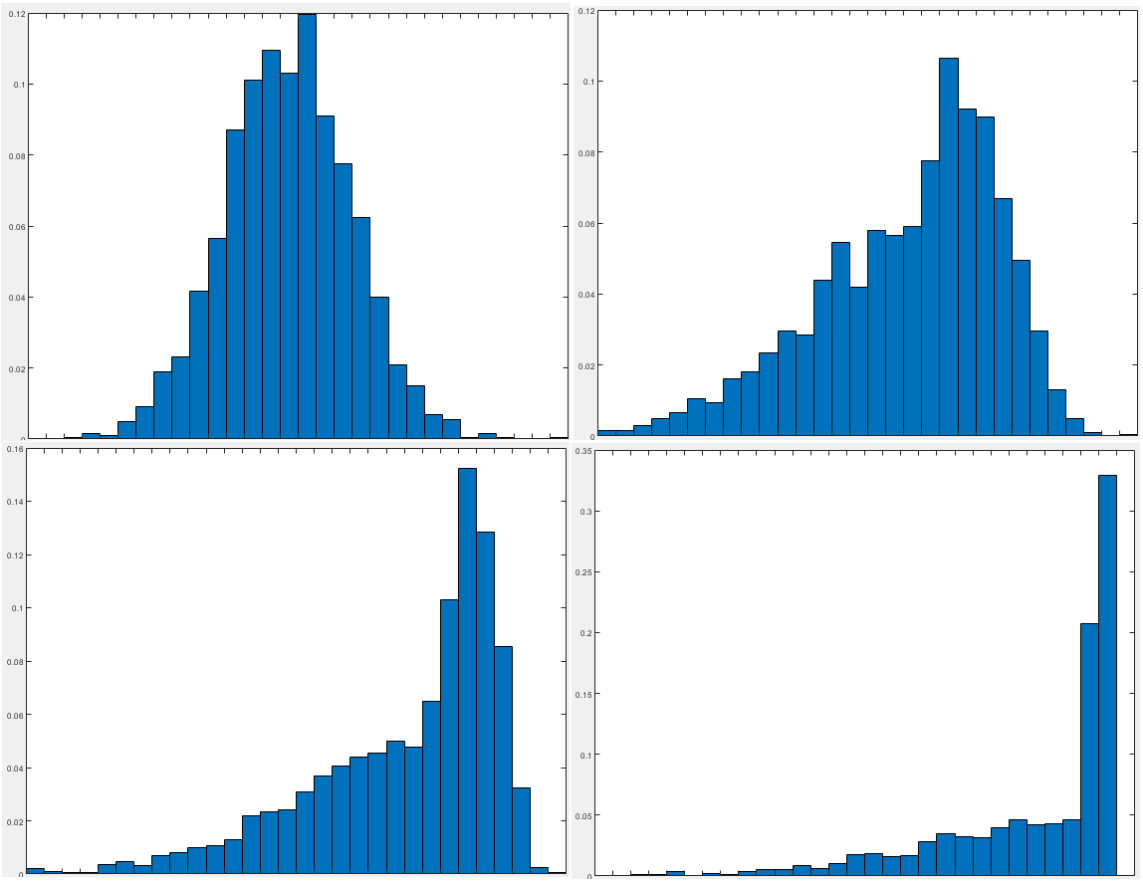
**AE Distance and warp-AE Distance** As previously described, the AE distances used were calculated in a generic manner, with the capability of reconstructing the cross-sectional distribution of the data.

### Testing for Similarity

It is possible to determine whether a distance measure has captured the increasing degrees of perturbation by cross-section and time sequence, individually and in combination. A highly effective distance measure would show a strictly increasing distance as perturbation increases. This can be approximated as a monotonicity, which was tested using the Mann-Kendall test. A null hypothesis $H_0 : p = 0$ indicated that a similarity measure did not show a statistically significant monotonicity, while the alternative hypothesis, $H_a : p \neq 0.0$ indicated that monotonicity could not be rejected. Rejection of the alternative hypothesis may indicate that a similarity approach is less appropriate for use in TCL memory gates applied to noisy time-sequential data with a cross-sectional component.

To sense check these distribution results, Kolmogorov-Smirnov tests were conducted on each test to determine whether the perturbations resulted in statistically significant differences between distributions. Should these tests indicate little difference between distributions, it would also indicate that measuring distances between different permutations should be fairly trivial. If the Kolmogorov-Smirnov test results indicated a statistically significant difference between distributions, it would indicate that measuring distances is more challenging. The Kolmogorov-Smirnov test showed statistically significant differences in almost all pairings, indicating the difficulty of interpreting time series similarity with conventional approaches and that measuring distances in this context is indeed non-trivial.

Figure 4.4: Distribution Distance Testing: Results



Note: These charts show examples of probability density functions ($pdf$s) of randomly generated, perturbed distributions used to test different similarity measures.

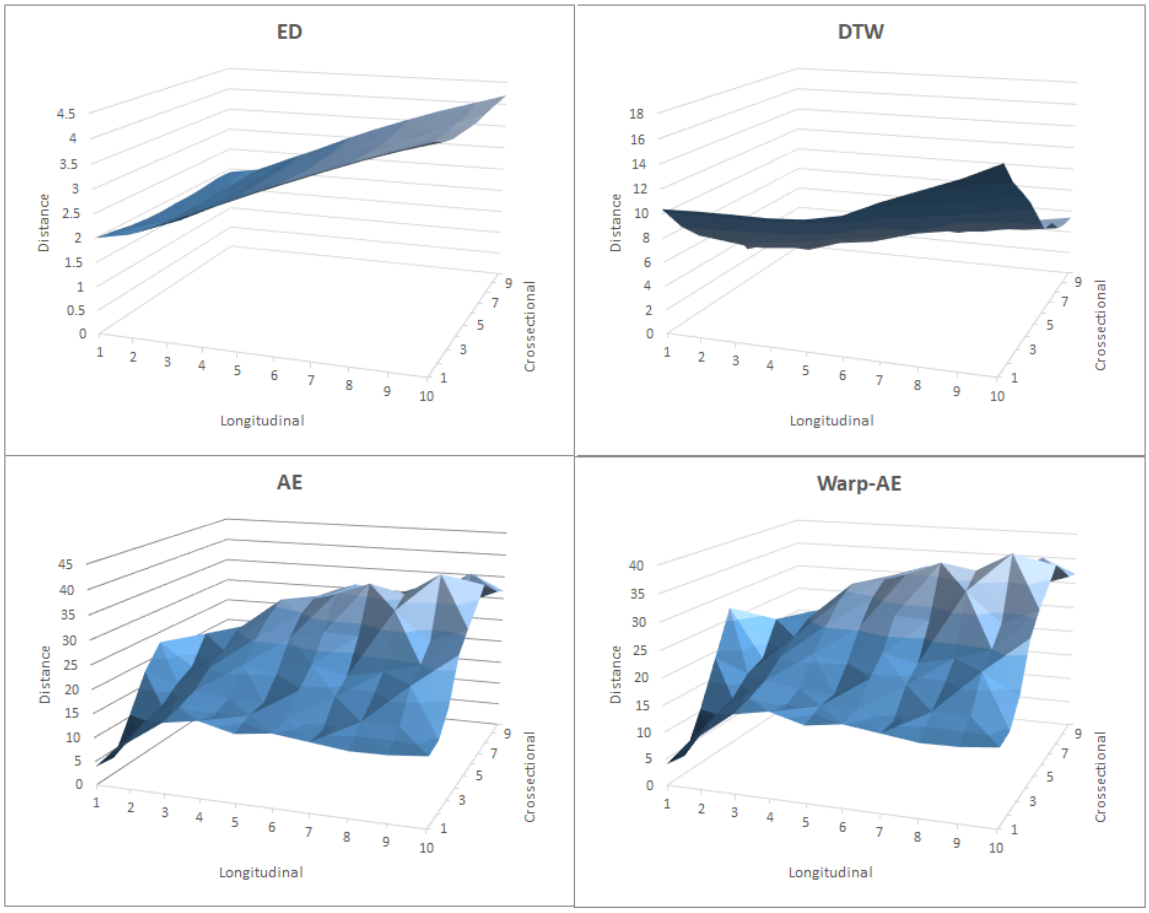
### Cross-sectional Results and Discussion

Cross-sectional testing produced greater insights than univariate testing, where combining a cross-sectional distribution with a time-series distribution was likely to be a more realistic case than perturbations of a simple univariate time-series formed of a sine wave. Performance of the distance measures was also starkly different when considering a cross-sectional element, with AE distance measures performing particularly well.

Mann-Kendall $p$-values were calculated for all cases where positive monotonicity was observed between Steps 1 and 5 and for the diagonal of $v_{cs}$ from 1 to 10. This identified which approaches resulted in increasing distances with the degree of perturbation (i.e. those that appeared to be effective in identifying the target distribution from the noise). In addition, AE and warp-AE showed the strongest monotonicity in these tests, with the ED showing as the least strong. The results are shown in Table 4.1. All approaches showed statistically significant monotonicity at the 5% level for cross-sectional perturbation in $v_{:,1}$, indicating that all might be used effectively in TCL. However, in time-sequential perturbations in $v_{1,:}$, only AE and warp-AE showed a monotonic increase in distance as the perturbation increased. The diagonal results were also tested and showed that only AE and warp-AE had statistically significant monotonicity at the 5% level. The results more broadly showed that distances plateaued cross-sectionally from Step 5.

These tests also showed that, in this setting, gauging distance time-sequentially is more challenging for the approaches tested than gauging distances cross-sectionally. First, given that many datasets in the real world come with a cross-sectional element, and given the degree of monotonicity observed in the distribution testing, this indicates that of all these approaches are likely to be suitable for application to TCL memory gates, but the tests indicate that AE-based approaches are likely to be the most effective.

Figure 4.5: Distribution Similarity Testing: Results




Note: These charts show how perturbing a random normal distribution time-sequentially and cross-sectionally influences similarity in a time series context.

Table 4.1: **Cross-sectional and Longitudinal Perturbation Tests**

***Diagonal Monotonicity Test* (p-value)***

| **ED** | **DTW** | **AE** | **wAE** |
|---|---|---|---|
| *na* | *na* | **0.0%** | **0.0%** |

***Cross-sectional: Step 1-5 Monotonicity Test* (p-value)***

| **ED** | **DTW** | **AE** | **wAE** |
|---|---|---|---|
| **0.0%** | **0.0%** | **0.0%** | **0.0%** |

***Longitudinal: Step 1-5 Monotonicity Test* (p-value)***

| **ED** | **DTW** | AE | **wAE** |
|---|---|---|---|
| *na* | *na* | **1.40%** | **3.60%** |

*Mann-Kendall $p$-value, *na*: wrong sign for monotonicity. Notes: The results of monotonicity testing to examine whether calculated distances increased as perturbation increased. The Mann-Kendall test $p$-values are shown. First, the diagonal of the grid $\boldsymbol{v_{i,i}}$ was tested, then the first row of $\boldsymbol{v_{1,:}}$ and first column of $\boldsymbol{v_{:,1}}$. *Na* indicates that the monotonicity had the wrong sign, implying an decrease in distance as the perturbation increased.

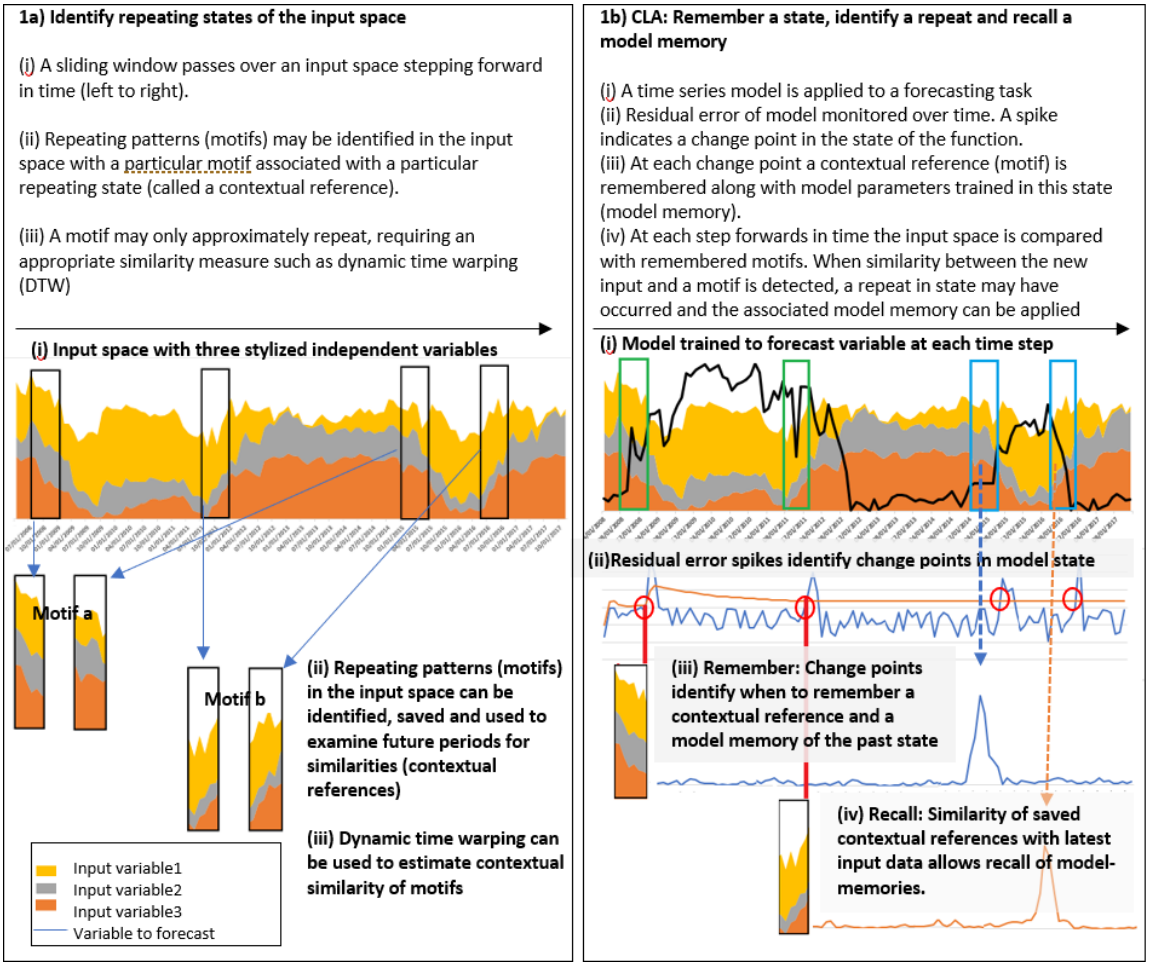
A simple method of describing how similarity and abrupt change can be used in memory gating for TCL is shown in Figure 4.6. In this example, Figure 4.6(1a) shows motifs (sequential patterns) providing a recall cue, where contextual similarity might prompt memory recalled from a memory structure. Figure 4.6(1b) shows how change points could cause the remembering of memories in a memory structure. This process of defining states using the change in a learner's residual is described in this chapter. Taking this simple framework, a description of how change might be applied to TCL remember gates follows.

## 4.2 Memory Remember Cues

### 4.2.1 Defining States Using Residual Change

One approach to addressing the stability-plasticity dilemma is using a FC architecture to learn this payoff, with the significant disadvantage of complexity. An alternative is

Figure 4.6: Temporal CL: Remember and Recall




Note: Simple sequential motifs driving TCL.

to use a thresholding approach, which can be defined by a single hyperparameter, $J_{Crit}$, denoting a confidence interval for change between temporal states or the marginal benefit of plasticity over stability. These ideas are related to CD [70] and have partially been investigated in the CDA literature [71], but these ideas are expanded here for application to TCL.

Assume we have a learner, $\phi_B$, parameterised by $\theta_B$, which can functionally approximate each time-varying state in a time series but suffers from CF (i.e. it cannot generalise over all states). We also assume that a state may reoccur at some future time. This describes the idea of CD [70], where the target distribution may change over time.

Change in the underlying function $\theta$ can be observed by identifying a change point $\delta_B$ in the out-of-sample absolute error process $\boldsymbol{\epsilon_B}$ of $\theta_B$, an approximation of the underlying function where the degree of change $D_{\boldsymbol{\epsilon}}$ can be defined as follows:

$$D_{\boldsymbol{\epsilon}}(\boldsymbol{\epsilon}_B||\boldsymbol{\epsilon}_{B,t}) = \int_{\chi} log\Bigg(d\boldsymbol{\epsilon}_B/d\boldsymbol{\epsilon}_{B,t}\Bigg). \tag{4.12}$$

What degree of change warrants remembering? Contrary to the principle of consolidation [123], with all else equal, memory should be expanded at a rate that achieves the highest marginal benefit given the available resources. This can be described in terms of cost and benefit. What level of imperfection of a learner in a given state warrants incurring the cost of remembering? This is a slightly different twist on the stability-plasticity dilemma, where plasticity can be thought of as a necessary cost for achieving CL, rather than as a benefit. If we now assume that the parameters $\boldsymbol{\theta}_{B,t}$ that are observed at the change point $\delta_B$ can be recalled when the respective states reoccur, the marginal cost, $MC$, per unit of $\delta_B$ for remembering these parameters can then be stated as follows:

$$MC = f(dc/d\delta_B), \tag{4.13}$$

where $c$ is the cost in resources for remembering $\theta_B$. If we assume memory recall and remembering are perfect and error free, this implies that the highest sensitivity to change would be better with all else equal:

$$\underset{\delta_B, c \leq C}{\operatorname{argmin}} f(dc/d\delta_B), \tag{4.14}$$

where $C$ is the total amount of resources available. However, in a real-world context, the change in $\boldsymbol{\epsilon}_B$ might not only reflect a change in the state but could also reflect changes in the sampling error or another false signal. If we assume the learner

generalises to the current state perfectly, the sampling error can be represented by the following:

$$D_{\boldsymbol{X}}(\boldsymbol{X}_{t-1} \| \boldsymbol{X}_t) = \sum_{x \epsilon \boldsymbol{X}} log\left(\boldsymbol{X}_{t-1} / \boldsymbol{X}_t\right), \tag{4.15}$$

where training data are $\boldsymbol{X}_{t-1}$ and the current input is $\boldsymbol{X}_t$, while $D_{\boldsymbol{X}}$ represents the sampling error in this discrete time step. This means that a state change point $J_{Crit}$ can be defined as $\delta_B$, which is adjusted for $D_{\boldsymbol{X}}$ as follows:

$$J_{Crit} = \delta_B - f(D_{\boldsymbol{X}}). \tag{4.16}$$

However, the problem is that inferring $J_{Crit}$ is generally intractable. Fortunately, regardless of the applied learner, a generic approach can be used to approximate $J_{Crit}$. As we observe, a memory management framework can be logically derived from these terms without needing to define or approximate a functional combination of these two processes.

### 4.2.2 Learning Residual Change

Theoretically, for a fair model of a state, $\boldsymbol{\epsilon_B}$ is approximately *i.i.d.* with a zero-valued mean. Therefore, the current model ceases to be a fair representation of the current state when $\boldsymbol{\epsilon_B}$ exceeds a certain confidence interval, in turn implying either a change in state, a sampling error, or a failure of the learner to generalise to the current state. In other words, a material deviation of the level of $\boldsymbol{\epsilon_B}$ is enough to indicate an important change in relation to dependent and independent variables that can be used in memory addressing. This confidence interval, $J_{Crit}$, represents a critical level for $\boldsymbol{\epsilon_B}$, indicating that a change point has occurred in the state:

$$\boldsymbol{M} = \left\{(\widetilde{\boldsymbol{X}_{\boldsymbol{m}}}, \theta_m)\right\} \leftarrow j(\epsilon_{B,t}, J_{Crit}) \begin{cases} 1 & \epsilon_{B,t} \geq J_{Crit} \\ 0 & \epsilon_{B,t} < J_{Crit}. \end{cases} \tag{4.17}$$

However, in the real world, an *i.i.d.* constraint to $\boldsymbol{\epsilon_B}$ is unrealistic; therefore, the assumption of a Gaussian distribution is likely to be unsafe. If we relax the *i.i.d.* constraint and consider alternative statistical interpretations of change in $\boldsymbol{\epsilon_B}$, this offers different options for defining and learning $J_{Crit}$:

- **Central limits and the law of large numbers:** Normality with the passage of time,
- **Alternative parametric:** Assumption of a distribution of $\boldsymbol{\epsilon_B}$ other than Gaussian.

•**Non-parametric:** Chebychev's inequality.

First, the passage of time is important for the interpretation of $\boldsymbol{\epsilon_B}$ (i.e. the number of samples, $\epsilon_t$, in the vector $\boldsymbol{\epsilon_B}$). Under the central limits, if we average $\sigma^2_{\delta_{t,B}}$ over time, we would expect a convergence to the state (i.e. sample) variance and, ultimately, the inter-state (i.e. population) variance over time. However, the law of large numbers indicates that the mean of the temporal distribution tends towards the population mean, which we might assume to be zero in the case of a learner residual observed over time.

This indicates that change is easier to discern over longer periods of time. This may present itself as a growing stability in a system driven by change points. As we observe in later sections, this occurs in practice, in relation to the memory remember gate proposed in this study (Figure 6.5). This has also been observed and used for change-point detection in CDA approaches [71] but not to drive memory augmentation for a CL problem.

Second, a simple parametric interpretation of $\boldsymbol{\epsilon_B}$ (e.g. a $t$-distribution) could be used. However, this would require anticipating the standard deviation of $\boldsymbol{\epsilon_B}$ as it stepped forward out of sample. An error in this assumption could cause too many or too few remember cues to occur.

Third, a non-parametric alternative is to define a confidence interval noting Chebychev's inequality, defined as $k_{\delta_B}$ standard deviations:

$$P(\boldsymbol{\delta}_B > j_{Crit}) \leq \frac{\sigma^2_{\boldsymbol{\delta}_B}}{k^2_{\delta_B}}. \tag{4.18}$$

Given a simple statistical interpretation of $\boldsymbol{\epsilon_B}$ based on Chebychev's inequality, this approach could be used in either early or late periods in the run time of a CL approach. In turn, this might allow critical levels to be determined in earlier and later periods. Using this principle, it is possible to learn $J_{Crit}$ as a non-parametric threshold.

In addition to considering $J_{Crit}$ to be a critical level, it can also be thought of as a hyperparameter of a CL system, which is possible to learn (or tune) over time. This learning process should aim to find an inflection point for $j_{Crit}$ that produces greater empirical net rewards from any final modelling outcome. Therefore, $J_{Crit}$ can be optimised at every time step to result in a level of sensitivity for remembering that forms an external memory, $\boldsymbol{M}$, resulting in the lowest empirical forecasting error for the CLA approach over the study term until time $T$:

$$J_{Crit} = \underset{j_{Crit} \epsilon jgrid}{\operatorname{argmin}} f(\boldsymbol{X_t}, j_{Crit}), \tag{4.19}$$

where $f$ is a CL approach expressed as a function of the input series and $j_{Crit}$, yielding $\epsilon_{B,t}$ (the absolute error of the base learner at time $t$). Moreover, $jgrid$ is a multi-point, equidistant set between the minimum and maximum values of $\boldsymbol{\epsilon_B}$, essentially a discretisation of the observed distribution of $\boldsymbol{\epsilon_B}$.

### 4.2.3 Stability-Plasticity Threshold: Complexity

Generally, CL requires a higher level of complexity when compared to other ML approaches. Statistical learning theory notes that complexity comes with a cost [88]. If an approach proves too complex based on the number of training examples and distribution of the test data, the approach will over-fit with ramifications for the generalisation out of sample. In ML (and deep learning, particularly), this is an open discussion [15, 83], but it seems wise to adhere to the principle of parsimony in the design of CL approaches.

A major advantage of using a hyperparameter-driven remember gate and not driving remembering using an FC approach is the natural reduction in parameters and model complexity. We can informally consider this question in the context of the Vapnik-Chervonenkis dimension (VC) [225], noting the denominator $e_{VC}$ of the famous VC dimension formula: and the commonly used rule of thumb, *the rule of 10*, describing the minimum ratio of training cases to parameters.

If $e_{VC}$ remains low, VC dimension suggests that fewer training instances can be used to result in the same confidence in a learner. However, in the real world, this cannot be assumed. While an approach could employ a more complex learner in order to generalise, this would reduce optimism in the approach. The alternative is to use a more simple learner and to respond to changes in residual by training the learner. An extension to this case is to use residual change to form a memory structure to allow long-term generalisation with a lower number of parameters and higher optimism.

As an example, DNC [87] has approximately 891,000 parameters [66]. In many real-world datasets, there are simply not enough data points to adhere to the *the rule of 10*. A working example can be considered to illustrate this point and to explain how the proposed TCL memory gates may have a significant complexity benefit.

In a financial context, around 3,000 investable equity securities exist in emerging markets, most with a five-year history, which is approximately 15,000 training instances. A commonly available commercial data-vendor, S&P CapitalIQ, supplies

as standard 2,385 separate data items with a weighted average annual frequency of 38 observations per annum. This results in around 90,630 observations per security, per annum, over a period of five to ten years.

In this case, model complexity might be up to 1,500 parameters. This is significantly less than DNC's parameterisation and the parameterisation of many DNNs, but is sufficient to parameterise a simpler base learner and tune simple hyperparameters.

# Chapter 5

# Continual Learning Augmentation

This chapter introduces CLA, a TCL framework. Section 5.1 relates the concepts of remember gates and recall gates together into the CLA framework using instance-based similarity approaches (ED and DTW) applied to sliding window base learners. Section 5.2 extends CLA to AE similarity approaches applied to sliding window and sequential-base learners.

## 5.1 CLA: Sliding Window Learners and Instance-based Similarity

In this section, a TCL framework is developed based on the remember and recall concepts discussed earlier in this thesis: CLA. The CLA memory augments a conventional learner for time series regression. The aim is to allow well-understood learners to be used in a CL framework in an explainable way. The memory functions of CLA are applied as a sliding window, stepping forward through time over the input data of one or more time series. The approach is initialised with an empty memory structure $\boldsymbol{M}$ after a base learner has been chosen to augment, $\phi$, parameterised by $\theta_B$. This base learner can be a sequential approach or a sliding window approach and can be applied to a multivariate input series $\boldsymbol{X}$, with $K$ variables over $T$ time steps. The chosen base learner produces a forecast value $\hat{y_{t+1}}$ in each period over time. A *remember* gate, $j$, appends a new memory $\boldsymbol{M_t^m}$ to $\boldsymbol{M}$ on a *remember cue* defined by the change in the base learner's absolute error at the time point $t$. A *recall* gate $g$ balances a mixture of base learner and memory forecasts to result in the final outcome of $\hat{y_{t+1}}$. Figure 5.1 shows the functional steps of remembering and recalling learner memories.

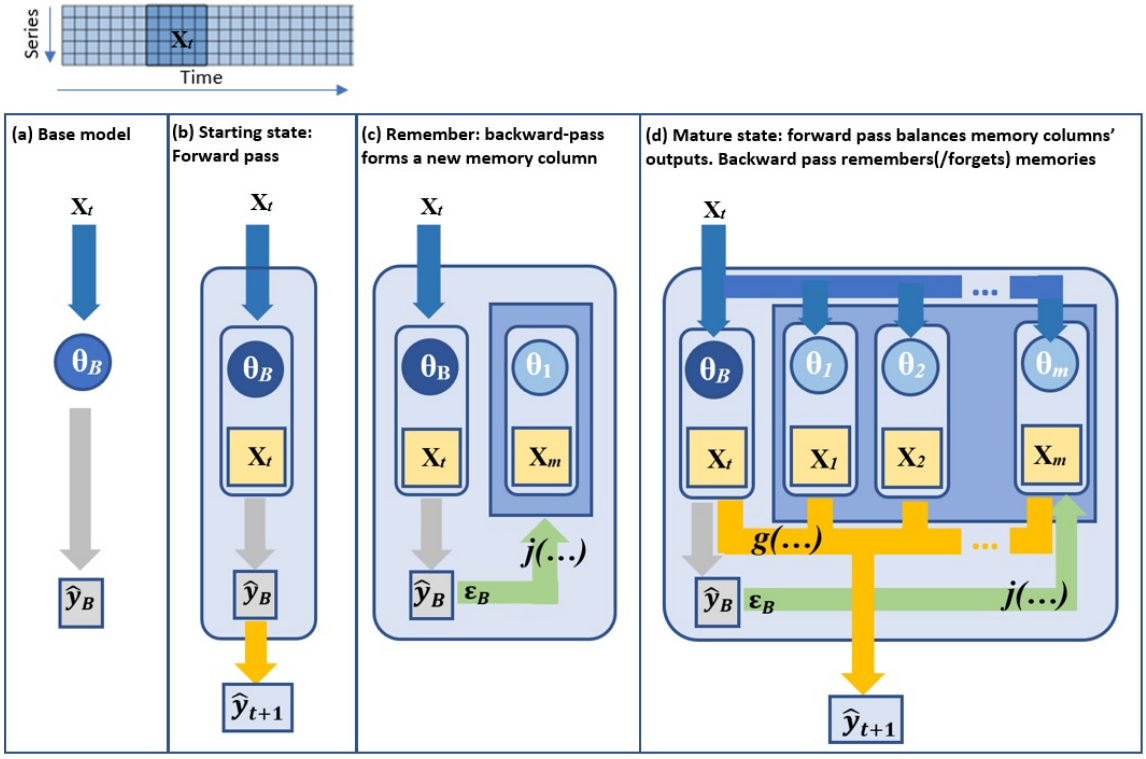



Figure 5.1: Columnar Architecture

Note: (a) A base learner, $\phi_B$, parameterised by $\theta_B$, is run, stepping forward through time, training at each time step. (b) This base learner is structured as a column containing parameters, $\theta_B$, and a contextual reference, $\boldsymbol{X_t}$. Initially the base model is run as normal, completing a forward pass with the input data $\boldsymbol{X_t}$ to forecast $(\hat{y_{B,t+1}})$. (c) As time steps on, $y_{t+1}$ becomes observable and a backward pass is conducted where the absolute error of the base learner, $\boldsymbol{\epsilon_B}$, determines if a change point has occurred, On a change, the remember gate $j$ copies the base learner column to a new memory column in $\boldsymbol{M}$. The base learner is then trained. (d) Over time more change points will be detected and more memory columns will be added. At each time step, all memory columns are run using the current input data $\boldsymbol{X_t}$. The forecast results of all the columns (including the base learner column) are balanced by the recall gate $g$ which uses the similarity of the current input, $X_{t-1}$ with the contextual reference of each memory, $\boldsymbol{X_m}$, to weight the output of each column, $\hat{y_B}$ and $\hat{y}_{m,t+1}$ of $\boldsymbol{M}$, to result in $\hat{y}_{(t+1)}$).

### 5.1.1 Memory Management

Repeating patterns are required in the input data to provide memory cues to remember and recall different past states. Model parameters are trained in a given past state, $\theta_m$, which can then be applied if that state approximately reoccurs in the future. When CLA forms a memory, it is stored as a column in an explicit memory structure, similar to [38], which changes in size over time as new memories are remembered and old ones are forgotten. Each memory column consists of a copy of a past base model parameterisation, $\theta_m$, and the training data $\boldsymbol{X_m}$ used to learn those parameters: $(\boldsymbol{X_m}, \theta_m)$. As the sliding window steps into a new period, CLA recalls one or more model memories by comparing the latest input data ($\boldsymbol{X_t}$) with the training data stored in each memory column ($\boldsymbol{X_m}$). Memories with training data that are more similar to the current input series have a higher weight applied to their output ($\hat{y}_{m,t+1}$) and therefore make a greater contribution to the final CLA output ($\hat{y}_{t+1}$).

### 5.1.2 Remember Gate

Remembering is triggered by changes in the absolute error series of the base learner, $\boldsymbol{\epsilon_B}$, as the approach steps forward through time. These changes are assumed to be associated with changes in the state, which are indicated by the function $j$, which defines a change and stores a pairing of the parameterisation of the base model $\theta_B$ and the contextual reference $\boldsymbol{X_t}$. Figure 5.1(c) shows how a change is detected by function $j$ from a backward pass, which then results in a new memory column being appended to $\boldsymbol{M}$:

$$\boldsymbol{M} = \big\{(\boldsymbol{X_1}, \theta_1), \ldots, (\boldsymbol{X_m}, \theta_m)\big\}. \tag{5.1}$$

Immediately after the remember event has occurred, a new base model is trained on the current input, overwriting $\theta_B$.

Theoretically, for a fair model of a state, $\boldsymbol{\epsilon_B}$ is approximately *i.i.d.* with a zero-valued mean. Therefore, the current base mode ceases to be a fair representation of the current state when $\boldsymbol{\epsilon_B}$ exceeds a certain confidence interval, implying a change in state. This is interpreted as a non-parametric threshold for the reasons described in Chapter 4. As discussed in Chapter 2, this $J_{Crit}$ represents a critical level for $\boldsymbol{\epsilon_B}$, indicating a change point has occurred in the state. Memories are only stored when the observed absolute error series, $\boldsymbol{\epsilon_B}$, spikes above a critical level, $J_{Crit}$:

The term $J_{Crit}$ is a hyperparameter, optimised at every time step, to result in a level of sensitivity to remembering that forms an external memory, $\boldsymbol{M}$, resulting in the lowest empirical forecasting error for the CLA approach over the study term until time $T$:

**Algorithm 1** Remember Gate $j$

**Require:** Initialise memory structure $\boldsymbol{M}$
**Require:** Initialise $J_{Crit}$
**Require:** Train base learner $\theta_{B,t=0}$
  # Step through time, period by period
  **for** all time steps $t$=1 in $T$ **do**
    # Base learner is run...
    $\hat{y}_t \leftarrow \phi(\boldsymbol{X_{t-1}}, \theta_{B,t-1})$
    # $y_t$ becomes observable
    #
    # ..... CLA back-pass starts .....
    $\epsilon_{B,t} = L(\hat{y}_t, y_t)$
    **if** $|\epsilon_{B,t}| \geq J_{Crit}$ **then**
      $\boldsymbol{X_m} \leftarrow \boldsymbol{X_{t-1}}$ store raw training instances
      append learner memory $(\boldsymbol{X_m}, \theta_{B,t-1})$ to $\boldsymbol{M}$
    **end if**
    # CLA Learns $J_{Crit}$ sensitivity
    $J_{Crit} \leftarrow$ learn and update $J_{Crit}$
    # ..... CLA backpass ends .....
    #
    $\theta_{B,t} \leftarrow (\boldsymbol{X_t}, \theta_{B,t})$ overwrite base learner
  **end for**

$$J_{Crit} = \underset{j_{Crit} \epsilon jgrid, Mc<C}{\operatorname{argmin}} f(\boldsymbol{X_t}, j_{Crit}), \tag{5.2}$$

where $f$ is the CLA approach expressed as a function of the input series and $j_{Crit}$, yielding $\epsilon_{B,t}$ (the absolute error of the base model at time $t$). In addition, $jgrid$ is a 20-point equidistant set between the minimum and maximum values of $\boldsymbol{\epsilon_B}$. $M$ is the number of memories in the memory structure, $c$ is the marginal cost of a memory and $C$ is the total cost in resources available.(Please see 4.2.2 for a discussion of marginal cost in relation to remembering). This function, in effect, balances the stability-plasticity of the CLA framework to the optimum empirical level at each time point.

### 5.1.3 Recall Gate

The recall of memories takes place in function $g$, which calculates $\hat{y}_{m,(t+1)}$, a mixture of the predictions from the current base model and model memories:

$$\hat{y}_{(t+1)} = g(\boldsymbol{X_t}, \boldsymbol{M^t}). \tag{5.3}$$

The mixture coefficients are based on comparing the similarity of the current time-varying context $\boldsymbol{X_t}$ with the contextual references $\boldsymbol{X_m}$ stored with each individual memory. Memories that are more similar to the current context have a greater weight in the final modelling outcome of CLA. To calculate contextual similarity, DTW is used. However, multivariate DTW is computationally expensive [199]. In addition to applying traditional constraints to the warping path, a sampling-based implementation reduces the expense further. (This is explained in detail earlier in this thesis). The mean sampled distance is used to determine the similarity between the current context and that of each memory.

In addition, DTW is only applied to a subset of $N$ randomly sampled instances from $\boldsymbol{X_m}$ and $\boldsymbol{X_t}$, sampling over rows, each of which represents different securities in the dataset:

$$\hat{D}(\boldsymbol{X_m}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} DTW(X_{m,r_1(D)}, X_{t,r_2(D)}), \tag{5.4}$$

where $\hat{D}$ is the expected distance, $N$ is the number of samples, and $r_1(D), r_2(D)$ are random integers between 1 and $D$.

### 5.1.4 Balancing

Two different approaches to memory balancing were used: first, the *best individual* (i.e. lowest distance) model memory:

$$\hat{y}_{t+1} = g_{Best}(\boldsymbol{X_t}, \boldsymbol{M_t^m}), \tag{5.5}$$

where $g_{Best}$ is an output function to select the model memory that is most similar to the current context:

$$\underset{m}{\operatorname{argmin}} \hat{D}(\boldsymbol{X_m}, \boldsymbol{X_t}), \tag{5.6}$$

where $\hat{y}_{t+1}$ is the regression output. Second, a *similarity-weighted* ensemble of all model memories, $g_{SimWeight}(\boldsymbol{X_t}, \boldsymbol{M})$, is as follows:

$$\hat{y}_{t+1} = \sum_{m=1}^{M} \phi(\boldsymbol{X_t}, \theta_m) \cdot \left[1 - \frac{\hat{D}(\boldsymbol{X_m}, \boldsymbol{X_t})}{\sum_{m=1}^{M} \hat{D}(\boldsymbol{X_m}, \boldsymbol{X_t})}\right], \tag{5.7}$$

where $M$ is the number of memories in the memory structure $\boldsymbol{M}$. As a past state is unlikely to perfectly repeat, a continuous function for balancing model memories is more likely to generalise better [87] than choosing the best single model (which is indeed found to be the case).

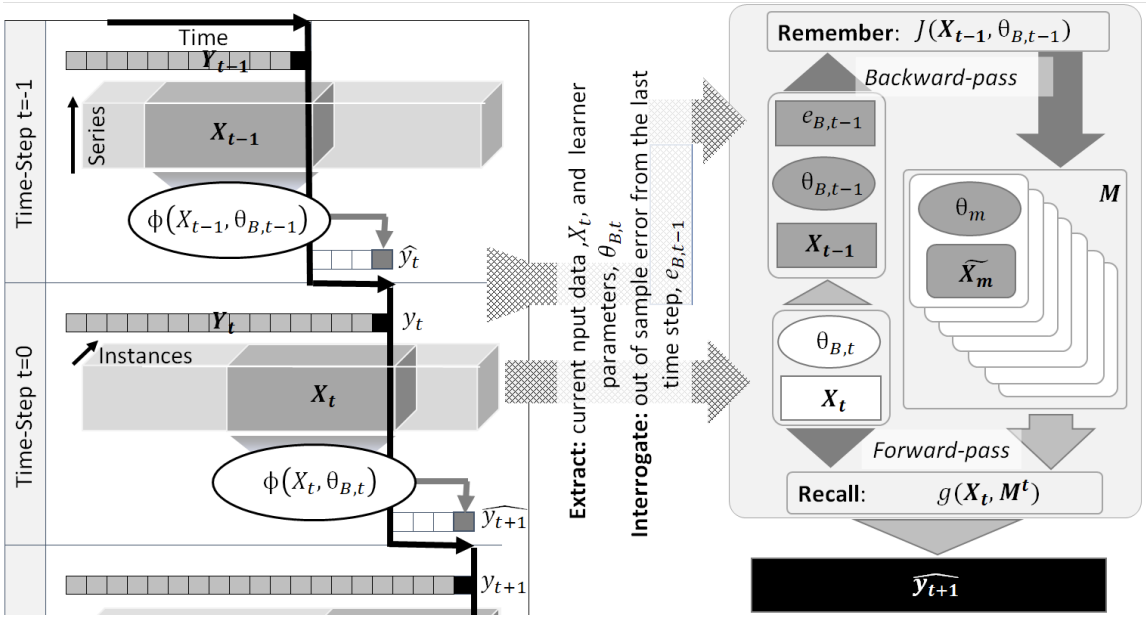
## 5.2 CLA: Recurrent Learners and Autoencoder Similarity

A number of refinements were made to CLA to allow recursive and sliding window learners to be augmented using the architecture. A schema of the overall approach is shown in Figure 5.2. The specific refinements made to CLA are explained next. Figure 5.2 shows the functional steps of remembering and recalling learner memories.

### 5.2.1 Memory Management Based on AEs

As the sliding window steps into a new period, CLA recalls one or more learner memories by comparing the latest input data ($\boldsymbol{X_t}$) with a representation of the training data stored in each memory column ($\widetilde{\boldsymbol{X_m}}$). Memories with training data that are more similar to the current input series have a higher weight applied to their output ($\hat{y}_{m,t+1}$) and therefore make a greater contribution to the final CLA output ($\hat{y}_{t+1}$).

Figure 5.2: Continual Learning Augmentation Architecture




Note: Continual learning augmentation architecture. Backward pass: extracts $\theta_{B,t-1}$, $\theta_{B,t-1}$ from the base learner, and $y_t$ becomes observable. *Remember gate*, $j$, assesses $|\boldsymbol{\epsilon_{B,t-1}}|$ for change. Memory column is added to $\boldsymbol{M}$. Forward pass: outputs of learner memories and base learner are balanced by the *recall gate*, $g$.

### 5.2.2 Remembering

Figure 5.2 shows how a change is detected by $j$, which then results in a new memory column being appended to $\boldsymbol{M}$:

$$\boldsymbol{M} = \left\{(\widetilde{\boldsymbol{X}_1}, \theta_1), \ldots, (\widetilde{\boldsymbol{X}_M}, \theta_M)\right\}. \tag{5.8}$$

**Algorithm 2** Remember gate $j$

**Require:** Initialise memory structure $\boldsymbol{M}$
**Require:** Initialise $J_{Crit}$
**Require:** Train base learner $\theta_{B,t=0}$
  # Step through time, period by period
  **for** all time-steps $t$=1 in $T$ **do**
    # Base learner is run...
    $\hat{y}_t \leftarrow \phi(\boldsymbol{X_{t-1}}, \theta_{B,t-1})$
    # $y_t$ becomes observable
    #
    # ..... CLA backpass starts .....
    $\epsilon_{B,t} = L(\hat{y}_t, y_t)$
    **if** $|\epsilon_{B,t}| \geq J_{Crit}$ **then**
      $\widetilde{\boldsymbol{X_m}} \leftarrow \boldsymbol{X_{t-1}}$ representation of training data
      append learner memory $(\widetilde{\boldsymbol{X_m}}, \theta_{B,t-1})$ to $\boldsymbol{M}$
    **end if**
    # CLA Learns $J_{Crit}$ sensitivity
    $J_{Crit} \leftarrow$ learn and update $J_{Crit}$
    # ..... CLA backpass ends .....
    #
    $\theta_{B,t} \leftarrow (\boldsymbol{X_t}, \theta_{B,t})$ overwrite base learner
  **end for**

### 5.2.3 Autoencoder-based Recall Gate

The recall of memories takes place in the recall gate $g$, which calculates $\hat{y}_{m,(t+1)}$, a mixture of the predictions from the current base learner and from learner memories:

$$\hat{y}_{(t+1)} = g(\boldsymbol{X_t}, \boldsymbol{M^t}). \tag{5.9}$$

The mixture coefficients are derived by comparing the similarity of the current time varying context $\boldsymbol{X_t}$ with the contextual references $\widetilde{\boldsymbol{X_m}}$ stored with each individual

memory. Memories that are more similar to the current context have a greater weight in the final outcome of CLA.

### 5.2.4 Recall: Testing Measures of Similarity

Several approaches for calculating contextual similarity are tested separately, using the CLA approach. Each is used to define $\widetilde{\boldsymbol{X_m}}$, either by simply storing past training examples or by using a process of contextual learning, essentially learning a representation of base-learner training data.

ED and DTW are applied first. Both approaches require $\widetilde{\boldsymbol{X_m}}$ to be raw training examples, which are required to be stored in each respective memory column, making both approaches relatively resource-hungry. Second, AE distance is used through a process of contextual learning. Rather than needing to store many training examples in a memory column, only the AE parameters are needed to form a reconstruction of the training data with the disadvantage that an AE must be trained in every time step. Third, a DTW filtered AE distance, introduced earlier in this thesis, is used: warp-AE. Again, an AE needs to be trained at every time step, but DTW processing expense is reduced because it is only run on AE reconstructions. We describe each approach in turn.

As described earlier in this thesis, ED and DTW are applied only to a subset of $N$ randomly sampled instances from $\widetilde{\boldsymbol{X_m}}$ and $\boldsymbol{X_t}$, sampling over rows, each of which represents different securities in the dataset:

$$\hat{D_{ED}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} ED(\widetilde{X_{m,r_1(D)}}, X_{t,r_2(D)}) \tag{5.10}$$

$$\hat{D_{DTW}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} DTW(\widetilde{X_{m,r_1(D)}}, X_{t,r_2(D)}), \tag{5.11}$$

where $\hat{D}$ is the dissimilarity, $N$ is the number of samples, and $r_1(D), r_2(D)$ are random integers between 1 and $D$.

The AE distance is used in a similar fashion to Aljundi et al. [5], using ReLU to avoid over-fitting. However, the use of AEs in CLA is different. Moreover, AEs are used for contextual learning for memory management to cope with noisy, real-world, multivariate time series. The use of ReLU aims to allow generalisation over the noise of otherwise similar time series subsequences. Additionally, the similarities returned from the AE implementation of CLA are also used to balance memory weighting:

$$\hat{D_{AE}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} ED(X_t, a(h(X_t))), \tag{5.12}$$

where $ED(X_t, a(h(X_t)))$ is the reconstruction loss of the current input, $X_t$ is calculated as the ED, and $a$ and $h$ are the encoder and decoder functions, respectively. warp-AE is designed to benefit from the lower memory usage of AEs compared to DTW while benefiting from the phase-invariant loss of DTW:

$$\hat{D_{wAE}}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t}) = 1/N \sum_{i=0}^{N} DTW(X_t, a(h(X_t))). \tag{5.13}$$

These (dis)similarities are used to determine memories to recall from $\boldsymbol{M}$ and to determine how to weight the contribution of each memory for the final outcome of CLA, $\hat{y}_{(t+1)}$. These different similarity functions were each tested in the memory recall gate for CLA, in turn, gaining new insight concerning the effectiveness of each similarity approach in the CL system, when applied to a complex multivariate time series problem.

### 5.2.5 Balancing

The base learner and all recalled memories are weighted by similarity to produce the final outcome of CLA, using the recall gate $g(\boldsymbol{X_t}, \boldsymbol{M})$:

$$\hat{y}_{t+1} = \sum_{m=1}^{M} \phi(\boldsymbol{X_t}, \theta_m) \cdot \left[1 - \frac{\hat{D}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t})}{\sum_{m=1}^{M} \hat{D}(\widetilde{\boldsymbol{X_m}}, \boldsymbol{X_t})}\right], \tag{5.14}$$

where $M$ is the number of memories in the memory structure $\boldsymbol{M}$. The previous research indicated this was the most powerful approach over selecting the best single memory [172]. (Notably, both balancing approaches significantly outperform equal weighing of all memories, indicating that CLA is gaining significantly more than a simple ensemble effect).

# Chapter 6

# Experimentation

This chapter introduces a testing framework for TCL and then reports experimentation using CLA applied to a real-world, temporal dataset with a large cross-sectional component: international and emerging equity stock selection simulations. Section 6.1 describes the development tools used in experimentation. Section 6.2 proposes a testing framework to benchmark the performance of TCL approaches. Section 6.3 introduces and reports experiments on an international equity dataset, using CLA configured with instance-based similarity approaches and sliding-window base learners. The experimental setup, results, and discussion are detailed. Section 6.4 introduces and reports on the experimentation on an emerging market equity dataset and uses CLA configured with AE similarity and recurrent-base learners. Again, the experimental setup, results, and discussion are detailed.

## 6.1 Tools and Software

MATLAB, Microsoft .NET, Excel, and SQL Server databases were used to drive experimentation. MATLAB's Deep Learning Toolbox was used for implementations of FFNNs, LSTMs, and AEs, whereas the functions for CLA were all developed from first principles and were encapsulated in MATLAB classes. Microsoft .NET was used to process data before parsing to MATLAB for analysis. Extensive programming was required in .NET and MATLAB to this end, including a trading simulation engine. Significant effort was expended for sanity checking results, including unit-testing and diagnostics in the code to avoid classic errors, such as data snooping. In addition, SQL Server was extensively used with several extraction scripts written for this purpose. Microsoft Excel was also used to shape and perform minor processing of the data, whereas MATLAB was also used for the statistical analysis of the results.

## 6.2 Testing for Augmentation Benefits

In this section, a framework for analysing the empirical results of TCL is discussed. This allows for the development and testing of gating that is applied in TCL when augmenting generic base learners in later experimentation. First, the simulated investment performance of a strategy is often used as a benchmark for examining investment models. A statistical framework has been developed in the finance literature to allow robust testing of such performance, which is discussed in this section. Second, a simple framework for examining the augmentation benefit of an approach is introduced.

### 6.2.1 Investment Returns as a Benchmark for TCL

The return of a strategy being non-zero can be tested using a simple ratio of risk to return, the Sharpe ratio [201]. A higher Sharpe ratio indicates higher returns per unit of risk (i.e. to a statistically significant level); therefore, a higher Sharpe ratio indicates a superior strategy. The Sharpe ratio can be tested for statistical significance, with the null hypothesis $H_0 : p = 0$ or the strategy return is not statistically significantly different from a return of zero, whereas the alternative hypothesis is $H_a : p \neq 0.0$. This uses a non-centred student $t$-test, with N−1 degrees of freedom [201]. Second, we can test how a TCL approach might augment a base learner by comparing the performance of the augmented learner with the unaugmented learner using an *information ratio* (IR), which we tested. These approaches are well known in the finance literature. Much debate has ensued over controlling statistical testing for various violations that financial return data present [143]. We review each measure in turn and how they are used to benchmark performance in experimental testing in this thesis.

#### Statistical Testing: Outright Benefit

To test whether the total outright performance of a strategy is statistically significant, a Sharpe ratio is calculated. This can be translated into a $t$-statistic and tested with the null hypothesis $H_0 : p = 0$, or the outright benefit is not statistically significantly different from a return of zero, whereas the alternative hypothesis is $H_a : p \neq 0.0$. First, the strategy returns and standard derivations are needed. The annualised total return (TR) is as follows:

$$T\bar{R}_p = \prod_{i=1}^{n} (1 + r_{p,i})^{T_a/T_{period}} - 1, \tag{6.1}$$

where $T_a$ is the number of years over which returns were generated, $r_{p,i}$. The annualised excess return is the following:

$$\bar{R}_p = \prod_{i=1}^{n} (1 + r_{p,i} - r_{f,i})^{T_a / T_{period}}, \tag{6.2}$$

where $r_{f,i}$ is the risk-free rate. The standard deviation of the returns is as follows:

$$\sigma_p = \sqrt{\sum_{i=1}^{n} (r_{p,i} - \bar{r})}, \tag{6.3}$$

where $T_{period}$ is the number of observations per annum. The Sharpe ratio, $SR$, is as follows:

$$SR_p = \log_{10}(\bar{R}_p) / \sigma_p. \tag{6.4}$$

In addition, the $t$-statistic is commonly expressed as follows [201]:

$$t_{SR} = SR_p \sqrt{T_{period}}, \tag{6.5}$$

A $t$-test can be conducted where the null hypothesis is $H_0 : p = 0.0$ (i.e. the excess return is not statistically significantly different from zero), whereas the alternative hypothesis is $H_a : p \neq 0.0$. The $p$-values can be stated for each $t$-statistic given.

### Statistical Testing of Continual Learning Augmentation Benefit

It also useful to test the performance of an augmented learner when compared to an unaugmented base learner to understand more about the benefits or drawbacks of augmentation. To do this, we can take the relative return (RR) of the simulated performance of an augmented learner relative to the simulated performance of an unaugmented base learner:

$$R\bar{R}_p = \prod_{i=1}^{n} (1 + r_{p,i}) / (1 + r_{b,i}) - 1, \tag{6.6}$$

where $r_{p,i}$ is the return of an augmented learners and $r_{b,i}$ is the simulated return of an unaugmented base learner. The standard deviation of the relative return $RR_p$ is known as the *tracking error* (in this case, it is not annualised):

$$TE_p = \sqrt{\sum_{i=1}^{n} ((1 + r_{p,i}) / (1 + r_{b,i}))}. \tag{6.7}$$

Similar to the Sharpe ratio, this can be expressed as a ratio, an IR, from which a $t$-statistic can be derived and on which a further $t$-test can be conducted, where the null hypothesis is $H_0 : p = 0.0$ (i.e. the augmentation benefit is not statistically significantly different from zero), whereas the alternative hypothesis is $H_a : p \neq 0.0$:

$$IR_p = \log_{10}(R\bar{R}_p)/TE_p, \tag{6.8}$$

The $t$-statistic can be expressed as follows:

$$t_{IR} = IR_p/(TE_p\sqrt{T_{period}}). \tag{6.9}$$

### Consistency of Augmentation

As well as testing the overall average augmentation benefit in terms of simulated returns, we can also test the consistency of any augmentation benefit using the *hit rate* of the relative returns, $R\bar{R}_p$. This is stated as the percentage of time points, $t$, in $T$, where an approach delivers a positive benefit, $R\bar{R}_p > 0$: $pctPos$. This provides an understanding of the consistency of the benefit. For instance, if the returns were beneficial from an approach tested over 10 years, but all those returns came in just a few periods, it might indicate an unstable result.

The sign test [40] that is used to statistically test the hit rate is a non-parametric test of limited statistical power and tends to be used in conjunction with other tests and observations. The null hypothesis $H_0 : p = 0.50$ implies that the strategy does not produce a hit rate that is different from a 50/50 split between positive and negative returning periods, while the alternative hypothesis is $H_a : p \neq 0.50$.

### Cross-sectional Augmentation

It is also common practice in financial analysis to examine the cross-sectional performance of a model, where instances in the cross-section tend to be different securities. This can be approached using a trading signal (or factor) calculated for each security and by sorting all securities by this signal at each time point independently across the term of the study. The returns of different quantiles can be calculated and assessed at a given frequency of rebalancing. This allows the analysis of a signal where the highest quantile in the sort order is expected to generate the highest return, and the lowest quantile is expected to result in the lowest return. Deciles are typically used. (See [58] for an exhaustive description of these techniques.) It is possible to examine these deciles for monotonicity, where a strictly increasing pattern is ideally expected from decile 1 (lowest forecast return) to decile 10 (highest forecast return), for example. This in turn can be statistically tested in two ways.

First, the Mann-Kendall, non-parametric test for monotonicity can be used. The Mann-Kendall test is commonly employed to detect monotonic trends in environmental time series. The null hypothesis $H_0$ is that the data come from a population with independent realisations and are identically distributed. The alternative hypothesis $H_a$ is that the data follow a monotonic trend. Second, a line of best fit is plotted across the return deciles, and this is assessed using an $F$-test where $H_0 : \beta = 0$ (i.e. no difference exists between the slope and intercept) $H_a : \beta \neq 0$. These tests can be conducted where the outright benefit tests are not conclusive to give an impression of whether any augmentation is occurring.

### 6.2.2 Augmentation Cost/Benefit

To determine whether applying augmentation is beneficial (or even sensible), it is important to understand the cost and benefits of augmentation wrt to the base learner to be augmented. This dynamic can be partly described by the *augmentation slope*, a simple approach introduced in this section.

If a model is to be beneficially augmented in any way, the base learner must be less than perfect. This is simply because augmentation, unless perfect itself, would detract from model accuracy. The error of the augmentation approach, $\epsilon_{\boldsymbol{M}}$, must not exceed the error of the base model, $\epsilon_B$, and any benefit of augmentation, $\prime\epsilon_{\boldsymbol{M}}$:

$$\epsilon_{\boldsymbol{M}} < \epsilon_B - \prime\epsilon_{\boldsymbol{M}}. \tag{6.10}$$

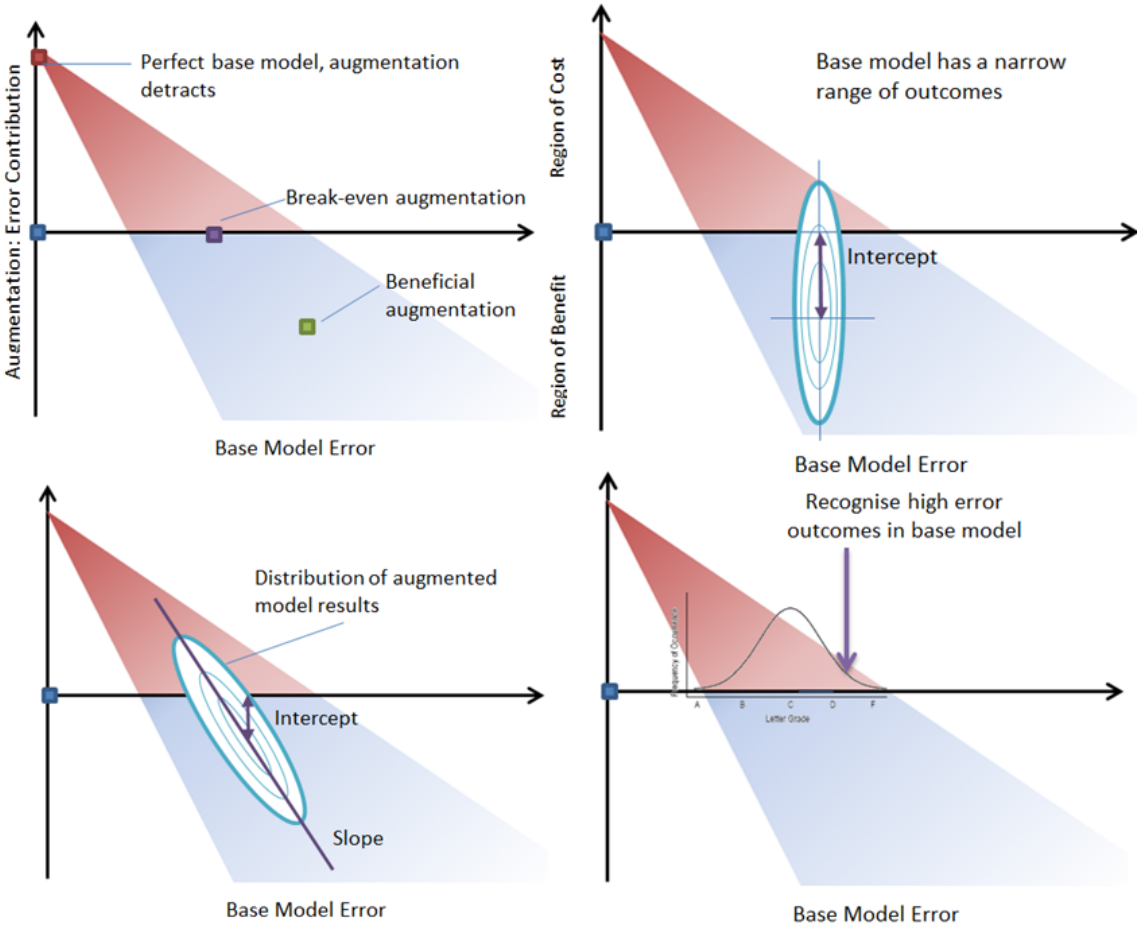
Generically, the effectiveness of an augmentation approach can simply be defined by the line of best fit of $\epsilon\prime_{\boldsymbol{M}}$ and $\epsilon_B$. This can be used to determine the break-even point of the augmentation benefit, which is explained in terms of the base learner error. The intercept term $\alpha_M$ and augmentation slope $\beta_M$ express a function of augmentation benefit. Moreover, $\alpha_M$ is the fixed benefit of augmentation, whereas the slope $\beta_M$ represents the combined error distributions for the base learner error and augmentation error. Although the intercept is more easily interpreted, the slope shows the change in augmentation benefit wrt the level of accuracy of a base learner. This dynamic is particularly important if the following criteria are true:

1. The base learner requires changing initialisation weights or hyperparameters if training is non-convex.

2. The learner is applied to different datasets, across which errors may vary.

The less accurate and broader the range of the error distribution of the base learner, the more potential exists for augmentation with all else equal (Figure 6.1),

Figure 6.1: Augmentation Slope




Note: (Top-left) Interplay between the base learner and augmentation approach is shown as a stylised cone anchored by the red square, denoting a "perfect" model with $\epsilon = 0$ where the augmentation can only add to the model error. The purple square denotes the *break-even* point, where the base model error equals the error augmentation. The green square is the point of rational augmentation, where augmentation benefits the base learner by a good margin. (Top-right) The degree of augmentation can be judged by the intercept and (Bottom-left) the *augmentation slope*. A negative slope implies a stronger augmentation benefit for every increase in the base learner error. (Bottom-right) It may be possible to recognise conditions under which the base learner error increases, which allows the selective application of augmentation.

implying that a negative slope coefficient ($\beta_M$) is an expected outcome. This is referred to as the *augmentation slope hypothesis*, which can be tested by statistically testing the line of best fit between the augmentation error and base learner error using an $F$-test where $Ho$: $\beta = 0$ (i.e. no difference between slope and intercept) and $Ha$: $\beta \neq 0$.

This simple model of augmentation can be used to test the efficacy of an augmentation approach, where the intercept and slope can be considered to have a $t$-distribution, making it possible to test the significance of the standard error of these coefficients. It may also be possible to anticipate the break-even augmentation of a CL approach applied to a certain dataset, which would further imply that the meta-learning of an augmentation benefit might be possible in an end-to-end CL framework. As we observe, this can approximately be achieved using a threshold value $J_{Crit}$.

## 6.3 Sliding Window Learners and Instance-based Similarity

### 6.3.1 Experimental Setup

The CLA approach was used in a regression task to forecast future expected returns of individual equity securities to drive an equity investment simulation in international developed and emerging market equities, a broad universe of opportunities. This is a cross-sectional forecasting task using two types of regression base learner: an FFNN (CLA-FFNN) and a linear (CLA-LIN) model. Moreover, DTW was taken as the similarity approach to drive the recall gate. Different memory-balancing approaches were also tested. Stock-level characteristics were used as the input dataset to batch-train both learners over all stocks in each period, forecasting US$ total returns 12 months ahead for each stock. When a forecast was in the top (bottom) decile, it was interpreted as a buy (sell) signal, and the corresponding simulated positions were taken in a crawl-forward fashion to result in indicative investment returns. The construction of the dataset is described, and the experimental setup is explained.

#### Dataset

Stock-level characteristics are commonly expressed using *factors* [61], and although CLA is designed as a complete reassessment of quantitative finance modelling approaches, factors are used here for comparative purposes. These were estimated

in-sample at each time step by regressing the style factor excess returns against each stock-level USD excess return stream:

$$r_{i,t} = \alpha_{i,t} + \beta_{MKT,i,t} x_{MKT,t} + \beta_{VAL,i,t} x_{VAL,i,t} + \epsilon_{i,t}, \tag{6.11}$$

where $r_{i,t}$ is the excess return of stock $i$ in period $t$, $x_{MKT,t}$ is the excess return of the All Countries World Ex-USA Equities Index, and $x_{VAL,t}$ is the relative return of the All Countries World Ex-USA Value Equities Index.

### Simulation Setup

Stock-level factor loadings populate a matrix, $\boldsymbol{X}$, which comprises the input data. Each row represents a stock appearing in the index at time $t$ (up to 4,500 stocks), and each column is related to a coefficient calculated on a specific time lag. In addition, $\boldsymbol{X}$ resulted from winsorising the raw input to eliminate outliers. An FFNN was trained in each period by separating the input data into training, cross-validation, and testing sets in 75/5/20 proportions. Separately, a simple linear regression learner was trained on 100% of the training data available at every time step. Long/short model portfolios were constructed every six months over the study term, simulating a rebalance every six months, using equally weighted long (buys) and short positions (sells). The simulation encompassed 4,500 international equities covering more than 30 countries across developed and emerging markets, corresponding to the All Countries World Ex-USA Equities Index between 2001 and 2017. [1] To account for the DTW sampling approach used, multiple simulation test runs were conducted for each test. Fifty simulations were run per test for this purpose. Both the *best* and separately *similarity-weighted* balancing approaches were tested. Testing was first conducted to investigate whether the results for CLA exhibited only an ensemble effect. An equally weighted balancing approach was also tested and generated weaker positive TRs relative to both the *best* and *similarity-weighted* balancing approaches, demonstrating that CLA exhibits more than an ensemble effect.

### Learner Setup

Both base learners were trained on cross-sectional data to give a simple model of the form:

$$E(R) = \boldsymbol{X}\boldsymbol{\theta}, \tag{6.12}$$

[1] Note that the first 24 months were used as a training period while testing, which was entirely out of sample and free from known data snooping biases, which started in 2003.

where $E(R)$ is the expected return. The CLA-LIN base learner consisted of simple linear parameters, $\boldsymbol{\theta}$, while the CLA-FFNN base learner was a shallow neural network approach. Both were trained using an L1 loss function. The CLA-FFNN consisted of one hidden layer of 10 units and sigmoid activations. Tuning measures were tested, such as testing difference activations, the hidden layer size, depth, and L2 versus L1 loss functions. The benefits were marginal.

### 6.3.2 Simulation Results

#### Long Only Tests

The CLA results for long and long/short simulations showed a significant return benefit over FFNN- and linear-base models, whereas tests of balancing approaches showed that the *similarity-weighted* approach outperformed the *best* approach. Long only tests produced positive absolute median TR (Figure 6.2 left) with statistically significant Sharpe ratios (a statistic representing risk-adjusted TRs) at the 5% level or better, across all tests, indicating that the overall results of the CLA augmented learners were beneficial (Table 6.1(a)). On the basis of the median results, the CLA-FFNN produced better TRs than the CLA-LIN. The results for different balancing showed that the *similarity-weighted* approach outperformed the *best* approach for CLA-FFNN and for the CLA-LIN base learners in terms of TRs.

Next, the augmentation benefit was considered (i.e. when CLA-FFNN and CLA-LIN were compared to simple base learner performance; Table 6.1). While both CLA-FFNN and CLA-LIN augmented base learner performance, CLA-LIN generated a higher RR (Figure 6.2 right). For the *similarity-weighted* balancing tests, the $t$-statistics of the IR were statistically significant for the median results at the 10% level or better for the FFNN tests and linear test. Sign tests on the augmentation hit rates for these tests were also statistically significant at the 5% level.

However, for the *best* approach tests, augmentation of both CLA-FFNN and CLA-LIN did not produce IRs with statistically significant $t$-statistics, although the hit rates for both tests were 76.5% and 60.8%, respectively, with statistically significant $p$-values at the 5% level from a *sign test*. This indicated that augmentation was evidenced but at a weaker level than for *similarity-weighted* tests.

Of all the tests conducted at this stage, the strongest augmentation benefit by return alone was for CLA-FFNN, using similarity-weighted balancing, generating 1.85%, followed by CLA-LIN using similarity-weighted balancing at 1.77%. It is also notable that all sign tests for augmentation across all median tests were statistically significant at the 5% level, with CLA-LIN generating a 100% hit rate in *similarity-weighted* testing. The positive performance of both CLA-FFNN and CLA-LIN tests

indicates that the CLA approach augmented both types of learner effectively.

Table 6.1: Long Only Simulation Tests, DTW, FFNN/Linear Base Learner, Similarity/Best (2002-2017 simulation tests)

**CLA-FFNN**

**DTW**

**Best**

| | | | | a) CLA Result: Total Return | | | | | b) Augmentation: Relative to Base Learner | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Obs | $T$ | $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | $p$ | $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | $p$ |
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 50 | 187 | 8.80% | 5.82% | 0.63 | 2.48 | 1.39% | **-2.67%** | 1.09% | -1.08 | -4.25 | 0.00% |
| **Median** | **50** | **187** | **13.19%** | **6.39%** | **0.84** | **3.32** | **0.11%** | **1.05%** | **1.66%** | **0.27** | **1.08** | **28.17%** |
| Mean | 50 | 187 | 12.90% | 7.00% | 0.75 | 2.97 | 0.33% | **0.97%** | 2.37% | 0.18 | 0.70 | 48.56% |
| Max | 50 | 187 | 15.68% | 8.61% | 0.73 | 2.90 | 0.42% | **4.80%** | 4.11% | 0.49 | 1.95 | 5.22% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 50 | 187 | **76.5%** | **23.5%** | **0.0%** | | | | | | | |

**DTW**

**Similarity**

| | | | | a) CLA Result: Total Return | | | | | b) Augmentation: Relative to Base Learner | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Obs | T | $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 50 | 187 | 11.10% | 5.27% | 0.87 | 3.42 | 0.08% | **-2.71%** | 1.48% | -1.83 | -3.18 | 0.17% |
| **Median** | **50** | **187** | **14.11%** | **5.81%** | **0.99** | **3.89** | **0.01%** | **1.85%** | **1.81%** | **1.02** | **1.74** | **8.36%** |
| Mean | 50 | 187 | 13.93% | 5.93% | 0.95 | 3.77 | 0.02% | **1.71%** | 1.99% | 0.86 | 1.46 | 14.50% |
| Max | 50 | 187 | 15.36% | 8.12% | 0.76 | 3.02 | 0.29% | **4.87%** | 4.04% | 1.21 | 2.02 | 4.51% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 50 | 187 | **90.1%** | **9.9%** | **0.0%** | | | | | | | |

**CLA-LIN**

**DTW**

**Best**

| | | | | a) CLA Result: Total Return | | | | | b) Augmentation: Relative to Base Learner | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Obs | T | $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 50 | 187 | 8.26% | 5.97% | 0.58 | 2.28 | 2.39% | **-2.24%** | 1.12% | -2.01 | -3.48 | 0.06% |
| **Median** | **50** | **187** | **11.12%** | **6.36%** | **0.72** | **2.84** | **0.50%** | **0.37%** | **1.48%** | **0.25** | **0.43** | **66.85%** |
| Mean | 50 | 187 | 11.30% | 6.70% | 0.69 | 2.74 | 0.68% | **0.52%** | 1.91% | 0.27 | 0.47 | 64.14% |
| Max | 50 | 187 | 14.67% | 8.59% | 0.69 | 2.73 | 0.69% | **3.58%** | 4.00% | 0.89 | 1.51 | 13.36% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 50 | 187 | **60.8%** | **39.2%** | **4.6%** | | | | | | | |

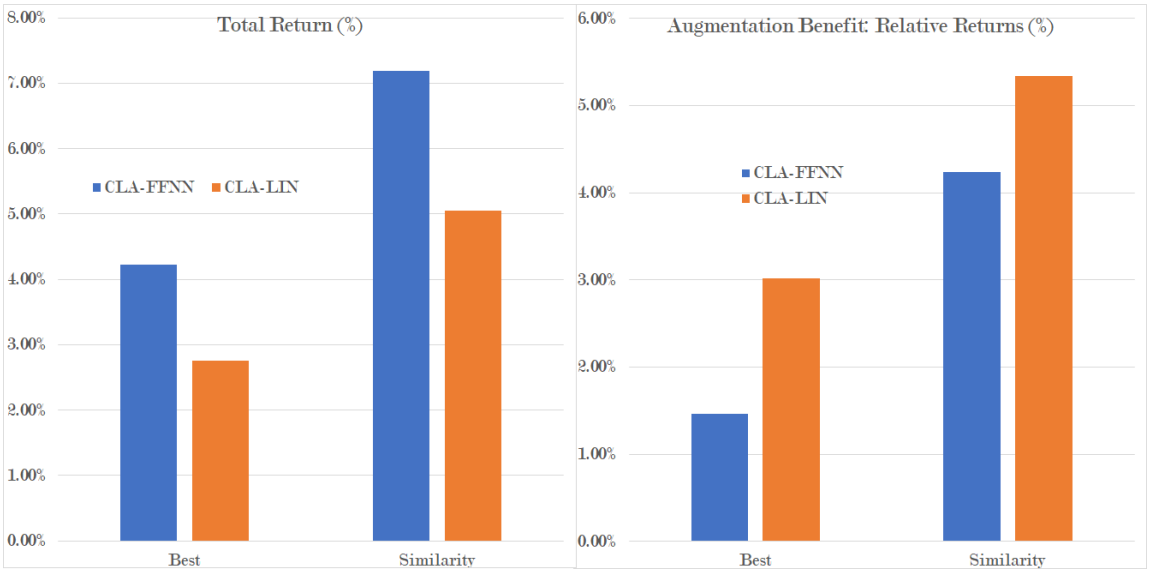
**DTW**

**Similarity**

| | | | | a) CLA Result: Total Return | | | | | b) Augmentation: Relative to Base Learner | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Obs | T | $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 50 | 187 | 12.09% | 5.63% | 0.88 | 3.47 | 0.06% | **1.24%** | 1.69% | 0.73 | 1.25 | 21.25% |
| **Median** | **50** | **187** | **12.69%** | **5.72%** | **0.91** | **3.58** | **0.04%** | **1.77%** | **1.77%** | **1.00** | **1.70** | **9.07%** |
| Mean | 50 | 187 | 12.69% | 5.72% | 0.91 | 3.58 | 0.04% | **1.78%** | 1.77% | 1.00 | 1.71 | 8.94% |
| Max | 50 | 187 | 13.83% | 5.81% | 0.97 | 3.82 | 0.02% | **2.79%** | 1.88% | 1.49 | 2.51 | 1.28% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 50 | 187 | **100.0%** | **0.0%** | **0.0%** | | | | | | | |

Notes: Long only investment simulations. Both CLA-FFNN and CLA-LIN were tested using $g_{best}$ (*best*) balancing and then using $g_{SimWeight}$ (*similarity weighted*). For each test, the minimum, maximum, mean, and median total return results are shown. Further, $TR_p$ (1) is the annualised total return, $\sigma_p$ (2) is the standard deviation, and $SR_p$ (3) is the Sharpe ratio. The augmentation benefit is shown as $RR_p$ (5), the annualised relative return of CLA over the base learner, the tracking error is $TE_p$ (6), and the information ratio is $IR_p$ (7). The $p$-values of the $t$-statistics of the $SR_p$ (4) and $IR_p$ are shown. The results of the sign tests are also shown for the hit rates, $pctPos$.

Figure 6.2: 6.3: Result Summary




Note: LSTM vs FFNN, long/short tests: Plot of median augmentation benefit by balancing method: *best* or *similarity weighted*.

### Long/Short Tests

The simulation results for long/short tests showed similarly positive results as the long only tests (Table 6.2). Whereas the long only tests demonstrated the ability of CLA to better forecast winning stocks, the long/short tests were designed to show whether CLA could also better identify the weakest stocks (i.e. those stocks to short).

The CLA approach produced positive absolute returns for all median long/short tests. Three of the four median tests showed TRs with a statistically significant Sharpe ratio $t$-statistics at the 5% level. The exception was for the median CLA-LIN using *best* balancing.

On the basis of the median results, the CLA-FFNN produced better total results than the CLA-LIN. The results for the different balancing approaches showed that the *similarity-weighted* approach outperformed the *best* approach for CLA-FFNN and for the CLA-LIN base learners.

Next, the augmentation benefit was considered. When long/short CLA-FFNN and CLA-LIN were compared to simple base learner performance (Table 6.2), all RRs for the median tests were positive, indicating a positive augmentation benefit for CLA. These tests also showed the IR, a statistic representing risk-adjusted returns

with statistically significant $t$-statistics at the 5% level for three of the four conducted tests. As with long only testing, CLA-LIN using the *best* approach was the one median test that did not have a statistically significant IR, although the hit rate was 76.5%, with a statistically significant p-value at the 5% level from a sign test. (In fact, this test was the weakest test in terms of value added from shorting the weakest stocks). The strongest augmentation benefit was demonstrated in CLA-LIN with *similarity-weighted* balancing at 5.34%, followed by CLA-FFNN similarity weighted at 4.24%. Again, all sign tests for augmentation across all median tests were statistically significant at the 5% level, with CLA-LIN generating a 100% hit rate in *similarity-weighted* testing.

Given that the augmentation benefits of long/short returns were more than double the equivalent of the long only, it indicates that CLA was particularly effective at identifying stocks that produced a poor future return and at shorting these weak performers. This might indicate that CLA has a stabilising effect on corner cases in the conducted tests.

Table 6.2: Long/Short Simulation Test, DTW, FFNN/Linear Base Learner, Similarity/Best Balancing

**CLA-FFNN**

**DTW**

**Best**

| | | Obs | T | a) CLA Result: Total Return $TR_p$ 1 | $\sigma_p$ 2 | $SR_p$ 3 | $t_{SR}$ | p 4 | b) Augmentation: Relative to Base Learner $RR_p$ 5 | $TE_p$ 6 | $IR_p$ 7 | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Min | 50 | 187 | -1.84% | 2.87% | -0.28 | -1.11 | 26.82% | **-5.98%** | 2.00% | -1.34 | -5.29 | 0.00% |
| | **Median** | **50** | **187** | **4.23%** | **3.51%** | **0.51** | **2.02** | **4.45%** | **1.47%** | **3.24%** | **0.20** | **0.77** | **44.20%** |
| | Mean | 50 | 187 | 4.32% | 3.97% | 0.46 | 1.83 | 6.93% | **1.52%** | 3.60% | 0.18 | 0.72 | 47.36% |
| | Max | 50 | 187 | 11.06% | 11.06% | 0.11 | 0.11 | 11.06% | **11.06%** | 11.06% | 0.11 | 0.11 | 11.06% |
| | | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| | **Hit Rate** | 50 | 187 | **76.5%** | **23.5%** | **0.0%** | | | | | | | |

**DTW**

**Similarity**

| | | Obs | T | a) CLA Result: Total Return $TR_p$ 1 | $\sigma_p$ 2 | $SR_p$ 3 | $t_{SR}$ | p 4 | b) Augmentation: Relative to Base Learner $RR_p$ 5 | $TE_p$ 6 | $IR_p$ 7 | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Min | 50 | 187 | 3.75% | 2.70% | 0.59 | 2.34 | 2.04% | **-0.72%** | 2.72% | -0.11 | -0.45 | 65.08% |
| | **Median** | **50** | **187** | **7.18%** | **3.15%** | **0.96** | **3.77** | **0.02%** | **4.24%** | **3.33%** | **0.54** | **2.14** | **3.37%** |
| | Mean | 50 | 187 | 7.10% | 3.21% | 0.93 | 3.66 | 0.03% | **4.06%** | 3.66% | 0.47 | 1.86 | 6.40% |
| | Max | 50 | 187 | 9.14% | 4.69% | 0.81 | 3.20 | 0.16% | **7.33%** | 8.81% | 0.35 | 1.38 | 17.01% |
| | | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| | **Hit Rate** | 50 | 187 | **94.1%** | **5.9%** | **0.0%** | | | | | | | |

**CLA-LIN**

**DTW**

**Best**

| | | Obs | T | a) CLA Result: Total Return $TR_p$ 1 | $\sigma_p$ 2 | $SR_p$ 3 | $t_{SR}$ | p 4 | b) Augmentation: Relative to Base Learner $RR_p$ 5 | $TE_p$ 6 | $IR_p$ 7 | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Min | 50 | 187 | -3.65% | 3.01% | -0.54 | -2.12 | 3.54% | **-3.41%** | 2.16% | -0.70 | -2.76 | 0.63% |
| | **Median** | **50** | **187** | **2.75%** | **3.37%** | **0.35** | **1.38** | **16.86%** | **3.01%** | **2.85%** | **0.45** | **1.79** | **7.54%** |
| | Mean | 50 | 187 | 2.54% | 3.64% | 0.30 | 1.18 | 24.01% | **2.81%** | 3.14% | 0.38 | 1.51 | 13.24% |
| | Max | 50 | 187 | 7.95% | 5.26% | 0.63 | 2.49 | 1.36% | **8.24%** | 5.05% | 0.68 | 2.69 | 0.78% |
| | | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| | **Hit Rate** | 50 | 187 | **86.3%** | **13.7%** | **0.0%** | | | | | | | |

**DTW**

**Similarity**

| | | Obs | T | a) CLA Result: Total Return $TR_p$ 1 | $\sigma_p$ 2 | $SR_p$ 3 | $t_{SR}$ | p 4 | b) Augmentation: Relative to Base Learner $RR_p$ 5 | $TE_p$ 6 | $IR_p$ 7 | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Min | 50 | 187 | 4.45% | 3.85% | 0.49 | 1.94 | 5.36% | **4.72%** | 3.21% | 0.62 | 2.46 | 1.47% |
| | **Median** | **50** | **187** | **5.05%** | **4.03%** | **0.53** | **2.09** | **3.76%** | **5.34%** | **3.32%** | **0.68** | **2.68** | **0.79%** |
| | Mean | 50 | 187 | 5.11% | 4.03% | 0.54 | 2.12 | 3.54% | **5.39%** | 3.32% | 0.69 | 2.71 | 0.74% |
| | Max | 50 | 187 | 6.50% | 4.23% | 0.65 | 2.55 | 1.15% | **6.77%** | 3.48% | 0.82 | 3.22 | 0.15% |
| | | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| | **Hit Rate** | 50 | 187 | **100.0%** | **0.0%** | **0.0%** | | | | | | | |

Notes: Long/short investment simulations. Similar tests to the long only were run but instead of just simulated purchases of the highest ranked stocks, short positions of a similar value were also taken in the lowest ranked stocks. For each test, the minimum, maximum, mean, and median TR results are shown, where $TR_p$ (1) is the annualised total return, $\sigma_p$ (2) is the standard deviation, and $SR_p$ (3) is the Sharpe ratio. The augmentation benefit is shown as $RR_p$ (5), the annualised relative return of CLA over the base learner, the tracking error is $TE_p$ (6), and the information ratio is $IR_p$ (7). The $p$-values of the $t$-statistics of the $SR_p$ (4) and $IR_p$ are shown. The results of sign tests are also shown for the hit rates, $pctPos$.

### Augmentation Benefits

Examining the distributions of the simulations for each CLA-FFNN test in Figures 6.3 and 6.4, as the base learner error increases, the augmentation benefit also increases, affirming the *augmentation benefit heuristic*. This is a negative slope in Figure 6.3, indicating a stronger augmentation benefit when the base learner error is higher, and vice versa. This property is exhibited by both CLA-FFNN simulation results in long only and long/short tests. The positive intercept terms for all CLA-FFNN tests (long and long/short) also indicate a positive outright benefit of CLA. These augmentation dynamics can further be examined using interpolations.

The $R^2$ are shown for the augmentation slopes for both the CLA-FFNN and CLA-LIN tests. The long-only tests show CLA-FFNN with *similarity-weighted* balancing with an augmentation slope of $R^2$ = 56.9% and an $F$-statistic that is statistically significant at the 5% level. Best tests using CLA-FFNN had an augmentation slope of $R^2$ = 7.5%, with an $F$-statistic (3.98) that is statistically significant at the 10% level ($p = 5.16\%$). The positive intercepts are also interesting for each slope, 0.1138 and 0.0594, for *similarity-weighted* and *best* balancing, respectively, (both with $t$-statistics that are statistically significant at the 5% level). This is a further indication of the positive augmentation benefit provided by CLA.

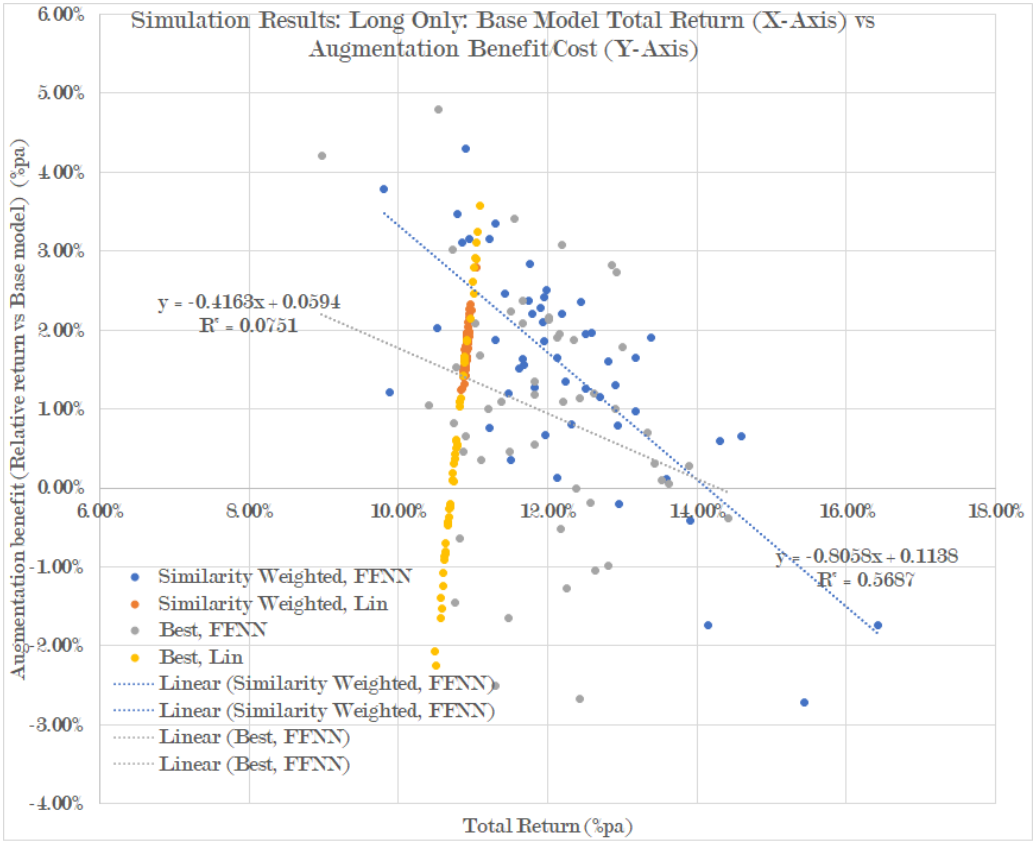
Long/short tests show *similarity weighted* CLA-FFNN with an augmentation slope of $R^2$ = 70.3% and an $F$-statistic that is statistically significant at the 5% level. The *best-weighted* tests using CLA-FFNN had an augmentation slope of $R^2$ = 16.7% and an $F$-statistic (9.97) that is statistically significant at the 5% level. Again, the augmentation slopes have positive intercepts for each slope, 0.0636 and 0.0328, for the *similarity-weighted* and *best* approaches, respectively (both with $t$-statistics that are statistically significant at the 5% level).

Figure 6.3 shows the *similarity-weighted* and *best* balancing approaches for the CLA-FFNN and CLA-LIN. Note that CLA-LIN when applied using either balancing approach, show very little variation in terms of TRs ($x$-axis), whereas the CLA-FFNN shows a wide variation, primarily owing to random-weight initialisation used by the FFNN (and not by CLA-LIN). Linear tests show high positive median values for long and long/short tests (Figure 5.1). In summary, the negative slopes exhibited particularly by the CLA-FFNN tests are supportive of the *augmentation benefit*.

### Memory Dynamics

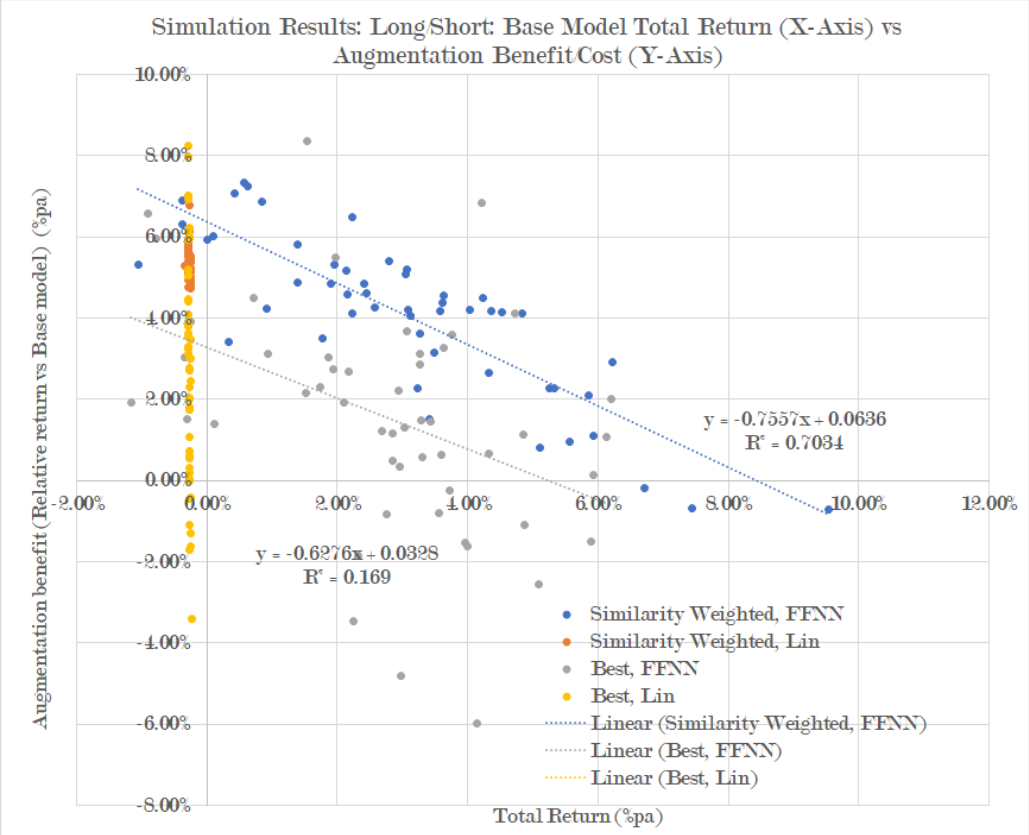
While the dynamics of remember and recall events of every CLA simulation run would be challenging to show succinctly, how CLA works can still be shown. Figure 6.5 shows the absolute error series of the base model, $\epsilon_B$ over one of the simulation

Figure 6.3: Memory Dynamics: Long Only




Note: Long only tests: Plot of all simulations for each test. The total return of the approach ($x$-axis) against the augmentation benefit (relative return versus base model, $y$-axis). *Augmentation slopes* are shown as simple linear lines of best fit.

Figure 6.4: Memory Dynamics: Long/Short




Note: Long/Short only tests: Plot of the total return of the approach ($x$-axis) against the augmentation benefit (relative return versus base model, $y$-axis). *Augmentation slopes* are shown as simple linear lines of best fit.

runs of CLA-FFNN. As CLA steps through time (left to right), $J_{Crit}$ is learned in discrete time. In addition, $J_{Crit}$ oscillates in the early periods before stabilising after the initial few years of the study period. This was typical of all conducted CLA tests and indicates that, after the initial variability, $J_{Crit}$ becomes stable within about 18 months of the model inception (Figure 6.5). This is consistent with the statistical basis for a residual change-driven remember gate described earlier in this study, where the central limits would indicate that the greater the samples in $\boldsymbol{\epsilon}_B$, the further its distribution tends towards normal.

It is also interesting that the level of $J_{Crit}$ appears to mirror the level of $\epsilon_B$ over this period. It appears that $J_{Crit}$ is adjusted down as the base learner absolute error series also falls. This implies that the base learner achieved better forecasting towards the end of the study period, which caused $J_{Crit}$ to decrease to achieve an optimal remembering level.

The spikes in $\epsilon_B$ also appear significant in the domain of application. After early oscillation, 2006 represents the start of the subprime crisis, where 2008 represents the failure of Lehman Brothers, and 2009 indicates the significant rally that followed the substantial quantitative easing by Western central banks. In addition, 2013 and 2014 represent the start of the European Central Bank's announcement of quantitative easing.

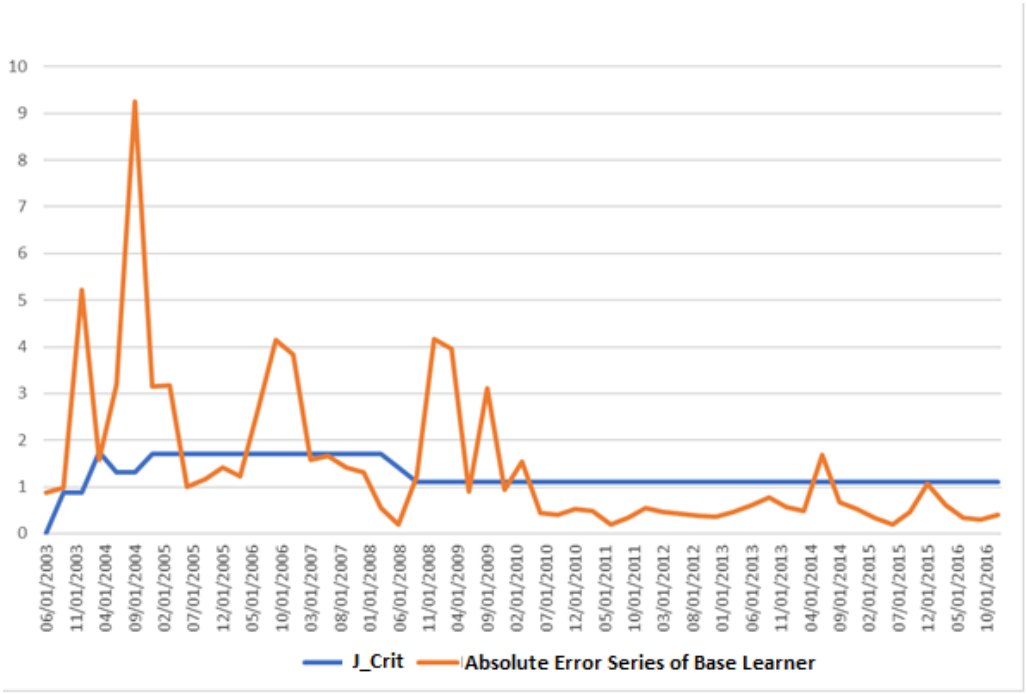
As $J_{Crit}$ can be considered a latent variable related to the distribution of $\epsilon_B$ as states change in the input data, a stable learned value of $J_{Crit}$ could be interpreted as an indication of the stability of the residual change-based remember gate.

### Explainable Memory

The CLA approach produces results that can be explained by examining which past learners have been applied to which approximately repeating states, and with further investigation, can explain why. Figure 6.6 shows an example of a simulation run, where 6.6a shows how the value of \$1 would have changed if invested in an investment strategy driven by CLA and, separately, by the base model. Figure 6.6(b) shows the memory structure of CLA, where a new memory can theoretically be appended at every step forwards in the simulation, although only four memories were remembered in this example.

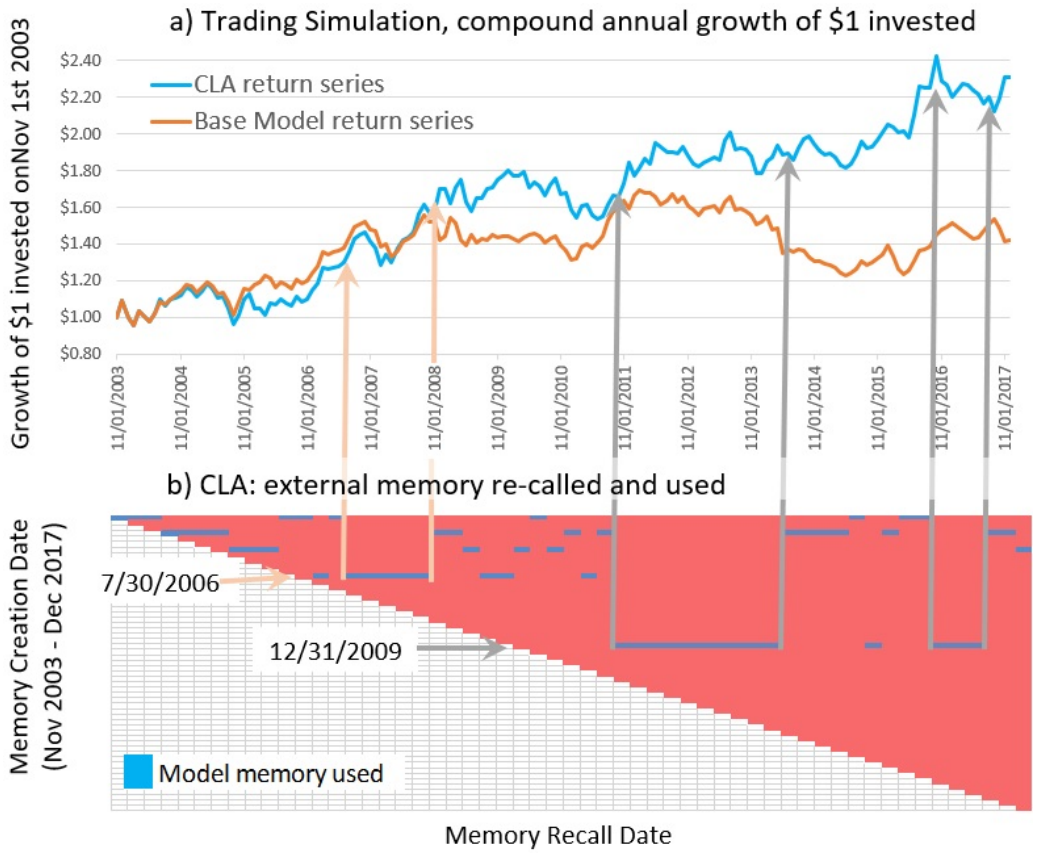
Two memories are examined. First, a memory formed in the period ending July 2006, which is used by CLA to outperform the base model in the period Sep 2007 to Oct 2008. Interestingly, this is over the period of the *Quant Quake* until just after the collapse of Lehman Brothers during the 2008 financial crisis. Second, a memory formed in the period ending December 2009 is used by CLA between 2011

Figure 6.5: Memory Dynamics (CLA-FFNN): Learning $J_{Crit}$




Note: $J_{Crit}$ is learned over time to define change points in the absolute error series $\epsilon_B$ of the base learner. It is notable that, as time passes, the error series becomes more stable, and as a result, $j_{Crit}$. This is consistent with the central limits described earlier in this thesis.

Figure 6.6: Memory Dynamics: Explainable Memory Addressing




Note: Explainable memory: How recalled memories contribute to simulation performance. The graph in a) uses a single example simulation and shows the growth of a $1 investment in 2003, using strategies driven by CLA or the base model, and b) shows a representation of the CLA memory structure, a memory triangle, where each row in the expanding triangle represents a potential memory. This external memory structure can grow by one memory at each step forward in the simulation, although, in practice, only four memories were remembered in this simulation. The top row in the memory-triangle graphic represents the base model. The memory with the highest weight in each period is highlighted.

and mid-2014, the period affected by the eurozone crisis and subsequent recovery. It is also used by CLA, albeit to a far lesser effect, in late 2016.

### 6.3.3 Discussion

In all cases, CLA using DTW and either *best* or *similarity-weighted* balancing with either FFNN or linear base learners has been shown to be capable of accumulating knowledge of changing states over time to produce a statistically significant augmentation benefit. Given our ability to examine memory remember and recall events and the sequential learning of $j_{crit}$, CLA is able to separate the temporal problem space sequentially into different states, and these memories can be used to aid future decision-making. Residual change, as a memory concept, has been effectively applied to TCL. Multivariate DTW has also been successfully applied as a similarity measure to drive a recall gate in a TCL memory structure.

The combined statistical testing framework has been used to establish that CLA has significantly augmented base learners in the application to the tested problem, which includes the 1) Sharpe ratio of the outright benefit of CLA, 2) IR of the augmentation benefit, 3) hit rates of the consistency of the augmentation, and 4) augmentation curve properties using *augmentation-slope* analysis.

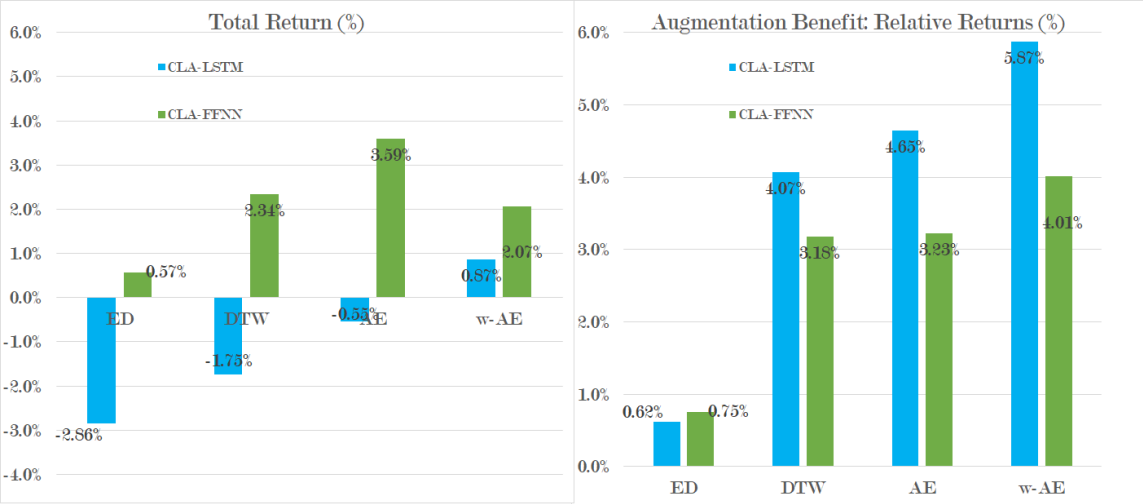
Moreover, CLA produces a positive, statistically significant forecasting benefit using FFNN and linear base models. Long and long/short tests show positive and statistically significant outright returns and a positive and statistically significant augmentation benefit relative to the base learners. The *similarity-weighted* model memories produce stronger results than simply picking the *best* model memory. If CLA were exploited in practice, the outperformance shown here would likely give a significant advantage to investment strategy returns.

## 6.4 Recurrent Learners and AE Similarity

### 6.4.1 Experimental Setup

First, a sliding window approach was tested using an FFNN and then a sequential learner, an LSTM. The FFNN applied as a sliding window was the best performer. Second, different time series similarity approaches were tested, which were used to drive CLA's memory-recall gate. The simple ED was found to underperform noise-invariant similarity approaches, DTW, and AEs. The best performing similarity approach was a hybrid introduced in this study, warp-AE. Third, the augmentation slope was again used to analyse the CLA results.

Figure 6.7: Summary of Test Results




Note: LSTM vs FFNN, Long Short tests: Plot of median augmentation benefit by distance measure.

Again, four-point statistical testing was conducted and indicated the augmentation benefits of CLA. In these tests, a fifth point was also introduced, monotonicity testing, which raised the bar further for analysing augmentation.

In the next section, an investment simulation setup is described, specifically addressing the use of SAEs or use in the CLA remember gate with warp-AE. In the second section, experiments are described for the sliding window and sequential learners, which were applied with different similarity approaches. Moreover, the benefits and costs of CL implementation choices are discussed, and memory use is explained. Finally, the results of the experiments in this chapter are discussed.

### Dataset

Factor loadings, as described above, were again used as a simple dataset for testing CLA.

As before, stock-level factor loadings populate a matrix, $\boldsymbol{X}$, which comprised the input data. Each row represents a stock appearing in the index at time $t$ (up to 5,500 stocks) and each column is related to a coefficient calculated on a specific time lag. In addition, $\boldsymbol{X}$ resulted from winsorising the raw input to eliminate any outliers.

### Simulation Setup

Again, a regression task to forecast the future expected returns of individual equity securities was used, but this time, it just focused on an alternative dataset of emerging market equities. This is used to drive equity investment simulations. The dataset consisted of stock-level characteristics at each time step. Tests were conducted to show the relative performance of a sliding window base learner, FFNN, and a sequential base learner, LSTM. Different similarity approaches were also used to drive the memory recall gate: ED, DTW, AE, and warp-AE.

In addition, the base learners were again batch-trained over all stocks at each time step, forecasting TRs in USD 12 months ahead for each stock. For the sliding window learner, a year-long, fixed-length sliding window of four quarters was used for training, and for the sequential learner, all historic data up to the current time were used for training. A stock-level forecast in the top (bottom) decile of the stocks in a period was interpreted as a buy (sell) signal. Only long/short tests were examined for these sequential tests.

The long/short model portfolios were constructed (i.e. rebalanced) every 6 months over the study term using equally weighted long (buys) and short positions (sells). The simulation encompassed 5,500 equities in total, covering 26 countries across emerging markets, corresponding to an Emerging Market Equities Index between 2006 and 2017. To account for the DTW sampling approach used and the differences in the random initialisation of neural components, several simulations were carried out per test.

Additionally, the monotonicity testing was conducted. As each conducted test resulted in a return forecast for each stock in each period, it was possible to calculate the performance for each decile of stocks in the sort order for each test. This resulted in 10 simulations conducted for every test. This was conducted over the test term for all tests. At each rebalance date in the simulation, all stocks were sorted by expected return. Whereas the top and bottom deciles of the stocks were taken to be buys for the main tests, deciles 2 to 10 were also selected to result in a total of 10 long only portfolio strategy simulations, each equating to a decile. These decile portfolios were used for monotonicity analysis to analyse, for example, whether decile 10 (the highest expected return for forecasted stocks) outperformed decile 9 and whether 9 outperformed 8 and so on. (Please see [58] for an exhaustive explanation).

### Learner Setup

The CLA-FFNN base learner took the form demonstrated in Figure 6.3.1, whereas the CLA-LSTM setup used an expanding window with more hidden units. The

LSTM base learner was trained using an Adam optimiser with 100 hidden units and a learning rate of 0.005, using an L2 loss. Again, the tuning measures were tested, such as different numbers of hidden units, and the benefits were again marginal. Changing the training window size had a material bearing on performance. If a sliding window was used to train the LSTM, the performance was greatly reduced compared to using an expanding window containing all past data. This has a material bearing on the resource usage of LSTM approaches versus FFNN when used in CLA.

### 6.4.2 Simulation Results

The CLA results showed a significant augmentation benefit for both base learners (see Figure 6.7, right). While tests of similarity approaches favoured adjustments for noise over simple ED, All CLA-FFNN TRs were positive, whereas most CLA-LSTM TRs were negative (see Figure 6.7, left). It was noted, however, that for median results, CLA provided a positive augmentation benefit (RR) over base learners in all cases.

Testing established that CLA-LSTM was not particularly effective, with only one test giving a positive TR (CLA-LSTM, warp-AE). The CLA-FFNN showed more promising returns with all four median tests showing positive TRs. These results might encourage the use of sliding window approaches, such as FFNN, over LSTMs in complex tasks driven by noisy time series of this nature. The CLA approach was found to augment all implementations of both learners. The results are illustrated in Table 6.3) for CLA-FFNN and in Table 6.4 for CLA-LSTM. We first discuss the differences between CLA-FFNN and CLA-LSTM, then ED vs DTW similarity for both learners. Then, we compare the AE distance.

The CLA-FFNN outperformed all the equivalent sequential learner tests in terms of TR, while the Sharpe ratios (Figure 6.9) were also superior. However, the augmentation benefit, gauged by RR and IR, was superior for CLA-LSTMs (6.8), although most augmentation tests for both learners were statistically significant at the 5% level (albeit the overall TR was still lower for all LSTM tests). All median tests evidenced a high consistency of augmentation by CLA. This was shown in the high hit rates with statistically significant sign tests at the 5% level for all median tests. This indicated that CLA had augmented both FFNN and LSTM learners using all distance approaches.

Tests of the different similarity approaches used in the recall gate showed varied results. The ED underperformed DTW tests for both CLA-FFNN and CLA-LSTM, in terms of both TR and augmentation benefit: RR and IR. This implies that the invariance to phase that DTW provides is an important consideration in a real-world

context. This was also demonstrated by the statistically significant IR $t$-statistics at the 10% level for DTW and not for the ED tests on both learners. (As noted though, all median tests for both learners showed high positive hit rates, indicating that augmentation occurred for all cases).

Next, the AE tests were examined for both learners. The AE distance showed higher TRs than DTW and demonstrated IRs with statistically significant $t$-statistics at the 10% level for both CLA-LSTM and CLA-FFNN, indicating that the AE distance is an appropriate approach. Moreover, warp-AE generated the highest single median test RR (5.87%) and IR (0.76) of all these tests, implying that adding a DTW filter to the AE distance is an interesting approach in this context. Whereas CLA-FFNN exhibited higher outright performance (TR), CLA-LSTM demonstrated a better augmentation benefit (RR). Again, all median tests using AE and warp-AE evidenced a high consistency of augmentation by CLA. This was shown in the high hit rates with statistically significant sign tests at the 5% level for all median tests. (We also observe that *augmentation slopes* statistically support this in the next section).

In summary, CLA proved successful for both LSTMs and FFNN approaches in all similarity tests with the weakest performance from the ED (in terms of RR and statistical significance). In addition, CLA across all tests generated RR of between 3.18% (CLA-FFNN, DTW) and 5.87% (CLA-LSTM, warp-AE) of augmentation benefits, depending on the similarity measure used, except for ED. Moreover, CLA-LSTM produced stronger augmentation results than CLA-FFNN, with generally higher RR and higher IRs. The FFNN augmentation showed good results with IRs ranging from 0.51 to 0.62 across similarity measures, with $t$-statistics that were statistically significant at the 10% level, with the exception of ED (IR = 0.20).

Table 6.3: Long/Short Simulation Test, FFNN Base Learner

**CLA-FFNN — Similarity Weighted — ED**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -4.69% | 2.65% | -0.79 | -2.73 | 0.72% | **-1.96%** | 1.17% | -0.73 | -2.55 | 1.18% |
| **Median** | **5** | **187** | **0.57%** | **2.83%** | **0.09** | **0.30** | **76.14%** | **0.75%** | **1.63%** | **0.20** | **0.70** | **48.74%** |
| Mean | 5 | 187 | -0.26% | 3.17% | -0.04 | -0.12 | 90.08% | **0.65%** | 1.63% | 0.17 | 0.60 | 54.87% |
| Max | 5 | 187 | 2.86% | 4.90% | 0.25 | 0.86 | 38.85% | **2.92%** | 6.61% | 0.19 | 0.66 | 51.00% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **86.3%** | **13.7%** | **0.00%** | | | | | | | |

**CLA-FFNN — Similarity Weighted — DTW**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | 1.16% | 2.55% | 0.20 | 0.68 | 49.72% | **-0.80%** | 1.96% | -0.18 | -0.62 | 53.43% |
| **Median** | **5** | **187** | **2.34%** | **2.84%** | **0.35** | **1.23** | **22.24%** | **3.18%** | **2.26%** | **0.60** | **2.10** | **3.78%** |
| Mean | 5 | 187 | 2.48% | 3.15% | 0.34 | 1.17 | 24.33% | **3.06%** | 2.26% | 0.58 | 2.02 | 4.52% |
| Max | 5 | 187 | 11.06% | 11.06% | 0.11 | 0.11 | 11.06% | **11.06%** | 11.06% | 0.11 | 0.11 | 11.06% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **76.5%** | **23.5%** | **0.00%** | | | | | | | |

**CLA-FFNN — Similarity Weighted — AE**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -0.87% | 6.04% | -0.06 | -0.22 | 82.79% | **-0.48%** | 2.08% | -0.10 | -0.35 | 72.80% |
| **Median** | **5** | **187** | **2.07%** | **6.04%** | **-0.06** | **-0.22** | **82.79%** | **4.01%** | **2.74%** | **0.62** | **2.18** | **3.12%** |
| Mean | 5 | 187 | 3.00% | 6.47% | 0.14 | 0.48 | 63.51% | **3.65%** | 2.74% | 0.57 | 1.98 | 4.91% |
| Max | 5 | 187 | 8.13% | 7.89% | 0.16 | 0.56 | 57.33% | **8.36%** | 4.01% | 0.87 | 3.04 | 0.28% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **100.0%** | **0.0%** | **0.00%** | | | | | | | |

**CLA-FFNN — Similarity Weighted — wAE**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -4.04% | 2.70% | -0.66 | -2.30 | 2.28% | **-6.55%** | 1.97% | -1.49 | -5.21 | 0.00% |
| **Median** | **5** | **187** | **3.59%** | **3.13%** | **0.49** | **1.69** | **9.27%** | **3.23%** | **2.69%** | **0.51** | **1.79** | **7.57%** |
| Mean | 5 | 187 | 2.66% | 3.54% | 0.32 | 1.11 | 26.69% | **1.90%** | 2.69% | 0.30 | 1.06 | 29.01% |
| Max | 5 | 187 | 6.17% | 4.98% | 0.52 | 1.81 | 7.26% | **6.53%** | 3.59% | 0.76 | 2.67 | 0.85% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **94.1%** | **5.9%** | **0.00%** | | | | | | | |

Notes: Long-short investment simulation results on the emerging market equity universe using an FFNN base learner. See the median results from five simulation runs per test. The simulations were for long/short emerging market equities between 2006 and 2017. No transaction costs were considered.

Table 6.4: Long/Short Simulation Test, LSTM Base Learner

**CLA-LSTM — Similarity Weighted — ED**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -5.63% | 3.35% | -0.75 | -2.60 | 1.02% | **-2.04%** | 3.15% | -0.28 | -0.99 | 32.31% |
| **Median** | **5** | **187** | **-2.86%** | **3.57%** | **-0.35** | **-1.22** | **22.34%** | **0.62%** | **3.28%** | **0.08** | **0.29** | **77.52%** |
| Mean | 5 | 187 | -2.85% | 3.56% | -0.35 | -1.22 | 22.43% | **1.95%** | 3.28% | 0.26 | 0.89 | 37.41% |
| Max | 5 | 187 | -0.35% | 3.73% | -0.04 | -0.14 | 88.78% | **8.95%** | 3.69% | 1.01 | 3.52 | 0.06% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **86.3%** | **13.7%** | **0.00%** | | | | | | | |

**CLA-LSTM — Similarity Weighted — DTW**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -2.84% | 4.34% | -0.29 | -2.28 | 2.40% | **0.33%** | 2.81% | 0.05 | 0.18 | 86.11% |
| **Median** | **5** | **187** | **-1.75%** | **4.58%** | **-0.17** | **-1.33** | **18.50%** | **4.07%** | **3.33%** | **0.52** | **1.82** | **7.13%** |
| Mean | 5 | 187 | -1.50% | 4.57% | -0.14 | -1.15 | 25.28% | **4.18%** | 3.33% | 0.54 | 1.87 | 6.39% |
| Max | 5 | 187 | 11.06% | 11.06% | 0.11 | 0.11 | 11.06% | **11.06%** | 11.06% | 0.11 | 0.11 | 11.06% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **76.5%** | **23.5%** | **0.00%** | | | | | | | |

**CLA-LSTM — Similarity Weighted — AE**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -3.28% | 3.29% | -0.44 | -1.52 | 12.96% | **-1.35%** | 2.18% | -0.27 | -0.95 | 34.48% |
| **Median** | **5** | **187** | **-0.55%** | **3.45%** | **-0.07** | **-0.24** | **81.07%** | **4.65%** | **2.64%** | **0.75** | **2.61** | **1.00%** |
| Mean | 5 | 187 | -0.22% | 3.48% | -0.03 | -0.09 | 92.54% | **4.81%** | 2.64% | 0.77 | 2.70 | 0.78% |
| Max | 5 | 187 | 2.15% | 3.64% | 0.25 | 0.88 | 38.02% | **11.34%** | 3.20% | 1.46 | 5.08 | 0.00% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **100.0%** | **0.0%** | **0.00%** | | | | | | | |

**CLA-LSTM — Similarity Weighted — wAE**

| | Obs | T | a) CLA Result: Total Return $TR_p$ | $\sigma_p$ | $SR_p$ | $t_{SR}$ | p | b) Augmentation: Relative to Base Learner $RR_p$ | $TE_p$ | $IR_p$ | $t_{IR}$ | p |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | | 4 | 5 | 6 | 7 | | |
| Min | 5 | 187 | -0.39% | 4.60% | -0.04 | -0.13 | 89.89% | **2.53%** | 2.79% | 0.39 | 1.35 | 17.76% |
| **Median** | **5** | **187** | **0.87%** | **4.81%** | **0.08** | **0.27** | **78.69%** | **5.87%** | **3.27%** | **0.76** | **2.64** | **0.92%** |
| Mean | 5 | 187 | 0.97% | 4.75% | 0.09 | 0.30 | 76.09% | **6.18%** | 3.27% | 0.80 | 2.77 | 0.63% |
| Max | 5 | 187 | 2.15% | 4.85% | 0.19 | 0.66 | 51.02% | **11.34%** | 3.80% | 1.23 | 4.28 | 0.00% |
| | | | **pctPos** | **pctNeg** | $p$ | | | | | | | |
| **Hit Rate** | 5 | 187 | **94.1%** | **5.9%** | **0.00%** | | | | | | | |

Notes: Long-short investment simulation results on the emerging market equity universe using the LSTM base learner. See the median results from five simulation runs per test. The simulations were for long/short emerging market equities between 2006 and 2017. No transaction costs were considered.

## Augmentation Benefit

The augmentation dynamic of these tests can be analysed as an *augmentation slope* (Figure 6.8), where both CLA-FFNN and CLA-LSTM tests show the characteristic negative slope across tests. This supports the *augmentation benefit heuristic*, where

the greater a base learner error, the higher the augmentation that is possible. In this section, we take the analysis further by examining the *break-even* cost of augmentation, which can be modelled from the *augmentation slope*.

The line equations of the *augmentation slope* define both the benefit and cost of each learner's augmentation. For CLA-FFNN, the augmentation slope (i.e. the line of best fit in Figure 6.8) has $R^2 = 71\%$, with a statistically significant $F$-statistic at the 5% level. For CLA-LSTM, $R^2 = 28\%$, with a statistically significant $F$-statistic (8.54) at the 5% level ($p = 0.78\%$). In the case of CLA-FFNN, a 2.0% intercept or fixed benefit of augmentation exists, while the variable cost/benefit increases by 0.76% for every 1% weaker the base learner performs (and deteriorates by 0.76% for every 1% increase). For CLA-LSTM, the fixed cost of augmentation is -1.0% while the variable cost/benefit of augmentation is 1.03% for every 1% weaker the base model performs. The variable cost/benefits imply that if the FFNN and LSTM base learners generally performed at > 2.0% and > 0.0%, respectively, the augmentation benefit of CLA would, on average, become a net cost. In other words, this is the implied *break-even* point of the CLA approach using these specific models and datasets, given the task at hand. Notably, although CLA-LSTM appears to benefit more than CLA-FFNN from augmentation overall, CLA-FFNN is less dependent on the performance of the base learner than CLA-LSTM. Both learners benefit from CLA augmentation. The FFNN base learner is a better proposition for a CL-based approach using CLA, with a higher fixed benefit and a higher break-even performance level.

### Monotonicity Analysis

Although statistical testing has strongly indicated an augmentation benefit for CLA in this testing, we add another analytical dimension: monotonicity testing. As described earlier, for every test conducted in this section, 10 simulations were run, where decile 10 was a simulation generated using the most attractive stocks, and decile 1 was the least attractive according to the expected return forecasts of CLA. The result would ideally be that, for all tests, a strictly increasing TR would occur from the decile 1 portfolio simulation to that for decile 10 for every test. Clearly, this is a very high bar to set, but this style of monotonicity testing was conducted on the median test results from these tests: CLA-FFNN and CLA-LSTM for each of the four tested distance measures. These results are shown as RR of the CLA augmented learner versus the base learner.

For each of the four CLA-FFNN median test results, RR, Figure 6.10 showed a generally positively monotonic result, where the lower decile portfolios underperformed the higher deciles, noting the positive slopes of the interpolation lines. This indicated

that CLA augmented the base FFNN learner as expected.

Again, for each of the four CLA-LSTM median test results, RR, Figure 6.9 showed a generally positively monotonic result, where the lower decile portfolios underperformed the higher deciles, noting the positive slopes of the interpolation lines. This indicated that CLA augmented the base FFNN learner as expected.

Statistical testing was conducted to determine statistically significant monotonicity using the Mann-Kendall test and a test for the lines of best fit (shown in Figures 6.10 and 6.9). These test results can be seen in Table 6.5 and show a good degree of statistically significant monotonicity in the augmentation results. Moreover, CLA-FFNN has statistically significant Mann-Kendall $p$-values for DTW and AE at the 10% level. Additionally, CLA-LSTM has statistically significant Mann-Kendall $p$-values for AE and DTW at the 10% level. In both cases, the $F$-test and the $R^2$ support these results.

Whereas AE and warp-AE show strong results in the first four points of testing, in the fifth, an approximately positive monotonicity can be observed from the decile TR and line of best-fit slopes. This is not statistically significant. Additionally, the augmentation benefit provided by the ED may not be high enough to be statistically significant in terms of the Sharpe ratio or IR but is statistically significant in terms of the hit rate and monotonicity of results, perhaps indicating a lower degree of augmentation from the ED.

Table 6.5: Augmentation Monotonicity

| | | **Mann-Kendall** | **Linear line of best fit** | | |
|---|---|---|---|---|---|
| | | **p** | **$R^2$** | **F-stat** | **p** |
| FFNN | **Euclidean distance** | 10.7% | 32.40 | 3.84 | **8.6%** |
| | **Dynamic time warping** | 15.2% | 32.90 | 3.92 | **8.3%** |
| | **Autoencoder** | 47.4% | 20.50 | 2.07 | 18.9% |
| | **Warp-autoencoder** | **0.2%** | 79.10 | 30.20 | **0.0%** |
| LSTM | **Euclidean distance** | 20.1% | 5.24 | 0.44 | 52.5% |
| | **Dynamic time warping** | 28.3% | 10.01 | 0.90 | 37.1% |
| | **Autoencoder** | **3.8%** | 57.20 | 10.70 | **1.2%** |
| | **Warp-autoencoder** | **3.2%** | 54.60 | 9.64 | **1.5%** |

Notes: Statistical testing of monotonicity over the quantile charts for Figure 6.9. Mann-Kendall $p$-values are listed along with $R^2$ and the statistical testing of a linear interpolation over the quantiles. FFNN: feed-forward neural network; LSTM: long short-term memory.

### Similarity Versus Error

Because CLA depends on similarity to guide memory selection, some level of correlation is expected to exist between the similarity of a learner memory and the out-of-sample error. This was found to be the case where the results of each similarity approach were tested for consistency in the correlation between the distance and error. This could be observed over time and cross-sectionally for each memory at each time step.

Memory-recall diagnostics were used to test the consistency of correlation between distance and error across memories. Using the expanding triangle of memories, seen in Figure 6.11, all possible memories were compared with the out-of-sample performance of each. At each time step, a cross-sectional correlation coefficient was calculated across memories. Separately, time series correlation coefficients were calculated for each memory over time. Each was tested using a sign test, and the results are reported in Table 6.6. This was intended to sense check the assumption that the error was dependent on similarity.

As expected, each similarity correlated fairly consistently with the forecasting error across memories in a given period and over time, with the sign tests showing significant $p$-values at the 5% level for all approaches cross-sectionally and over time. This would indicate that the different conducted similarity tests are generally associated with the model error, which indicates that these similarity measures are, in principle, good drivers of recall for a CL recall gate in a noisy time series context.

Table 6.6: Similarity vs Error in Memory Recall

| | Cross-sectional Correlation | | | |
|---|---|---|---|---|
| | **ED** | **DTW** | **AE** | **wAE** |
| **pctPos** | 70% | **79%** | 77% | 79% |
| **pctNeg** | 28% | 21% | 23% | 21% |
| ***p*-value** | **0.33%** | **0.00%** | **0.01%** | **0.00%** |

| | Time series Correlation | | | |
|---|---|---|---|---|
| | **ED** | **DTW** | **AE** | **wAE** |
| **pctPos** | 98% | **77%** | 65% | 100% |
| **pctNeg** | 2% | **23%** | 35% | 0% |
| ***p*-value** | **0.00%** | **0.01%** | **1.78%** | **0.00%** |

Notes: Correlations of similarity and forecasting error for sampled simulation runs for each similarity approach. pctPos: the percentage occurrence of positively signed correlation coefficients taken a) cross-sectionally (i.e. the correlation of similarity and error across all memories at each time step) and b) longitudinally (i.e. the correlation of similarity and error across all time steps for each memory).

### Explainable Memory

The CLA approach produces outcomes that can be explained and attributed to its memory. Figure 6.11 shows an example of one simulation run using a FFNN learner and AE distance and shows how certain memories were applied at certain time points to result in specific outcomes. The expanding triangle (bottom) shows how memories can be added at each time step forward that the learner takes. Bold lines represent memories that dominated memory recall at different points in time. In this case, at least three memories are remembered (lower chart, black lines) and recalled at different times. A learner memory remembered in January 2007, a period of turbulence in the financial markets, adds the most value (top: note the largest increase in the area chart (CLA return) versus the line (base learner return)). This memory is more appropriate than the base learner in the period of the 2008 financial crisis and its aftermath involving concerted fiscal stimulus (September 2008 to December 2010). It was again recalled in 2013 and then in 2016, which are both periods where fiscal stimulus also dominated the market returns (in Europe and China, respectively).

**Memory Resource Cost**

The choice of distance measure came with a material difference in machine memory overhead. ED and DTW approaches being far more memory intensive than AE approaches. ED and DTW required training exemplars to be stored as a contextual reference in each memory column. In this experiment this resulted in approximately 30,000 64bit floating point variables requiring storing for each memory. Each memory might therefore require 240KB to store a single contextual reference. (Notably given 32GB RAM resources of a modern, high spec personal computer, this would allow for over 130,000 separate memories to be saved in RAM alone, more than enough for the domain considered herein). However, the use of AEs dramatically reduced memory requirements. Given the typical encoder/decoder AE used in this study was a two layered AEs of ten and five hidden units respectively, the memory required equated to approximately 2x200 64bit floating point variables for an AE contextual reference in each memory column. This in turn equated to 3.2KB per memory, only 1.3% of the memory required for the equivalent ED or DTW driven remember-gate.

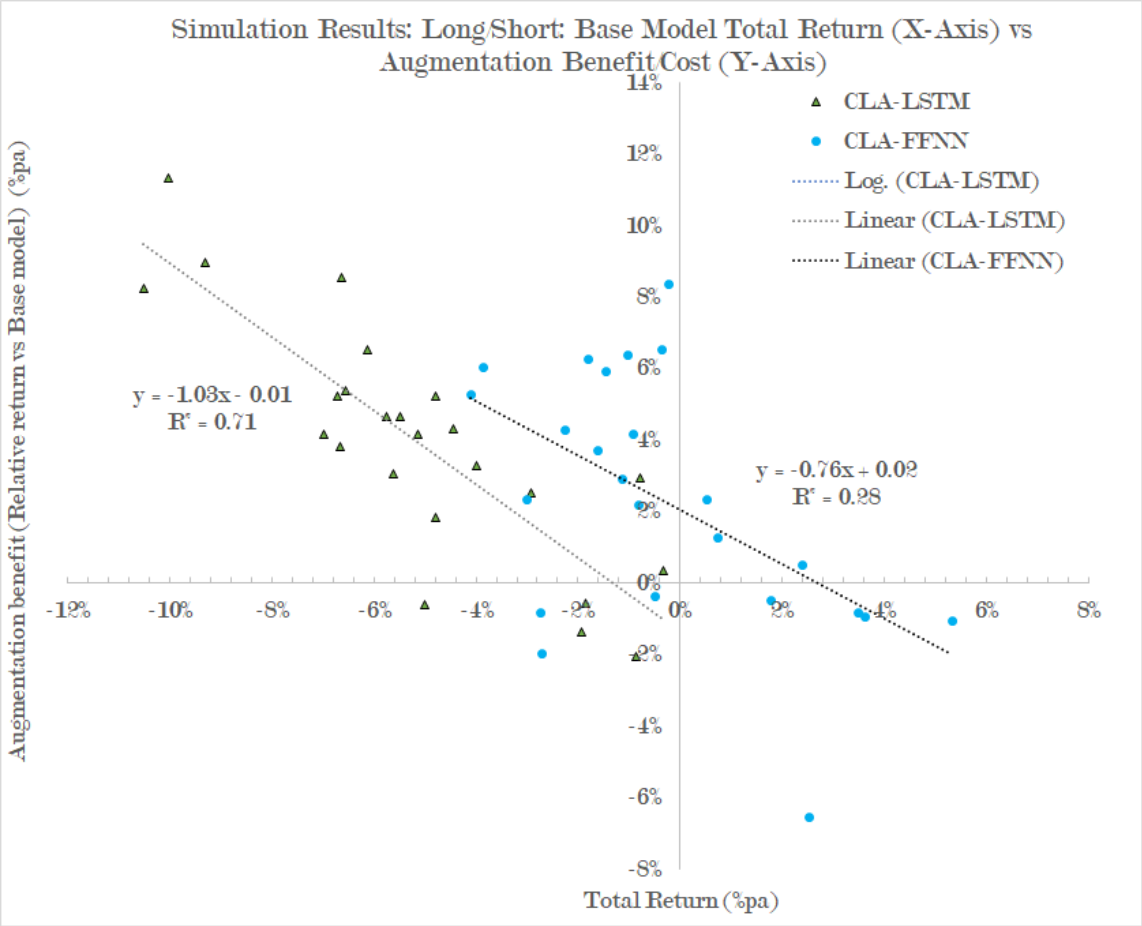
### 6.4.3 Discussion

We have empirically demonstrated that, when applied to a real-world financial task involving noisy time series, a sliding window learner (CLA-FFNN) is superior to a sequential learner (CLA-LSTM) in this application with this configuration. Testing different similarity approaches applied to a recall gate showed poor performance of the simple ED when compared to DTW. This strongly implies that the timing of data points is crucial in this task and likely in other real-world problems involving noisy time series. Simulation tests also showed that the AE distance is a good alternative to DTW. These results imply that AE dimensionality reduction and generalisation (using ReLU in this case) are almost equivalent to DTW-driven memory recall. warp-AE was proposed to benefit from both the generalisation of AEs and the phase invariance of DTW, an approach that produced the strongest investment performance and augmentation benefit of the similarity approaches that were tested. The analysis of the results using the augmentation-slope statistical test framework introduced in this study demonstrated a higher benefit for CLA-FFNN when compared to CLA-LSTMs; despite the seemingly higher augmentation benefit per se of LSTMs.

In summary, the most successful CL choices in these tests were found to be the sliding window CLA-FFNN learner combined with a recall gate using warp-AE similarity. These tests also affirm CLA as a real-world TCL approach with the flexibility to augment CL using different types of learners. As has been noted, this distance variant also came with a substantial resource economy when compared to

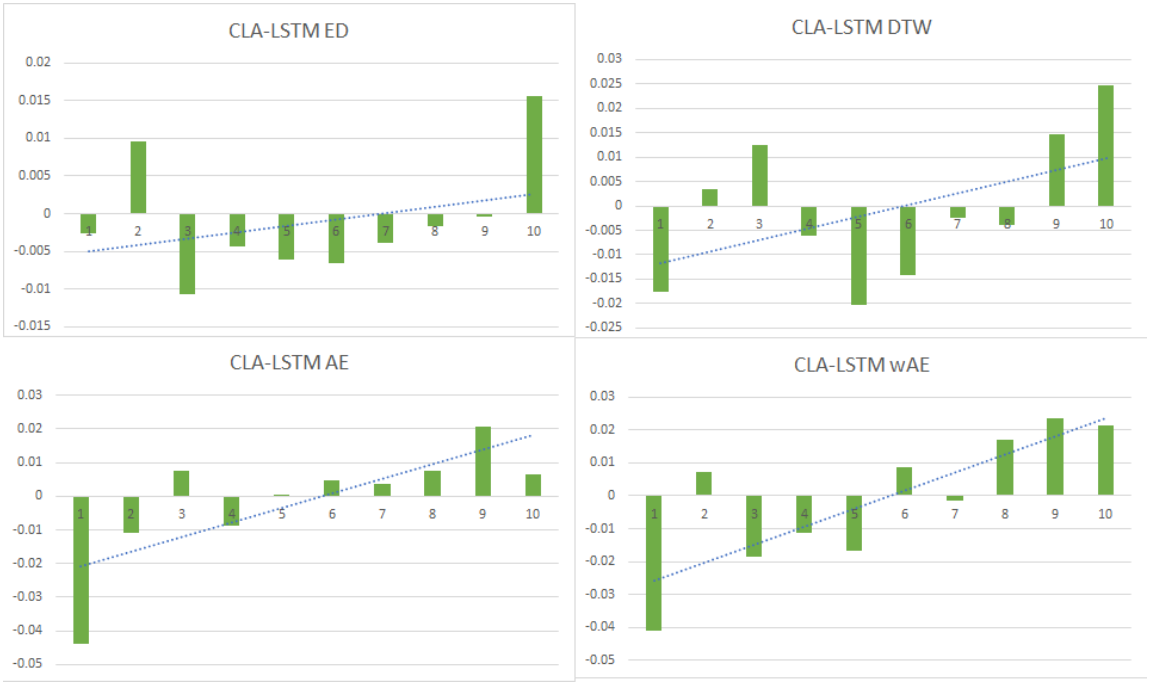
ED and DTW.

Figure 6.8: Augmentation Slopes



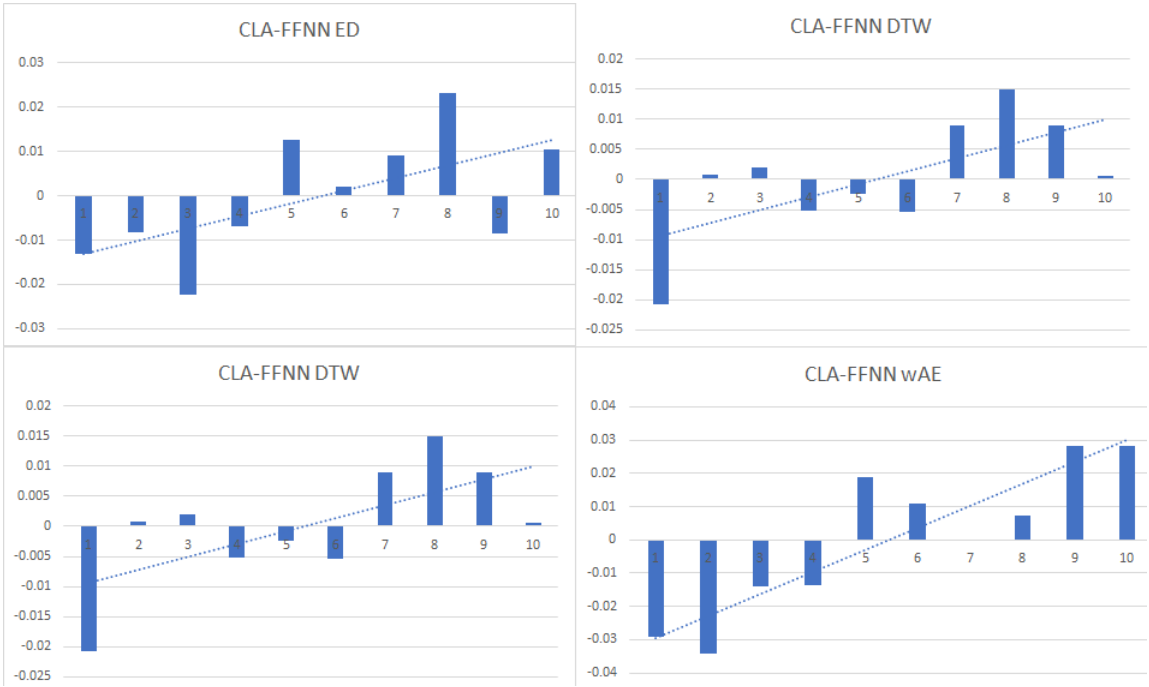

Note: LSTM vs FFNN, long/short tests: Plot of all simulations for each test. The total return of the approach ($x$-axis) against the augmentation benefit (relative return versus base model, $y$-axis). *Augmentation slopes* are shown as simple linear lines of best fit.

Figure 6.9: Monotonicity of Augmentation: CLA-LSTM Median Test Deciles



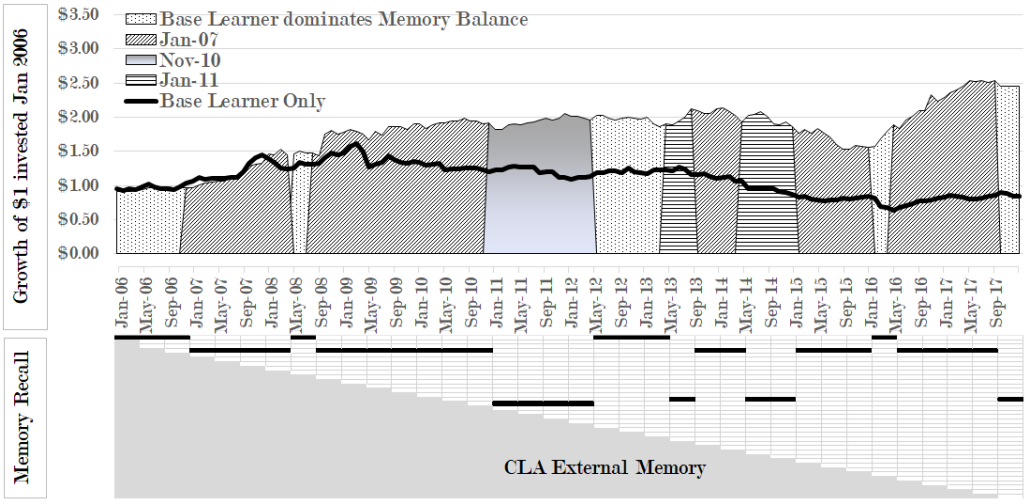

Note: Augmentation benefit monotonicity is noted in all distance measures by a positive slope coefficient: $RR_p$ of each decile portfolio of CLA-LSTM, annualised over the study term. Decile 1 relates to a portfolio of stocks in the lowest 10% returns forecasted by CLA-LSTM at each rebalance date, simulated as described.

Figure 6.10: Monotonicity of Augmentation: CLA-FFNN Median Test Deciles




Note: Augmentation benefit monotonicity is noted in all distance measures by a positive slope coefficient: $RR_p$ of each decile portfolio of CLA-FFNN, annualised over the study term. Decile 1 refers to a portfolio of stocks in the lowest 10% returns forecasted by CLA-FFNN at each rebalance date, simulated as described.

Figure 6.11: Memory Dynamics: Explainable Memory Addressing




Note: Explainable memory: Simple diagnostics reveal the recall of learner memories. Top: Investment returns of an unaugmented LSTM learner (line) shown next to CLA returns attributed by the highest weighted recalled memory (area). Lower chart: Representation of the CLA memory structure where each row in the expanding triangle represents a potential learner memory. The highest weighted recalled memories are highlighted.

# Chapter 7

# Conclusions

This thesis has addressed the principle hypothesis underlying TCL that *a real-world problem based on a state-varying time series with a large cross-sectional component can be addressed in a sequential, open-world fashion that addresses CF*. This has been achieved by the development of CLA and its successful application to complex real-world financial management tasks.

The secondary hypothesis was also addressed *by remembering models of temporal states, recalling these models in future periods when the input space is similar to an associated past state*. This has been achieved through the development of CLA's remember and recall gates.

Four key research questions underlying the motivations for TCL were also addressed:

1. Open-world TCL: CLA has been shown to be able to define state-based memories without prior definition.

2. Remembering using residual change: Residual change has been implemented as a threshold that can be learned by the CLA framework over time.

3. Memory recall using similarity: Different similarity approaches have been shown to have varying influence on the augmentation benefit from the CLA framework. It has been shown that similarity measures based on AE and warp-AE produced the most compelling augmentation in the CLA approach, as tested here.

4. Temporal state-based memory addressing: The combination of CLA's recall and remember gates have been shown to be able to define and address temporal states and to apply these states effectively in future periods. The strong

implication of this is that past states approximately repeat and that knowledge of these can be stored and used in the future.

Section 7.1 makes summarising and concluding remarks about CLA, empirical testing, and potential interpretability. Section 7.2 summarises the contributions of this work, and Section 7.3 explores possible future areas of research for TCL.

## 7.1 Continual Learning Augmentation

The CLA framework has allowed different elements of CL, ML, time series research, and CDA to be combined and, in some cases, repurposed to create a TCL memory-augmentation approach that has been shown to improve the performance of time series-based learners in specific real-world financial tasks. Testing shows that, when applied to complex real-world financial management tasks, CLA can significantly augment the performance of base learners, such as LSTM, FFNN, and OLS regression. The dynamics of augmentation are also studied using various statistical tests introduced to the CL context, including *augmentation slope*.

The CLA framework has been shown to be capable of using a remember gate in defining and accumulating knowledge of temporally changing time series states and is capable of recalling this knowledge using a recall gate when these past states appear to approximately reoccur. This has been achieved in a potentially explainable way. Moreover, CLA is different from traditional CL approaches that tend to make limiting assumptions of easy task delineation, dependencies, and more. Traditional CL approaches have also been limited by development and testing in relation to overly simplistic datasets, which is directly addressed in this research using complex, noisy data in a real-world application.

In addition, CLA also extends CDA approaches with distinct memory concepts and state-based addressing in several ways, including the use of an explainable memory addressing visualisation. Further, CLA can significantly augment LSTMs, which indicates that, in certain cases, the explicit long-term memory structure of CLA can contribute to the implicit recurrent memory of LSTMs. The implications of these findings are likely to be significant.

Certain configurations of CLA gates are less effective than others. This has been expressed as both outright financial performance and the benefit of augmentation. The main findings are described below.

**Remember gate** The *residual change*, an old idea, can be repurposed for a TCL memory recall gate to separate time series into explicit and different state-based

memories analogous to tasks in conventional CL and, in this regard, is different from conventional CDA memory. In turn, this can allow a system to learn and remember states of a time series in an open-world fashion. These ideas are also found to work effectively in a specific real-world context during testing. It is possible to sequentially learn the change threshold, the critical value $J_Crit$. The development of old ideas to facilitate a TCL remember gate is likely to be a significant development in time series modelling and, more importantly, in understanding how TCL might be applied to real-world situations.

**Recall gate** The recall gate for CLA was researched and tested using different forms of similarity. Different similarity approaches had different effects on the performance of the recall gate. Testing of different similarity approaches that were applied to this recall gate showed poor performance for the simple ED when compared to DTW. This strongly implies that the timing of data points is important in the tested tasks and, likely, in other real-world problems involving noisy time series.

Most interesting was that the AE distance was generally more effective in both synthetic and real-world testing than DTW, which relies on a more resource-expensive memory, containing training examples. These results confirm that AEs used in TCL can be effective, as indicated in the CL research. The AE dimensionality reduction and generalisation, which were implemented using ReLU with sparse regularisation, are shown in testing to improve the DTW-driven TCL memory recall. warp-AE was proposed to benefit from both the generalisation of AEs and the phase invariance of DTW, an approach that performed well in testing.

In summary, testing established that, in a specific real-world context, the DTW distance is more effective than the simple ED. Additionally, the AE distance, which uses a latent representation of a state, is generally more effective in both synthetic and real-world testing than DTW. This indicated that the sparse form of AE representations was more effective in the tested contexts than the more resource-intensive use of training examples in DTW. Moreover, warp-AE, which applies a DTW filter to an AE reconstruction, improves performance further in both synthetic (Figure 4.1.1) and real-world testing (Figure 6), indicating that a benefit exists from the phase invariance of distance calculations in the context of the testing.

**Augmentation dynamics** A simple empirical approach for examining the cost/benefit of TCL results was also introduced, *augmentation slope*. This made it possible to compare the *augmentation slopes* of different CLA configurations and base learners to better understand the dynamics of augmentation. This confirmed a higher augmentation benefit for FFNN when compared to LSTM base learners, despite that LSTMs

seem to have a higher augmentation benefit per se. The degree of augmentation possible generally increases with the level of base model error. This was shown by the negative *augmentation-slope* coefficients in testing.

**Accuracy of outcomes** If CLA were exploited in practice for an investment approach, indications are that CLA would provide a significant advantage to investment strategy returns. Moreover, CLA produces positive, statistically significant outcomes for certain base learners in the tested real-world financial management tasks. Both long and long/short tests showed positive and statistically significant augmentation benefits for linear, FFNN, and LSTM base learners. Most interesting is perhaps the statistically significant augmentation benefit over LSTMs. However, the outright returns of CLA-LSTMs were poor when compared to CLA-FFNN. Of the different similarity weighting schemes in the recall gate, the weighting of learner memories using *similarity* produces stronger results than simply picking the *best* learner memory.

In summary, the most successful TCL choice was a sliding window FFNN learner combined with a recall gate using AE-based similarity. These real-world empirical tests bare out the more stylised, synthetic-based empirical testing that was also carried out on time series similarity approaches, which also found AE-based similarity to be superior to the ED and DTW distance on longitudinal and cross-sectional similarity testing. Testing in this thesis also affirms CLA as a real-world applicable approach with the flexibility to augment different types of learners in CL.

**Explainable of memory use** The memory structure of CLA and the visualisations introduced here can potentially be interpreted by domain experts in terms of which past state is relevant to forecasting in the current state. This allows objective comparisons to be made between relevant past states and the current state and allows for a better understanding of the characteristics of the current state in the context of similar past states. This information is expected to be helpful to provide insight to domain experts to guide decision-making.

## 7.2 Contributions

In this research, it has been shown that a TCL framework, CLA, can acquire knowledge sequentially, related to time series with a large cross-sectional component. This has been shown in experimental testing using a real-world financial problem: international and emerging market equity investing. On the broader question of CF, the CLA framework has, to some extent, addressed this, but CF remains an open question.

The remember gate for CLA was developed, and residual change thresholds can be learned over time to define changing temporal states. This change threshold is essentially the stability-plasticity trade-off for the CLA framework. These change points are effective cues for remembering learner parameters with a contextual reference. This memory information is useful in identifying an approximate recurrence of the memorised state and for recalling the appropriate learner parameters from an external memory structure.

A recall gate based on similarity was also researched and was effective in testing. Different similarity approaches were tested to compare the contextual reference in each memory with the current input space. Where the similarity was high, a memory would be chosen and its learner parameters were applied to the current input space. Different distance approaches have different levels of success in synthetic testing and in application to a real-world task.

Different memory balancing schemes were also tested to either select the single most similar memory, to equally weight all memories in the external memory, or to weight memories by similarity. In addition, CLA is configured to produce effective results in testing. Some configurations were relatively effective, and some less so.

Furthermore, CLA can successfully augment several types of time series learners in the tests. Based on the introduced CLA framework and the testing, the posited hypothesis cannot be rejected. The four research questions have also been successfully addressed with specific major contributions as follows:

1. Time series memory structure: The CLA approach has been researched to build, maintain, and use a state-based and addressed memory structure. This is slightly different from the reviewed CL approaches, in that it is applied to time series states rather than tasks. It is also an extension of the minimalist memory concepts of CDA, as TCL memories are related to states and can be stored indefinitely.

2. Simple learner memory augmentation: Recurrent FFNN and OLS regression learners can be memory-augmented using a generalised deep architecture.

3. Recall gate: Data-mining approaches for time series similarity are repurposed for use in a TCL memory gate, an approach not known to be adopted in the CL literature. This is used to drive pattern recognition in multivariate time series input data to propose memories to recall. This is believed to be a novel recall gate.

4. Remember gate based on multivariate residual change: The use of *residual change* is well known in the drift adaptation and concept change literature but

may be a novel approach for memory gating in the explicit, state-based external memory structure researched in this study.

Minor contributions of this work follow:

1. Open-world learning in the real-world: Real-world applied TCL approaches are reported in this thesis. These approaches were tested on real-world temporal data problems and were effective in the tested contexts. A review of the literature indicates that this is one of only a few time series applied to open-world CL approaches. While many CDA approaches operate on streams of data, they do not have task-based learners.

2. Potentially Interpretable memory: It is possible to extract memory remember and recall information from the memory structure introduced in this work. A visualisation to explain which memory, did what and when is developed to allow interpretation by domain experts.

## 7.3 Future Work

Our results indicate that CLA may be effectively applied to other problems on noisy and non-stationary time series problems inside and outside of the finance domain. While the framework presented in this thesis is directly applicable to quantitative investment, CLA is also intended for application to other fields.

### 7.3.1 Residual Change Distribution

It is also noted than the nature of absolute error change, as a memory concept, could hold more benefits for memory augmentation or model selection, as has been investigated in the CD literature where change in the absolute error *distribution* could be used in TCL to better identify changing states and to better learn more appropriate parameterisations.

### 7.3.2 Abrupt Change versus Gradual Change

While driving the remember-gate using abrupt change has been demonstrably successful, using change-points, it is also possible that in certain datasets or applications gradual change might be more important. Future work might investigate gradual change, such as is proposed in [229], as an additional driver of remember-cues.

### 7.3.3 Applying Domain Knowledge Proactively

It might be possible to inject synthetic memories of specific anticipated events, and thus incorporate the specific domain knowledge of experts in the memory structure. It is envisaged that this could be achieved by using past events as a basis to model expected outcomes. For example, should an event such as the global financial crisis occur again, perhaps banks would be less likely to suffer re-capitalisations owing to changes in policy makers decisions that would be exogenous to the base learner used. In this case a more benign outcome could be incorporated into the base learner parameters for the banking sector should a similar event appear to be occurring again.

### 7.3.4 Abrupt Change versus Gradual Change

While driving the remember-gate using abrupt change has been demonstrably successful, using change-points, it is also possible that in certain datasets or applications gradual change might be more important. Future work might investigate gradual change, such as is proposed in [229], as an additional driver of remember-cues.

### 7.3.5 Remember the Best of a State

Further tests have been conducted on change, and although the CLA implementation developed in this thesis uses abrupt change as a delineator of states and a cue to remember, in the latest testing, remember cues based on a fall in absolute error might prove more effective. This can be considered as remembering the "best learner" parameters within a state (ie between two change points). Univariate testing showed that saving learner parameters when the absolute error fell below a certain critical level was more effective than remembering when the absolute error rose over a certain level. It might also be the case that an upper and lower change threshold might be most effective (i.e. identifying an uncommonly poor or accurate model of a state and responding to either event). More testing should be conducted to determine whether this change was made or whether it would make the results any less interpretable in the context of changing states in the dataset.

### 7.3.6 Attention

Attention mechanisms could be added to the CLA framework to improve the performance of the recall-gate and remember-gate, now important research questions have been addressed in this thesis relating to TCL. Firstly, the AE similarity function in the recall-gate, where attention could be applied to the input data to optimise the

similarity calculation for each memory. Secondly, attention might also be used in the balancing function of the recall-gate.

It is envisaged that, instead of simply balancing learner memories using a closed-form similarity function, it may be possible to weight inputs to a similarity function and apply attention. This would essentially learn attention for a similarity function over time for each learner memory, which is likely to improve the performance of the remember gate in the CLA approach. Other forms of similarity might also be tested in this framework, such as CNNs and restricted Boltzmann machines to further assess reconstruction-based similarity.

In addition to a more effective remember gate, this attention mechanism could be used to forget memories more efficiently. If a similarity function resulted in down-weighting inputs to the extent that it never applied a high weight in the balancing function of CLA, this implies that the memory should be forgotten. It is likely that this additional forget gate could also be implemented using a forget threshold to be learned, which might look very similar to the process of learning $J_{Crit}$.

Including attention mechanisms would involve a higher degree of parameterisation of the approach, a disadvantage, but it might be possible to robustly improve modelling outcomes. This would technically be a straightforward extension of the approach but would increase the complexity, which, in the real-world, would require considerable work to mitigate, perhaps centred on regularisation.

### 7.3.7 The Continual Learning Augmentation Unit

The most interesting area for future development of the CLA approach is in the construction of a generic *CLA unit* for use in existing ML architectures. It is envisaged that a CLA unit would *wrap* other units or elements of existing ML architectures in a similar way as shown in this study where base learners, such as linear, FFNN, and LSTM have been augmented by CL. The same principle might be applied to layers or larger architectural elements of an existing ML system, which is otherwise exposed to CF. The CLA unit would accept inputs from upstream layers (or units) and create memories based on the backpass of downstream errors into the CLA-wrapped element of the architecture. In this way, it could support CL in specific areas of an existing ML approach or in the overall approach itself, as has been demonstrated is possible with LSTM, FFNN, and linear learners in this thesis. It would also be possible to use CLA to trigger a retraining event of the ML architecture, should change be detected. Thus, a traditional ML approach could be made state/task aware and therefore could support many tasks in an open-world fashion. This is envisaged as the next area of development of the CLA framework and would allow more extensive and generic

testing in areas of any type of sequential learning.

# Chapter 8

# Bibliography


[1] Hanady Abdulsalam, David B Skillicorn, and Patrick Martin. "Classification using streaming random forests". In: *IEEE Transactions on knowledge and Data Engineering* 23.1 (2010), pp. 22–36.

[2] David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. "A Learning Algorithm for Boltzmann Machines*". In: *Cognitive Science* 9.1 (1985), pp. 147–169. DOI: 10.1207/s15516709cog0901_7. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1207/s15516709cog0901_7. URL: https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog0901_7.

[3] Ratnadip Adhikari and R. K. Agrawal. "An Introductory Study on Time Series Modeling and Forecasting". In: *CoRR* abs/1302.6613 (2013). arXiv: 1302.6613. URL: http://arxiv.org/abs/1302.6613.

[4] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. "Time-series Clustering - A Decade Review". In: *Inf. Syst.* 53.C (Oct. 2015), pp. 16–38. ISSN: 0306-4379. DOI: 10.1016/j.is.2015.04.007. URL: http://dx.doi.org/10.1016/j.is.2015.04.007.

[5] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. "Expert Gate: Lifelong Learning with a Network of Experts". In: *CoRR* abs/1611.06194 (2017). arXiv: 1611.06194. URL: http://arxiv.org/abs/1611.06194.

[6] Miquel N. Alonso, Gilberto Batres-Estrada, and Aymeric Moulin. "Deep Learning in Finance: Prediction of Stock Returns with Long Short-Term Memory Networks". In: *Big Data and Machine Learning in Quantitative Investment*. John Wiley & Sons, Ltd, 2018. Chap. 13, pp. 251–277. ISBN: 9781119522225. DOI: 10.1002/9781119522225.ch13. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119522225.ch13. URL:

https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119522225.ch13.

[7] Naomi S. Altman. "An introduction to kernel and nearest-neighbor nonparametric regression". English (US). In: *American Statistician* 46.3 (Jan. 1992), pp. 175–185. ISSN: 0003-1305. DOI: 10.1080/00031305.1992.10475879.

[8] Amirhossein Amiri and Saeed Allahyari. "Change Point Estimation Methods for Control Chart Postsignal Diagnostics: A Literature Review". In: *Quality and Reliability Engineering International* 28.7 (2012), pp. 673–685. DOI: 10.1002/qre.1266. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/qre.1266. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/qre.1266.

[9] Anthony Bagnall et al. "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances". In: *Data Mining and Knowledge Discovery* 31.3 (May 2017), pp. 606–660. ISSN: 1573-756X. DOI: 10.1007/s10618-016-0483-9. URL: https://doi.org/10.1007/s10618-016-0483-9.

[10] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2014. arXiv: 1409.0473 [cs.CL].

[11] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. 2018. arXiv: 1803.01271 [cs.LG].

[12] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". In: *CoRR* abs/1803.01271 (2018). arXiv: 1803.01271. URL: http://arxiv.org/abs/1803.01271.

[13] Wei Ning Bao, J. H. Yue, and Yulei Rao. "A deep learning framework for financial time series using stacked autoencoders and long-short term memory". In: *PloS one*. 2017.

[14] Ziv Bar-Joseph et al. "Continuous Representations of Time-Series Gene Expression Data". In: *Journal of Computational Biology* 10.3-4 (2003). PMID: 12935332, pp. 341–356. DOI: 10.1089/10665270360688057. eprint: https://doi.org/10.1089/10665270360688057. URL: https://doi.org/10.1089/10665270360688057.

[15] Peter L. Bartlett and Wolfgang Maass. "Vapnik-Chervonenkis dimension of neural nets". English. In: *The Handbook of Brain Theory and Neural Networks*. 2nd ed. MIT Press, 2003, pp. 1188–1192.

[16] Arslan Basharat and Mubarak Shah. "Time series prediction by chaotic modeling of nonlinear dynamical systems". In: *2009 IEEE 12th International Conference on Computer Vision* (2009), pp. 1941–1948.

[17] Gustavo E. A. P. A. Batista, Xiaoyue Wang, and Eamonn J. Keogh. "A Complexity-Invariant Distance Measure for Time Series". In: *SDM*. 2011.

[18] Y. Bengio, P. Simard, and P. Frasconi. "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks* 5.2 (June 1994), pp. 157–166. DOI: 10.1109/72.279181.

[19] George.E.P. Box and Gwilym M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden-Day, 1976.

[20] G.E.P. Box and G.M. Jenkins. *Time series analysis: forecasting and control*. Holden-Day series in time series analysis and digital processing. Holden-Day, 1976. ISBN: 9780816211043. URL: https://books.google.co.uk/books?id=1WVHAAAAMAAJ.

[21] James Bradbury et al. *Quasi-Recurrent Neural Networks*. 2016. arXiv: 1611.01576 [cs.NE].

[22] Evans J Brown R Durbin J. "Techniques for testing the constancy of regression relationships over time". In: *Journal of the Royal Statistical Society Series B (methodological)* 37.2 (1975), pp. 149–192.

[23] Mikel Canizo et al. "Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study". In: *Neurocomputing* 363 (2019), pp. 246 –260. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.07.034. URL: http://www.sciencedirect.com/science/article/pii/S0925231219309877.

[24] Jian Cao, Zhi Li, and Jian Li. "Financial time series forecasting model based on CEEMDAN and LSTM". In: *Physica A: Statistical Mechanics and its Applications* 519 (2019), pp. 127 –139. ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2018.11.061. URL: http://www.sciencedirect.com/science/article/pii/S0378437118314985.

[25] José Carmona-Cejudo et al. "GNUsmail: Open Framework for On-line Email Classification." In: Jan. 2010, pp. 1141–1142. DOI: 10.3233/978-1-60750-606-5-1141.

[26] Rich Caruana. "Multitask Learning". In: *Machine Learning* 28.1 (July 1997), pp. 41–75. ISSN: 1573-0565. DOI: 10.1023/A:1007379606734. URL: https://doi.org/10.1023/A:1007379606734.

[27] Sung-Hyuk Cha. "Comprehensive Survey on Distance/Similarity Measures Between Probability Density Functions". In: *Int. J. Math. Model. Meth. Appl. Sci.* 1 (Jan. 2007).

[28] Kin pong Chan and Ada Wai-Chee Fu. "Efficient time series matching by wavelets". In: *Proceedings 15th International Conference on Data Engineering (Cat. No.99CB36337)* (1999), pp. 126–133.

[29] Jinmiao Chen and Narendra S. Chaudhari. "Learning Long-Term Dependencies in Segmented Memory Recurrent Neural Networks". In: *Advances in Neural Networks – ISNN 2004*. Ed. by Fu-Liang Yin, Jun Wang, and Chengan Guo. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 362–369. ISBN: 978-3-540-28647-9.

[30] Yueguo Chen et al. "SpADe: On Shape-based Pattern Detection in Streaming Time Series". In: *2007 IEEE 23rd International Conference on Data Engineering* (2007), pp. 786–795.

[31] Zhiyuan Chen and Bing Liu. "Lifelong Machine Learning Second Edition". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning*. 2018.

[32] Zhiyuan Chen and Bing Liu. "Topic Modeling Using Topics from Many Domains, Lifelong Learning and Big Data". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*. ICML'14. Beijing, China: JMLR.org, 2014, pp. II–703–II–711. URL: http://dl.acm.org/citation.cfm?id=3044805.3044971.

[33] Jianpeng Cheng, Li Dong, and Mirella Lapata. *Long Short-Term Memory-Networks for Machine Reading*. 2016. arXiv: 1601.06733 [cs.CL].

[34] Kyunghyun Cho et al. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. 2014. arXiv: 1409.1259 [cs.CL].

[35] KyungHyun Cho et al. "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches". In: *CoRR* abs/1409.1259 (2014). arXiv: 1409.1259. URL: http://arxiv.org/abs/1409.1259.

[36] Junyoung Chung et al. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *CoRR* abs/1412.3555 (2014). arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555.

[37] David A. Cieslak and Nitesh V. Chawla. "A framework for monitoring classifiers' performance: when and why failure occurs?" In: *Knowledge and Information Systems* 18.1 (Jan. 2009), pp. 83–108. ISSN: 0219-3116. DOI: 10.1007/s10115-008-0139-1. URL: https://doi.org/10.1007/s10115-008-0139-1.

[38] Dan C. Ciresan, Ueli Meier, and Jurgen Schmidhuber. "Multi-column Deep Neural Networks for Image Classification". In: *CoRR* abs/1202.2745 (2012). arXiv: 1202.2745. URL: http://arxiv.org/abs/1202.2745.

[39] Ronan Collobert and Jason Weston. "A unified architecture for natural language processing: Deep neural networks with multitask learning". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 160–167.

[40] Charles J. Corrado and Terry L. Zivney. "The Specification and Power of the Sign Test in Event Study Hypothesis Tests Using Daily Stock Returns". In: *Journal of Financial and Quantitative Analysis* 27.3 (1992), 465–478. DOI: 10.2307/2331331.

[41] RODNEY M. J. COTTERILL. "A review of: "Modelling Brain Function: The World of Attractor Neural Networks" Daniel J. Amit Cambridge: Cambridge University Press, 1989 ISBN 0-521-36100-1, xvii + 226pp., £25.00". In: *Connection Science* 3.1 (1991), pp. 94–95. DOI: 10.1080/09540099108946576. eprint: https://doi.org/10.1080/09540099108946576. URL: https://doi.org/10.1080/09540099108946576.

[42] Koby Crammer et al. "Online Passive-Aggressive Algorithms". In: *J. Mach. Learn. Res.* 7 (Dec. 2006), pp. 551–585. ISSN: 1532-4435. URL: http://dl.acm.org/citation.cfm?id=1248547.1248566.

[43] Fernando Crespo and Richard Weber. "A methodology for dynamic data mining based on fuzzy clustering". In: *Fuzzy Sets and Systems* 150.2 (2005), pp. 267 –284. ISSN: 0165-0114. DOI: https://doi.org/10.1016/j.fss.2004.03.028. URL: http://www.sciencedirect.com/science/article/pii/S016501140400140X.

[44] Zhicheng Cui, Wenlin Chen, and Yixin Chen. *Multi-Scale Convolutional Neural Networks for Time Series Classification*. 2016. arXiv: 1603.06995 [cs.CV].

[45] Mannila H Renganathan G Smyth P Das G Lin K. "Rule discovery from time series". In: *Proceedings of the 4th Int'l Conference on Knowledge Discovery and Data Mining* (), pp. 16–22.

[46] Grégoire Mesnil Yann Dauphin et al. "Unsupervised and Transfer Learning Challenge: a Deep Learning Approach". In: *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*. Ed. by Isabelle Guyon et al. Vol. 27. Proceedings of Machine Learning Research. PMLR, July 2012, pp. 97–110. URL: http://proceedings.mlr.press/v27/mesnil12a.html.

[47] Matthias De Lange et al. "Continual learning: A comparative study on how to defy forgetting in classification tasks". In: *arXiv preprint arXiv:1909.08383* (2019).

[48] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. arXiv: 1810.04805 [cs.CL].

[49] Thomas G. Dietterich. "Machine Learning for Sequential Data: A Review". In: *Structural, Syntactic, and Statistical Pattern Recognition*. Ed. by Terry Caelli et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 15–30. ISBN: 978-3-540-70659-5.

[50] G. Ditzler and R. Polikar. "Hellinger distance based drift detection for nonstationary environments". In: *2011 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*. Apr. 2011, pp. 41–48. DOI: 10.1109/CIDUE.2011.5948491.

[51] Eizaburo Doi, Doru C. Balcan, and Michael S. Lewicki. "A Theoretical Analysis of Robust Coding over Noisy Overcomplete Channels". In: *Advances in Neural Information Processing Systems 18*. Ed. by Y. Weiss, B. Schölkopf, and J. C. Platt. MIT Press, 2006, pp. 307–314. URL: http://papers.nips.cc/paper/2867-a-theoretical-analysis-of-robust-coding-over-noisy-overcomplete-channels.pdf.

[52] Pedro Domingos and Geoff Hulten. "Mining high-speed data streams". In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2000, pp. 71–80.

[53] Saso Džeroski and Bernard Ženko. "Is Combining Classifiers with Stacking Better than Selecting the Best One?" In: *Machine Learning* 54.3 (Mar. 2004), pp. 255–273. ISSN: 1573-0565. DOI: 10.1023/B:MACH.0000015881.36452.6e. URL: https://doi.org/10.1023/B:MACH.0000015881.36452.6e.

[54] Page E. "On problems in which a change in a parameter occurs at an unknown point". In: *Biometrika* 44.1/2 (1957), pp. 248–252.

[55] Jeffrey L. Elman. "Finding Structure in Time". In: *Cognitive Science* 14.2 (1990), pp. 179–211. DOI: 10.1207/s15516709cog1402_1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1207/s15516709cog1402_1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog1402_1.

[56] Smith A Engle R. "Stochastic Permanent Breaks". In: *The Review of Economics and Statistics* 81.4 (1999), pp. 553–574.

[57] P. A. Estevez and R. Nakano. "Hierarchical mixture of experts and Max-Min propagation neural networks". In: *Proceedings of ICNN'95 - International Conference on Neural Networks*. Vol. 1. Nov. 1995, 651–656 vol.1. DOI: 10.1109/ICNN.1995.488257.

[58] F.J. Fabozzi, S.M. Focardi, and P.N. Kolm. *Quantitative Equity Investing: Techniques and Strategies*. Frank J. Fabozzi Series. Wiley, 2010. ISBN: 9780470262474. URL: https://books.google.se/books?id=l2LCDAEACAAJ.

[59] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. "Fast Subsequence Matching in Time-series Databases". In: *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*. SIGMOD '94. Minneapolis, Minnesota, USA: ACM, 1994, pp. 419–429. ISBN: 0-89791-639-5. DOI: 10.1145/191839.191925. URL: http://doi.acm.org/10.1145/191839.191925.

[60] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. "Fast Subsequence Matching in Time-Series Databases". In: *ACM SIGMOD Record* 23 (June 2000). DOI: 10.1145/191839.191925.

[61] Eugene F. Fama and Kenneth R. French. "Common Risk Factors in the Returns On Stocks And Bonds". In: *Journal of Financial Economics* 33 (1993), pp. 3–56.

[62] Sebastian Farquhar and Yarin Gal. "Towards robust evaluations of continual learning". In: *arXiv preprint arXiv:1805.09733* (2018).

[63] Hassan Ismail Fawaz et al. "Deep learning for time series classification: a review". In: *CoRR* abs/1809.04356 (2018). arXiv: 1809.04356. URL: http://arxiv.org/abs/1809.04356.

[64] Chrisantha Fernando et al. "PathNet: Evolution Channels Gradient Descent in Super Neural Networks". In: *CoRR* abs/1701.08734 (2017). arXiv: 1701.08734. URL: http://arxiv.org/abs/1701.08734.

[65] Thomas Fischer and Christopher Krauss. *Deep learning with long short-term memory networks for financial market predictions*. FAU Discussion Papers in Economics 11/2017. Friedrich-Alexander University Erlangen-Nuremberg, Institute for Economics, 2017. URL: https://ideas.repec.org/p/zbw/iwqwdp/112017.html.

[66] Jörg Franke, Jan Niehues, and Alex Waibel. "Robust and Scalable Differentiable Neural Computer for Question Answering". In: *CoRR* abs/1807.02658 (2018). arXiv: 1807.02658. URL: http://arxiv.org/abs/1807.02658.

[67] Robert M. French. "Catastrophic forgetting in connectionist networks". In: *Trends in Cognitive Sciences* 3.4 (1999), pp. 128 –135. ISSN: 1364-6613. DOI: https://doi.org/10.1016/S1364-6613(99)01294-2. URL: http://www.sciencedirect.com/science/article/pii/S1364661399012942.

[68] T. Fu, Y. Hung, and F. Chung. "Improvement algorithms of perceptually important point identification for time series data mining". In: *2017 IEEE 4th International Conference on Soft Computing Machine Intelligence (ISCMI)*. Nov. 2017, pp. 11–15. DOI: 10.1109/ISCMI.2017.8279589.

[69] Tak-Chung Fu et al. "Pattern Discovery from Stock Time Series Using Self-Organizing Maps". In: *7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2001.

[70] João Gama et al. "A Survey on Concept Drift Adaptation". In: *ACM Comput. Surv.* 46.4 (Mar. 2014), 44:1–44:37. ISSN: 0360-0300. DOI: 10.1145/2523813. URL: http://doi.acm.org/10.1145/2523813.

[71] João Gama et al. "Learning with Drift Detection". In: *Advances in Artificial Intelligence – SBIA 2004*. Ed. by Ana L. C. Bazzan and Sofiane Labidi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 286–295. ISBN: 978-3-540-28645-5.

[72] M. Gavrilov et al. "Mining the stock market: Which measure is best". In: *Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining*. Citeseer. 2000. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.126.8075&rep=rep1&type=pdf.

[73] Johannes Gehrke et al. "BOAT&Mdash;Optimistic Decision Tree Construction". In: *SIGMOD Rec.* 28.2 (June 1999), pp. 169–180. ISSN: 0163-5808. DOI: 10.1145/304181.304197. URL: http://doi.acm.org/10.1145/304181.304197.

[74] Xin Geng and Kate Smith-Miles. "Incremental Learning". In: *Encyclopedia of Biometrics*. Ed. by Stan Z. Li and Anil Jain. Boston, MA: Springer US, 2009, pp. 731–735. ISBN: 978-0-387-73003-5. DOI: 10.1007/978-0-387-73003-5_304. URL: https://doi.org/10.1007/978-0-387-73003-5_304.

[75] Mathieu Germain et al. "MADE: masked autoencoder for distribution estimation". In: *Proceedings of the 32nd International Conference on Machine Learning*. Vol. 37. JMLR: W&CP. 2015, pp. 881–889.

[76] Felix A. Gers, Douglas Eck, and Jürgen Schmidhuber. "Applying LSTM to Time Series Predictable through Time-Window Approaches". In: *Artificial Neural Networks — ICANN 2001*. Ed. by Georg Dorffner, Horst Bischof, and Kurt Hornik. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 669–676. ISBN: 978-3-540-44668-2.

[77] John Geweke and Michael Keane. "Smoothly mixing regressions". In: *Journal of Econometrics* 138.1 (2007). 50th Anniversary Econometric Institute, pp. 252 –290. ISSN: 0304-4076. DOI: https://doi.org/10.1016/j.jeconom.2006.05.022. URL: http://www.sciencedirect.com/science/article/pii/S0304407606000959.

[78] Zoubin Ghahramani and Sam T. Roweis. "Learning Nonlinear Dynamical Systems Using an EM Algorithm". In: *Advances in Neural Information Processing Systems 11*. Ed. by M. J. Kearns, S. A. Solla, and D. A. Cohn. MIT Press, 1999, pp. 431–437. URL: http://papers.nips.cc/paper/1594-learning-nonlinear-dynamical-systems-using-an-em-algorithm.pdf.

[79] Ross B. Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *CoRR* abs/1311.2524 (2013). arXiv: 1311.2524. URL: http://arxiv.org/abs/1311.2524.

[80] Milan Gocic and Slavisa Trajkovic. "Analysis of changes in meteorological variables using Mann-Kendall and Sen's slope estimator statistical tests in Serbia". In: *Global and Planetary Change* 100 (2013), pp. 172 –182. ISSN: 0921-8181. DOI: https://doi.org/10.1016/j.gloplacha.2012.10.014. URL: http://www.sciencedirect.com/science/article/pii/S0921818112002032.

[81] João Bártolo Gomes, Ernestina Menasalvas, and Pedro A. C. Sousa. "CALDS: Context-aware Learning from Data Streams". In: *Proceedings of the First International Workshop on Novel Data Stream Pattern Mining Techniques*. StreamKDD '10. Washington, D.C.: ACM, 2010, pp. 16–24. ISBN: 978-1-4503-0226-5. DOI: 10.1145/1833280.1833283. URL: http://doi.acm.org/10.1145/1833280.1833283.

[82] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *arXiv e-prints*, arXiv:1412.6572 (Dec. 2014), arXiv:1412.6572. arXiv: 1412.6572 [stat.ML].

[83] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *CoRR* abs/1412.6572 (2014).

[84] Ian J. Goodfellow et al. "Generative Adversarial Networks". In: *ArXiv* abs/1406.2661 (2014).

[85] Alex Graves, Greg Wayne, and Ivo Danihelka. "Neural Turing Machines". In: *CoRR* abs/1410.5401 (2014). arXiv: 1410.5401. URL: http://arxiv.org/abs/1410.5401.

[86] Alex Graves et al. "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks". In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML '06. Pittsburgh, Pennsylvania, USA: ACM, 2006, pp. 369–376. ISBN: 1-59593-383-2. DOI: 10.1145/1143844.1143891. URL: http://doi.acm.org/10.1145/1143844.1143891.

[87] Alex Graves et al. "Hybrid computing using a neural network with dynamic external memory". In: *Nature* 538.7626 (Oct. 2016), pp. 471–476. ISSN: 00280836. URL: http://dx.doi.org/10.1038/nature20101.

[88] Kesten C. Green and J. Scott Armstrong. "Simple versus complex forecasting: The evidence". In: *Journal of Business Research* 68.8 (2015). Special Issue on Simple Versus Complex Forecasting, pp. 1678 –1685. ISSN: 0148-2963. DOI: https://doi.org/10.1016/j.jbusres.2015.03.026. URL: http://www.sciencedirect.com/science/article/pii/S014829631500140X.

[89] Tian Guo, Tao Lin, and Nino Antulov-Fantulin. "Exploring Interpretable LSTM Neural Networks over Multi-Variable Data". In: *CoRR* abs/1905.12034 (2019). arXiv: 1905.12034. URL: http://arxiv.org/abs/1905.12034.

[90] Sherri K. Harms, Jitender S. Deogun, and Tsegaye Tadesse. "Discovering Sequential Association Rules with Constraints and Time Lags in Multiple Sequences". In: *ISMIS*. 2002.

[91] "Hebb, D. O. Organization of behavior. New York: Wiley, 1949, pp. 335, $4.00". In: *Journal of Clinical Psychology* 6.3 (1950), pp. 307–307. DOI: 10.1002/1097-4679(195007)6:3<307::AID-JCLP2270060338>3.0.CO;2-K. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/1097-4679\%28195007\%296\%3A3\%3C307\%3A\%3AAID-JCLP2270060338\%3E3.0.CO\%3B2-K.

URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/1097-4679\%28195007\%296\%3A3\%3C307\%3A\%3AAID-JCLP2270060338\%3E3.0.CO\%3B2-K.

[92] G. E. Hinton. "Connectionist Learning Procedures". In: *Artif. Intell.* 40.1-3 (Sept. 1989), pp. 185–234. ISSN: 0004-3702. DOI: 10.1016/0004-3702(89)90049-0. URL: http://dx.doi.org/10.1016/0004-3702(89)90049-0.

[93] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. "Distilling the Knowledge in a Neural Network". In: *NIPS Deep Learning and Representation Learning Workshop*. 2015. URL: http://arxiv.org/abs/1503.02531.

[94] S. Hochreiter. *Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München*. 1991.

[95] Schmidhuber J Hochreiter S. "Long short term memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[96] Rie Honda et al. "Mining of Moving Objects from Time-Series Images and its Application to Satellite Weather Imagery". In: *Journal of Intelligent Information Systems* 19.1 (July 2002), pp. 79–93. ISSN: 1573-7675. DOI: 10.1023/A:1015516504614. URL: https://doi.org/10.1023/A:1015516504614.

[97] J. J. Hopfield. "Neural networks and physical systems with emergent collective computational abilities". In: *Proceedings of the National Academy of Sciences of the United States of America* 79.8 (Apr. 1982), pp. 2554–2558. ISSN: 0027-8424. URL: http://view.ncbi.nlm.nih.gov/pubmed/6953413].

[98] Frank Höppner. "Discovery of Temporal Patterns. Learning Rules About the Qualitative Behaviour of Time Series". In: *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*. PKDD '01. Springer-Verlag, 2001, pp. 192–203. ISBN: 3-540-42534-9. URL: http://dl.acm.org/citation.cfm?id=645805.669998.

[99] Carol Hsin. "Implementation and Optimization of Differentiable Neural Computers". In: ().

[100] Tsuyoshi Idé and Koji Tsuda. "Change-Point Detection using Krylov Subspace Learning". In: *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 515–520. DOI: 10.1137/1.9781611972771.54. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611972771.54. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611972771.54.

[101] David Isele and Akansel Cosgun. "Selective experience replay for lifelong learning". In: *Thirty-second AAAI conference on artificial intelligence*. 2018.

[102] A. S. Iwashita and J. P. Papa. "An Overview on Concept Drift Learning". In: *IEEE Access* 7 (2019), pp. 1532–1547. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2886026.

[103] Bai J. "On the partial sums of residuals in autoregressive and moving average models". In: *Journal of Timeseries Analysis* 14.3 (1991).

[104] Robert A. Jacobs et al. "Adaptive Mixtures of Local Experts". In: *Neural Comput.* 3.1 (Mar. 1991), pp. 79–87. ISSN: 0899-7667. DOI: 10.1162/neco.1991.3.1.79. URL: http://dx.doi.org/10.1162/neco.1991.3.1.79.

[105] V. K. Jandhyala and I. B. MacNeill. "Residual partial sum limit process for regression models with applications to detecting parameter changes at unknown times". In: *Stochastic Processes and their Applications* 33.2 (1989), pp. 309–323. URL: https://EconPapers.repec.org/RePEc:eee:spapps:v:33:y:1989:i:2:p:309-323.

[106] MacNeill I Jandhyala V. "The change point problem: a review of applications". In: *Developments in water science* 27 (1986), pp. 381–387.

[107] Wei Jiang, Lianjie Shu, and Daniel W. Apley. "Adaptive CUSUM procedures with EWMA-based shift estimators". In: *IIE Transactions* 40.10 (2008), pp. 992–1003. DOI: 10.1080/07408170801961412. eprint: https://doi.org/10.1080/07408170801961412. URL: https://doi.org/10.1080/07408170801961412.

[108] Michael I. Jordan and Robert A. Jacobs. "Hierarchical Mixtures of Experts and the EM Algorithm". In: *Neural Comput.* 6.2 (Mar. 1994), pp. 181–214. ISSN: 0899-7667. DOI: 10.1162/neco.1994.6.2.181. URL: http://dx.doi.org/10.1162/neco.1994.6.2.181.

[109] Manuel Baena-Garca Jose et al. "Early Drift Detection Method". In: *Fourth international workshop on knowledge discovery from data streams* 6 (Dec. 2006), pp. 77–86. URL: http://dl.acm.org/citation.cfm?id=1248547.1248566.

[110] Ana Justel, Daniel Peña, and Ruben Zamar. "A multivariate Kolmogorov-Smirnov test of goodness of fit". In: *Statistics & Probability Letters* 35 (Oct. 1997), pp. 251–259. DOI: 10.1016/S0167-7152(97)00020-5.

[111] M.W. Kadous, Mohammed Waleed Kadous, and Supervisor Claude Sammut. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series*. Tech. rep. Unknown, 2002.

[112] Lukasz Kaiser et al. "Learning to Remember Rare Events". In: *CoRR* abs/1703.03129 (2017). arXiv: 1703.03129. URL: http://arxiv.org/abs/1703.03129.

[113] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. "A Convolutional Neural Network for Modelling Sentences". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, June 2014, pp. 655–665. DOI: 10.3115/v1/P14-1062. URL: https://www.aclweb.org/anthology/P14-1062.

[114] Pentti Kanerva. *Sparse Distributed Memory*. Cambridge, MA, USA: MIT Press, 1988. ISBN: 0262111322.

[115] Yoshinobu Kawahara and Masashi Sugiyama. "Change-Point Detection in Time-Series Data by Direct Density-Ratio Estimation". In: *Proceedings of the 2009 SIAM International Conference on Data Mining*, pp. 389–400. DOI: 10.1137/1.9781611972795.34. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611972795.34. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611972795.34.

[116] Maurice George Kendall. "Rank correlation methods." In: (1948).

[117] Eamonn Keogh, Stefano Lonardi, and Chotirat Ann Ratanamahatana. "Towards Parameter-free Data Mining". In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '04. Seattle, WA, USA: ACM, 2004, pp. 206–215. ISBN: 1-58113-888-1. DOI: 10.1145/1014052.1014077. URL: http://doi.acm.org/10.1145/1014052.1014077.

[118] Eamonn Keogh et al. "Compression-based data mining of sequential data". In: *Data Mining and Knowledge Discovery* 14 (Feb. 2007), pp. 99–129. DOI: 10.1007/s10618-006-0049-3.

[119] Eamonn J Keogh and Michael J Pazzani. "Derivative dynamic time warping". In: *Proceedings of the 2001 SIAM international conference on data mining*. SIAM. 2001, pp. 1–11.

[120] Seyed Mohammad Khansari-Zadeh and Aude Billard. "Learning Stable Nonlinear Dynamical Systems With Gaussian Mixture Models". In: *IEEE Transactions on Robotics* 27 (2011), pp. 943–957.

[121] Chang-Jin Kim and Charles Nelson. *State-Space Models with Regime Switching: Classical and Gibbs-Sampling Approaches with Applications*. 1st ed. Vol. 1. The MIT Press, 1999. URL: https://EconPapers.repec.org/RePEc:mtp:titles:0262112388.

[122] Sangyeon Kim and Myungjoo Kang. "Financial series prediction using Attention LSTM". In: *CoRR* abs/1902.10877 (2019). arXiv: 1902.10877. URL: http://arxiv.org/abs/1902.10877.

[123] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *CoRR* abs/1612.00796 (2016). arXiv: 1612.00796. URL: http://arxiv.org/abs/1612.00796.

[124] Ralf Klinkenberg. "Learning drifting concepts: Example selection vs. example weighting". In: *Intell. Data Anal.* 8 (Aug. 2004), pp. 281–300. DOI: 10.3233/IDA-2004-8305.

[125] Ralf Klinkenberg and Thorsten Joachims. "Detecting Concept Drift with Support Vector Machines". In: *In Proceedings of the Seventeenth International Conference on Machine Learning (ICML*. Morgan Kaufmann, 2000, pp. 487–494.

[126] Jan Koutník et al. "A Clockwork RNN". In: *CoRR* abs/1402.3511 (2014). arXiv: 1402.3511. URL: http://arxiv.org/abs/1402.3511.

[127] Ivan Koychev. "Gradual Forgetting for Adaptation to Concept Drift". In: *In Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*. 2000, pp. 101–106.

[128] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: 2009.

[129] Ludmila I. Kuncheva and Christopher J. Whitaker. "Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy". In: *Machine Learning* 51.2 (May 2003), pp. 181–207. ISSN: 1573-0565. DOI: 10.1023/A:1022859003006. URL: https://doi.org/10.1023/A:1022859003006.

[130] Guokun Lai et al. *Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks*. 2017. arXiv: 1703.07015 [cs.LG].

[131] Frantzeska Lavda et al. "Continual Classification Learning Using Generative Models". In: *arXiv preprint arXiv:1810.10612* (2018).

[132] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. "Data augmentation for time series classification using convolutional neural networks". In: 2016.

[133] Y. LeCun et al. "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural Computation* 1.4 (Dec. 1989), pp. 541–551. DOI: 10.1162/neco.1989.1.4.541.

[134] Y. Lecun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[135] Sang-Woo Lee et al. “Overcoming Catastrophic Forgetting by Incremental Moment Matching”. In: *CoRR* abs/1703.08475 (2017). arXiv: 1703.08475. URL: http://arxiv.org/abs/1703.08475.

[136] Zhizhong Li and Derek Hoiem. “Learning without Forgetting”. In: *CoRR* abs/1606.09282 (2016). arXiv: 1606.09282. URL: http://arxiv.org/abs/1606.09282.

[137] Jessica Lin, Eamonn Keogh, and Wagner Truppel. “Clustering of Streaming Time Series is Meaningless”. In: *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. DMKD ’03. San Diego, California: ACM, 2003, pp. 56–65. DOI: 10.1145/882082.882096. URL: http://doi.acm.org/10.1145/882082.882096.

[138] Jessica Lin et al. “Finding Motifs in Time Series”. In: *Proceedings of the Second Workshop on Temporal Data Mining* (Oct. 2002).

[139] Jason Lines, Sarah Taylor, and Anthony Bagnall. “Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 12.5 (2018), p. 52.

[140] Zachary Chase Lipton et al. “Learning to Diagnose with LSTM Recurrent Neural Networks”. In: *CoRR* abs/1511.03677 (2015). arXiv: 1511.03677. URL: http://arxiv.org/abs/1511.03677.

[141] Song Liu et al. “Change-point detection in time-series data by relative density-ratio estimation”. In: *Neural Networks* 43 (2013), pp. 72 –83. ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2013.01.012. URL: http://www.sciencedirect.com/science/article/pii/S0893608013000270.

[142] Xialei Liu et al. “Rotate your Networks: Better Weight Consolidation and Less Catastrophic Forgetting”. In: *CoRR* abs/1802.02950 (2018). arXiv: 1802.02950. URL: http://arxiv.org/abs/1802.02950.

[143] Andrew W. Lo. “The Statistics of Sharpe Ratios”. In: *Financial Analysts Journal* 58.4 (2002), pp. 36–52. DOI: 10.2469/faj.v58.n4.2453. eprint: https://doi.org/10.2469/faj.v58.n4.2453. URL: https://doi.org/10.2469/faj.v58.n4.2453.

[144] David Lopez-Paz and Marc'Aurelio Ranzato. "Gradient Episodic Memory for Continuum Learning". In: *CoRR* abs/1706.08840 (2017). arXiv: 1706.08840. URL: http://arxiv.org/abs/1706.08840.

[145] Viktor Losing, Barbara Hammer, and Heiko Wersing. "KNN Classifier with Self Adjusting Memory for Heterogeneous Concept Drift". In: Dec. 2016. DOI: 10.1109/ICDM.2016.0040.

[146] Ian B. MacNeilt. "Detecting unknown interventions with application to forecasting hydrological data". In: *JAWRA Journal of the American Water Resources Association* 21.5 (1985), pp. 785–796. DOI: 10.1111/j.1752-1688.1985.tb00172.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1752-1688.1985.tb00172.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1752-1688.1985.tb00172.x.

[147] Marcus A. Maloof and Ryszard S. Michalski. "Selecting Examples for Partial Memory Learning". In: *Machine Learning* 41.1 (Oct. 2000), pp. 27–52. ISSN: 1573-0565. DOI: 10.1023/A:1007661119649. URL: https://doi.org/10.1023/A:1007661119649.

[148] Henry B Mann. "Nonparametric tests against trend". In: *Econometrica: Journal of the Econometric Society* (1945), pp. 245–259.

[149] Rosario Mantegna. "Hierarchical structure in financial markets". In: *The European Physical Journal B: Condensed Matter and Complex Systems* 11.1 (1999), pp. 193–197. URL: https://EconPapers.repec.org/RePEc:spr:eurphb:v:11:y:1999:i:1:p:193-197.

[150] Pierre-François Marteau. "Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 31.2 (Feb. 2009), pp. 306–318. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2008.76. URL: http://dx.doi.org/10.1109/TPAMI.2008.76.

[151] Michael McCloskey and Neal J. Cohen. "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem". In: ed. by Gordon H. Bower. Vol. 24. Psychology of Learning and Motivation. Academic Press, 1989, pp. 109 –165. DOI: https://doi.org/10.1016/S0079-7421(08)60536-8. URL: http://www.sciencedirect.com/science/article/pii/S0079742108605368.

[152] Edward Meeds and Simon Osindero. "An Alternative Infinite Mixture of Gaussian Process Experts". In: *Proceedings of the 18th International Conference on Neural Information Processing Systems*. NIPS'05. Vancouver, British Columbia, Canada: MIT Press, 2005, pp. 883–890. URL: http://dl.acm.org/citation.cfm?id=2976248.2976359.

[153] C Monteiro et al. "Wind power forecasting : state-of-the-art 2009." In: (Nov. 2009). DOI: 10.2172/968212.

[154] Michael C. Mozer. "Induction of Multiscale Temporal Structure". In: *Proceedings of the 4th International Conference on Neural Information Processing Systems*. NIPS'91. Morgan Kaufmann Publishers Inc., 1991, pp. 275–282. ISBN: 1-55860-222-4. URL: http://dl.acm.org/citation.cfm?id=2986916.2986950.

[155] Abdullah Mueen and Nikan Chavoshi. "Enumeration of Time Series Motifs of All Lengths". In: *Knowl. Inf. Syst.* 45.1 (Oct. 2015), pp. 105–132. ISSN: 0219-1377. DOI: 10.1007/s10115-014-0793-4. URL: http://dx.doi.org/10.1007/s10115-014-0793-4.

[156] Abdullah Mueen and Eamonn Keogh. "Online Discovery and Maintenance of Time Series Motifs". In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '10. Washington, DC, USA: ACM, 2010, pp. 1089–1098. ISBN: 978-1-4503-0055-1. DOI: 10.1145/1835804.1835941. URL: http://doi.acm.org/10.1145/1835804.1835941.

[157] Adeel Mufti, Svetlin Penkov, and Subramanian Ramamoorthy. "Iterative Model-Based Reinforcement Learning Using Simulations in the Differentiable Neural Computer". In: *arXiv preprint arXiv:1906.07248* (2019).

[158] Klaus-Robert Müller et al. "Predicting Time Series with Support Vector Machines". In: vol. 1327. Apr. 2006, pp. 999–1004. DOI: 10.1007/BFb0020283.

[159] K. S. Narendra and K. Parthasarathy. "Identification and control of dynamical systems using neural networks". In: *IEEE Transactions on Neural Networks* 1.1 (Mar. 1990), pp. 4–27. DOI: 10.1109/72.80202.

[160] H. Nyquist. "Certain Topics in Telegraph Transmission Theory". In: *Proceedings of the IEEE* 90 (Mar. 2002), pp. 280 –305. DOI: 10.1109/5.989875.

[161] Tim Oates. "Identifying Distinctive Subsequences in Multivariate Time Series by Clustering". In: *KDD*. 1999.

[162] Tim Oates, Laura Firoiu, and Paul R. Cohen. "Clustering Time Series with Hidden Markov Models and Dynamic Time Warping". In: 1999.

[163] Bruno A. Olshausen and David J. Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision Research* 37.23 (1997), pp. 3311 –3325. ISSN: 0042-6989. DOI: https://doi.org/10.1016/S0042-6989(97)00169-7. URL: http://www.sciencedirect.com/science/article/pii/S0042698997001697.

[164] Bruno A. Olshausen and David J. Field. "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision Research* 37.23 (1997), pp. 3311 –3325. ISSN: 0042-6989. DOI: https://doi.org/10.1016/S0042-6989(97)00169-7. URL: http://www.sciencedirect.com/science/article/pii/S0042698997001697.

[165] M. Garcia Ortiz et al. "Generative Models from the perspective of Continual Learning". 2018. URL: https://openaccess.city.ac.uk/id/eprint/22452/.

[166] German Ignacio Parisi et al. "Continual Lifelong Learning with Neural Networks: A Review". In: *CoRR* abs/1802.07569 (2018). arXiv: 1802.07569. URL: http://arxiv.org/abs/1802.07569.

[167] Emilio Parisotto, Jimmy Ba, and Ruslan Salakhutdinov. "Actor-Mimic: Deep Multitask and Transfer Reinforcement Learning". In: *CoRR* abs/1511.06342 (2015).

[168] Seongsik Park et al. "Quantized Memory-Augmented Neural Networks". In: *CoRR* abs/1711.03712 (2017). arXiv: 1711.03712. URL: http://arxiv.org/abs/1711.03712.

[169] M. Pechenizkiy et al. "Online mass flow prediction in CFB boilers with explicit detection of sudden concept drift". English. In: *SIGKDD Explorations* 11.2 (2009), pp. 109–116. ISSN: 1931-0145.

[170] A. N. Pettitt. "A Non-Parametric Approach to the Change-Point Problem". In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 28.2 (1979), pp. 126–135. DOI: 10.2307/2346729. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.2307/2346729. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.2307/2346729.

[171] Daniel Philps, Artur S. d'Avila Garcez, and Tillman Weyde. "Making Good on LSTMs Unfulfilled Promise". In: *33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada* abs/1911.04489 (2019).

[172] Daniel Philps et al. “Continual Learning Augmented Investment Decisions”. In: *NeurIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy, Montreal, Canada* abs/1812.02340 (2018). arXiv: 1812.02340. URL: http://arxiv.org/abs/1812.02340.

[173] Daniel O. Price. “Sequential Analysis. By Abraham Wald. New York: John Wiley and Sons, Inc., 1947. 212 pp.
4.00”. In: *Social Forces* 27.2 (Dec. 1948), pp. 170–171. ISSN: 0037-7732. DOI: 10.2307/2572319. eprint: http://oup.prod.sis.lan/sf/article-pdf/27/2/170/5861907/27-2-170a.pdf. URL: https://doi.org/10.2307/2572319.

[174] Yao Qin et al. “A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction”. In: Aug. 2017, pp. 2627–2633. DOI: 10.24963/ijcai.2017/366.

[175] Alec Radford. “Improving Language Understanding by Generative Pre-Training”. In: 2018.

[176] Jack W. Rae et al. “Scaling Memory-Augmented Neural Networks with Sparse Reads and Writes”. In: *CoRR* abs/1610.09027 (2016). arXiv: 1610.09027. URL: http://arxiv.org/abs/1610.09027.

[177] Vijay Raghavan and Alaaeldin Hafez. “Dynamic Data Mining”. In: *Proceedings of the 13th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: Intelligent Problem Solving: Methodologies and Approaches*. IEA/AIE ’00. New Orleans, Louisiana, USA: Springer-Verlag, 2000, pp. 220–229. ISBN: 3-540-67689-9. URL: http://dl.acm.org/citation.cfm?id=352552.352668.

[178] Thanawin Rakthanmanon et al. “Searching and mining trillions of time series subsequences under dynamic time warping”. In: *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD’12*. ACM Press, 2012. ISBN: 9781450314626. DOI: 10.1145/2339530.2339576.

[179] Thanawin Rakthanmanon et al. “Searching and Mining Trillions of Time Series Subsequences Under Dynamic Time Warping”. In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’12. Beijing, China: ACM, 2012, pp. 262–270. ISBN: 978-1-4503-1462-6. DOI: 10.1145/2339530.2339576. URL: http://doi.acm.org/10.1145/2339530.2339576.

[180] Liva Ralaivola and Florence d'Alché Buc. "Dynamical Modeling with Kernels for Nonlinear Time Series Prediction". In: *NIPS*. 2003.

[181] Marc'Aurelio Ranzato et al. "Efficient Learning of Sparse Representations with an Energy-based Model". In: *Proceedings of the 19th International Conference on Neural Information Processing Systems*. NIPS'06. MIT Press, 2006, pp. 1137–1144. URL: http://dl.acm.org/citation.cfm?id=2976456.2976599.

[182] Marc' aurelio Ranzato, Y lan Boureau, and Yann L. Cun. "Sparse Feature Learning for Deep Belief Networks". In: *Advances in Neural Information Processing Systems 20*. Ed. by J. C. Platt et al. Curran Associates, Inc., 2008, pp. 1185–1192. URL: http://papers.nips.cc/paper/3363-sparse-feature-learning-for-deep-belief-networks.pdf.

[183] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN: 026218253X.

[184] Chotirat Ann Ratanamahatana and Eamonn J. Keogh. "Everything you know about Dynamic Time Warping is Wrong". In: 2004.

[185] Sylvestre-Alvise Rebuffi et al. "iCaRL: Incremental Classifier and Representation Learning". In: July 2017, pp. 5533–5542. DOI: 10.1109/CVPR.2017.587.

[186] Mark Bishop Ring. "Continual learning in reinforcement environments". PhD thesis. University of Texas at Austin Austin, Texas 78712, 1994.

[187] S. W. Roberts. "Control Chart Tests Based on Geometric Moving Averages". In: *Technometrics* 1.3 (1959), pp. 239–250. DOI: 10.1080/00401706.1959.10489860. eprint: https://www.tandfonline.com/doi/pdf/10.1080/00401706.1959.10489860. URL: https://www.tandfonline.com/doi/abs/10.1080/00401706.1959.10489860.

[188] A Robins. "Catastrophic Forgetting, Rehearsal and Pseudorehearsal". In: *Connection Science* 7.2 (1995), pp. 123–146. DOI: 10.1080/09540099550039318. eprint: https://doi.org/10.1080/09540099550039318. URL: https://doi.org/10.1080/09540099550039318.

[189] Anthony Robins. "Sequential learning in neural networks: A review and a discussion of pseudorehearsal based methods". In: *Intelligent Data Analysis* 8 (Aug. 2004), pp. 301–. DOI: 10.3233/IDA-2004-8306.

[190] David Rolnick et al. "Experience Replay for Continual Learning". In: *CoRR* abs/1811.11682 (2018). arXiv: 1811.11682. URL: http://arxiv.org/abs/1811.11682.

[191] Andrei A. Rusu et al. "Progressive Neural Networks". In: *CoRR* abs/1606.04671 (2016). arXiv: 1606.04671. URL: http://arxiv.org/abs/1606.04671.

[192] Paul Ruvolo and Eric Eaton. "ELLA: An Efficient Lifelong Learning Algorithm". In: *30th International Conference on Machine Learning, ICML 2013* 28 (Jan. 2013).

[193] H Sakoe and S Chiba. "Dynamic-programming algorithm optimization for spoken word recognition". English. In: *IEEE TRANSACTIONS ON ACOUSTICS SPEECH AND SIGNAL PROCESSING* 26.1 (1978), pp. 43–49.

[194] Marcos Salganicoff. "Tolerating Concept and Sampling Shift in Lazy Learning UsingPrediction Error Context Switching". In: *Artif. Intell. Rev.* 11.1-5 (Feb. 1997), pp. 133–155. ISSN: 0269-2821. DOI: 10.1023/A:1006515405170. URL: https://doi.org/10.1023/A:1006515405170.

[195] Cicero Dos Santos and Bianca Zadrozny. "Learning Character-level Representations for Part-of-Speech Tagging". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 2014, pp. 1818–1826. URL: http://proceedings.mlr.press/v32/santos14.html.

[196] Jeffrey C. Schlimmer and Richard H. Granger. "Incremental learning from noisy data". In: *Machine Learning* 1.3 (Sept. 1986), pp. 317–354. ISSN: 1573-0565. DOI: 10.1007/BF00116895. URL: https://doi.org/10.1007/BF00116895.

[197] Lukas Schott et al. "Robust Perception through Analysis by Synthesis". In: *CoRR* abs/1805.09190 (2018). arXiv: 1805.09190. URL: http://arxiv.org/abs/1805.09190.

[198] Terrence J. Sejnowski and Charles R. Rosenberg. "Parallel Networks that Learn to Pronounce English Text". In: *Complex Systems* 1 (1987).

[199] Skyler Seto, Wenyu Zhang, and Yichen Zhou. "Multivariate Time Series Classification Using Dynamic Time Warping Template Selection for Human Activity Recognition". In: *CoRR* abs/1512.06747 (2015). arXiv: 1512.06747. URL: http://arxiv.org/abs/1512.06747.

[200] Shai Shalev-Shwartz et al. "Pegasos: primal estimated sub-gradient solver for SVM". In: *Mathematical Programming* 127.1 (Mar. 2011), pp. 3–30. ISSN: 1436-4646. DOI: 10.1007/s10107-010-0420-4. URL: https://doi.org/10.1007/s10107-010-0420-4.

[201] William F. Sharpe. "The Sharpe Ratio". In: *The Journal of Portfolio Management* 21.1 (1994), pp. 49–58. ISSN: 0095-4918. DOI: 10.3905/jpm.1994.409501. eprint: https://jpm.pm-research.com/content/21/1/49.full.pdf. URL: https://jpm.pm-research.com/content/21/1/49.

[202] Noam Shazeer et al. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer". In: *CoRR* abs/1701.06538 (2017). arXiv: 1701.06538. URL: http://arxiv.org/abs/1701.06538.

[203] Donald Shepard. "A Two-dimensional Interpolation Function for Irregularly-spaced Data". In: *Proceedings of the 1968 23rd ACM National Conference*. ACM '68. ACM, 1968, pp. 517–524. DOI: 10.1145/800186.810616. URL: http://doi.acm.org/10.1145/800186.810616.

[204] Xingjian Shi et al. *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*. 2015. arXiv: 1506.04214 [cs.CV].

[205] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. "Temporal pattern attention for multivariate time series forecasting". In: *Machine Learning* 108.8-9 (2019), pp. 1421–1441.

[206] Hanul Shin et al. "Continual Learning with Deep Generative Replay". In: *CoRR* abs/1705.08690 (2017). arXiv: 1705.08690. URL: http://arxiv.org/abs/1705.08690.

[207] Lianjie Shu and Wei Jiang. "A New EWMA Chart for Monitoring Process Dispersion". In: *Journal of Quality Technology* 40.3 (2008), pp. 319–331. DOI: 10.1080/00224065.2008.11917737. eprint: https://doi.org/10.1080/00224065.2008.11917737. URL: https://doi.org/10.1080/00224065.2008.11917737.

[208] D Siegmund. "Change-points: from sequential detection to biology and back". In: *Sequential Analysis* 32.2-14 (2013), pp. 43–46.

[209] Daniel L. Silver. "Machine Lifelong Learning: Challenges and Benefits for Artificial General Intelligence". In: *Artificial General Intelligence*. Ed. by Jürgen Schmidhuber, Kristinn R. Thórisson, and Moshe Looks. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 370–375. ISBN: 978-3-642-22887-2.

[210] Daniel L. Silver and Robert E. Mercer. "The Task Rehearsal Method of Life-Long Learning: Overcoming Impoverished Data". In: *Advances in Artificial Intelligence*. Ed. by Robin Cohen and Bruce Spencer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 90–101. ISBN: 978-3-540-47922-2.

[211] Jake Snell, Kevin Swersky, and Richard S. Zemel. "Prototypical Networks for Few-shot Learning". In: *CoRR* abs/1703.05175 (2017). arXiv: 1703.05175. URL: http://arxiv.org/abs/1703.05175.

[212] R.J. Solomonoff. "A formal theory of inductive inference. Part I". In: *Information and Control* 7.1 (1964), pp. 1 –22. ISSN: 0019-9958. DOI: https://doi.org/10.1016/S0019-9958(64)90223-2. URL: http://www.sciencedirect.com/science/article/pii/S0019995864902232.

[213] Pablo Sprechmann et al. "Memory-based Parameter Adaptation". In: *CoRR* abs/1802.10542 (2018). arXiv: 1802.10542. URL: http://arxiv.org/abs/1802.10542.

[214] Stefan H. Steiner and Mark Jones. "Risk-adjusted survival time monitoring with an updating exponentially weighted moving average (EWMA) control chart". In: *Statistics in Medicine* 29.4 (2010), pp. 444–454. DOI: 10.1002/sim.3788. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.3788. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.3788.

[215] Y. Sun et al. "Concept Drift Adaptation by Exploiting Historical Knowledge". In: *IEEE Transactions on Neural Networks and Learning Systems* 29.10 (Oct. 2018), pp. 4822–4832. DOI: 10.1109/TNNLS.2017.2775225.

[216] Tae Kyung Sung, Namsik Chang, and Gunhee Lee. "Dynamics of Modeling in Data Mining: Interpretive Approach to Bankruptcy Prediction". In: *Journal of Management Information Systems* 16.1 (1999), pp. 63–85. DOI: 10.1080/07421222.1999.11518234. eprint: https://doi.org/10.1080/07421222.1999.11518234. URL: https://doi.org/10.1080/07421222.1999.11518234.

[217] Christian Szegedy et al. "Intriguing properties of neural networks". In: *CoRR* abs/1312.6199 (2013).

[218] U Thissen et al. "Using support vector machines for time series prediction". In: *Chemometrics and Intelligent Laboratory Systems* 69.1 (2003), pp. 35 –49. ISSN: 0169-7439. DOI: https://doi.org/10.1016/S0169-7439(03)00111-4. URL: http://www.sciencedirect.com/science/article/pii/S0169743903001114.

[219] Sebastian Thrun and Tom M Mitchell. *Learning one more thing*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER SCIENCE, 1994.

[220] Amal Rannen Triki et al. "Encoder Based Lifelong Learning". In: *CoRR* abs/1704.01920 (2017). arXiv: 1704.01920. URL: http://arxiv.org/abs/1704.01920.

[221] Amal Rannen Triki et al. "Encoder Based Lifelong Learning". In: *CoRR* abs/1704.01920 (2017). arXiv: 1704.01920. URL: http://arxiv.org/abs/1704.01920.

[222] Kuniaki Uehara and Mitsuomi Shimada. "Extraction of Primitive Motion and Discovery of Association Rules from Human Motion Data". In: *Progress in Discovery Science: Final Report of the Japanese Dicsovery Science Project*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 338–348. ISBN: 978-3-540-45884-5. DOI: 10.1007/3-540-45884-0_24. URL: https://doi.org/10.1007/3-540-45884-0_24.

[223] Paul E. Utgoff. "Incremental Induction of Decision Trees". In: *Machine Learning* 4.2 (1989), pp. 161–186. ISSN: 1573-0565. DOI: 10.1023/A:1022699900025. URL: https://doi.org/10.1023/A:1022699900025.

[224] Paul E. Utgoff, Neil C. Berkman, and Jeffery A. Clouse. "Decision Tree Induction Based on Efficient Tree Restructuring". In: *Machine Learning*. 1997, pp. 5–44.

[225] V Vapnik and Y Alexey. "Chervonenkis (1964)."On the Uniform Convergence of Relative Frequencies of Events to their Probabilities,"". In: *Theory of Probability and its Applications, v16 (2)* (), pp. 264–280.

[226] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[227] Ragav Venkatesan et al. "A Strategy for an Uncompromising Incremental Learner". In: *CoRR* abs/1705.00744 (2017). arXiv: 1705.00744. URL: http://arxiv.org/abs/1705.00744.

[228] Oriol Vinyals et al. "Matching Networks for One Shot Learning". In: *CoRR* abs/1606.04080 (2016). arXiv: 1606.04080. URL: http://arxiv.org/abs/1606.04080.

[229] Michael Vogt, Holger Dette, et al. "Detecting gradual changes in locally stationary processes". In: *The Annals of Statistics* 43.2 (2015), pp. 713–740.

[230] A. Waibel et al. “Phoneme recognition using time-delay neural networks”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.3 (Mar. 1989), pp. 328–339. DOI: 10.1109/29.21701.

[231] Jack Wang, Aaron Hertzmann, and David J Fleet. “Gaussian Process Dynamical Models”. In: *Advances in Neural Information Processing Systems 18*. Ed. by Y. Weiss, B. Schölkopf, and J. C. Platt. MIT Press, 2006, pp. 1441–1448. URL: http://papers.nips.cc/paper/2783-gaussian-process-dynamical-models.pdf.

[232] Zhiguang Wang, Weizhong Yan, and Tim Oates. “Time series classification from scratch with deep neural networks: A strong baseline”. In: *2017 International joint conference on neural networks (IJCNN)*. IEEE. 2017, pp. 1578–1585.

[233] Jason Weston, Sumit Chopra, and Antoine Bordes. “Memory Networks”. In: *CoRR* abs/1410.3916 (2014). arXiv: 1410.3916. URL: http://arxiv.org/abs/1410.3916.

[234] Gerhard Widmer and Miroslav Kubat. “Effective Learning in Dynamic Environments by Explicit Context Tracking”. In: *ECML*. 1993.

[235] Gerhard Widmer and Miroslav Kubat. “Learning in the presence of concept drift and hidden contexts”. In: *Machine Learning* 23.1 (Apr. 1996), pp. 69–101. ISSN: 1573-0565. DOI: 10.1007/BF00116900. URL: https://doi.org/10.1007/BF00116900.

[236] Christopher K. I. Williams and David Barber. “Bayesian Classification With Gaussian Processes”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 20.12 (Dec. 1998), pp. 1342–1351. ISSN: 0162-8828. DOI: 10.1109/34.735807. URL: https://doi.org/10.1109/34.735807.

[237] Yimin Xiong and Dit-Yan Yeung. “Time series clustering with ARMA mixtures”. In: *Pattern Recognition* 37.8 (2004), pp. 1675 –1689. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2003.12.018. URL: http://www.sciencedirect.com/science/article/pii/S0031320304000585.

[238] Shuliang Xu and Junhong Wang. “Dynamic Extreme Learning Machine for Data Stream Classification”. In: *Neurocomput.* 238.C (May 2017), pp. 433–449. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2016.12.078. URL: https://doi.org/10.1016/j.neucom.2016.12.078.

[239] Takehisa Yairi, Yoshikiyo Kato, and Koichi Hori. “Fault Detection by Mining Association Rules from Housekeeping Data”. In: *In Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space*. 2001.

[240] Zhilin Yang et al. *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. 2019. arXiv: 1906.08237 [cs.CL].

[241] H Yu. "High moment partial sum processes of residuals in ARMA models and their applications". In: *Journal of time series analysis* 28.1 (2007), pp. 72–91.

[242] Seniha Esen Yuksel, Joseph N. Wilson, and Paul D. Gader. "Twenty Years of Mixture of Experts". In: *IEEE Transactions on Neural Networks and Learning Systems* 23 (2012), pp. 1177–1193.

[243] George Udny Yule and Karl Pearson. "VII. On the association of attributes in statistics: with illustrations from the material of the childhood society, &c". In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 194.252-261 (1900), pp. 257–319. DOI: 10.1098/rsta.1900.0019. eprint: https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.1900.0019. URL: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.1900.0019.

[244] Wojciech Zaremba and Ilya Sutskever. "Reinforcement Learning Neural Turing Machines". In: *CoRR* abs/1505.00521 (2015). arXiv: 1505.00521. URL: http://arxiv.org/abs/1505.00521.

[245] Wojciech Zaremba and Ilya Sutskever. "Reinforcement Learning Neural Turing Machines". In: *CoRR* abs/1505.00521 (2015). arXiv: 1505.00521. URL: http://arxiv.org/abs/1505.00521.

[246] Poorya ZareMoodi, Sajjad Kamali Siahroudi, and Hamid Beigy. "A Support Vector Based Approach for Classification Beyond the Learned Label Space in Data Streams". In: *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. SAC '16. Pisa, Italy: ACM, 2016, pp. 910–915. ISBN: 978-1-4503-3739-7. DOI: 10.1145/2851613.2851652. URL: http://doi.acm.org/10.1145/2851613.2851652.

[247] Friedemann Zenke, Ben Poole, and Surya Ganguli. "Continual learning through synaptic intelligence". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 3987–3995.

[248] Han Zhang et al. *Self-Attention Generative Adversarial Networks*. 2018. arXiv: 1805.08318 [stat.ML].

[249] Junting Zhang et al. "Class-incremental learning via deep model consolidation". In: *arXiv preprint arXiv:1903.07864* (2019).

[250] Peter Zhang. "Zhang, G.P.: Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model. Neurocomputing 50, 159-175". In: *Neurocomputing* 50 (Jan. 2003), pp. 159–175. DOI: 10.1016/S0925-2312(01)00702-0.

[251] Saizheng Zhang et al. "Architectural Complexity Measures of Recurrent Neural Networks". In: *CoRR* abs/1602.08210 (2016). arXiv: 1602.08210. URL: http://arxiv.org/abs/1602.08210.

[252] Xiang-Lilan Zhang, Zhi-Gang Luo, and Ming Li. "Merge-Weighted Dynamic Time Warping for Speech Recognition". In: *Journal of Computer Science and Technology* 29.6 (Nov. 2014), pp. 1072–1082. ISSN: 1860-4749. DOI: 10.1007/s11390-014-1491-0. URL: https://doi.org/10.1007/s11390-014-1491-0.

[253] Zheng Zhang, Ping Tang, and Rubing Duan. "Dynamic Time Warping Under Pointwise Shape Context". In: *Inf. Sci.* 315.C (Sept. 2015), pp. 88–101. ISSN: 0020-0255. DOI: 10.1016/j.ins.2015.04.007. URL: https://doi.org/10.1016/j.ins.2015.04.007.

[254] Zheng Zhang et al. "Dynamic Time Warping Under Limited Warping Path Length". In: *Inf. Sci.* 393.C (July 2017), pp. 91–107. ISSN: 0020-0255. DOI: 10.1016/j.ins.2017.02.018. URL: https://doi.org/10.1016/j.ins.2017.02.018.

[255] Ji Hanlee Zhang N Siegmund D and Li J. "Detecting simultaneous change-points in multiple sequences". In: *Biometrika* 97 (2010), pp. 631–646.

[256] Yi Zheng et al. "Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks". In: *Web-Age Information Management*. Ed. by Feifei Li et al. Cham: Springer International Publishing, 2014, pp. 298–310. ISBN: 978-3-319-08010-9.

[257] Guanyu Zhou, Kihyuk Sohn, and Honglak Lee. "Online Incremental Feature Learning with Denoising Autoencoders". In: *AISTATS*. 2012.

[258] Indre Žliobaite, Mykola Pechenizkiy, and João Gama. "An Overview of Concept Drift Applications". In: vol. 16. Jan. 2016, pp. 91–114. ISBN: 978-3-319-26987-0. DOI: 10.1007/978-3-319-26989-4_4.