MACHINE LEARNING AND INFORMATION-THEORETIC APPROACHES FOR FINANCIAL APPLICATIONS

BY

KELECHI MOISE IKEGWU

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Informatics
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2021

Urbana, Illinois

Doctoral Committee:

Professor Robert J. Brunner, Chair, Director of Research
Professor Theodore Sougiannis
Assistant Professor Jeff McMullin
Assistant Professor Joseph Yun

Abstract

Firm-specific information contributes to resource allocation in capital markets. The prediction and inference of future firm-specific information are often critical factors that can lead to a prosperous economy. Machine Learning and Information Theory offer new, alternative methods to aid in the prediction and inference of future firm-specific information. Thus, we explore approaches from machine learning and information theory to improve existing research areas in accounting and finance. In Chapter 2, we utilized random forest trees to predict the annual direction of profitability for firms with a minimal amount of information. We demonstrate that we can out-perform benchmark models typically used in financial accounting research.

Major public firms announce their annual earnings during the first quarter of the year. The embedding information in these announcements effects the announcing firm and is transferred to or incorporated by other firms. Existing literature in finance and accounting focuses on measuring information transfers as effects stemming from firm-specific information releases. In Chapter 3, we discuss a solution to estimate information transfer via transfer entropy between random processes. We show that our solution to estimate information transfer scales better with data set size and is up to 1,072 times faster than all existing, open source solutions for large datasets. In Chapter 4, we present an alternative approach to estimate information transfer between firms centered around earnings announcements. We show that information transfer between firms is stronger for firms on days with unexpected earnings news. We also show that communities of firms are not adequately captured by characteristics that are the focus of existing literature.

To my Father, Mother, Joshua, Sandra, Ezinne, Natalia, Khamille, and Khalil.

Acknowledgments

First and foremost, I would like to thank my advisor, Robert Brunner, for his guidance, patience, and support during my graduate studies at the University of Illinois at Urbana-Champaign. I also wish to thank my committee members: Theodore Sougiannis, Jeff McMullin, and Joseph Yun, for their help and support. I am also grateful to Junhyung Kim, Vincent Reverdy, William Biscarri, Samantha Thrush, Matias Carrasco Kind, Jacob Trauger, Alice Perng, and Howard Hardiman.

— Kelechi Moise Ikegwu

This work was partially funded by the Graduate College Fellowship program at the University of Illinois. The work was partially funded by the National GEM Consortium. This work utilizes resources provided by the Innovative Systems Laboratory at the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign.

Table of Contents

Chapter 1 Introduction **1**
- 1.1 Machine Learning 1
- 1.2 Network Science 4
- 1.3 Information Transfer in Financial Markets 8
- 1.4 Figures and Tables 12

Chapter 2 Directional Profitability Predictions with Random Forests . . **20**
- 2.1 Introduction 20
- 2.2 Data 21
- 2.3 Methods 23
- 2.4 Random Forest Implementation to Predict Directional Profitability 26
- 2.5 Random Walk Benchmark Models 26
- 2.6 Comparative Analyses 29
- 2.7 Results 31
- 2.8 Conclusion and Future Work 34
- 2.9 Figures and Tables 36

Chapter 3 Information Transfer Estimation on Large Data **55**
- 3.1 Introduction 55
- 3.2 Estimating Transfer Entropy 56
- 3.3 PyIF 59
- 3.4 Comparative Analysis 60
- 3.5 Results 63
- 3.6 Conclusion 64
- 3.7 Figures and Tables 65

Chapter 4 Cross-Firm Information Transfers During Earnings Season: A Network Approach **71**
- 4.1 Introduction 71
- 4.2 Data 73
- 4.3 Estimating Dynamic Information Transfer Between Firms 73
- 4.4 Exploratory Data Analysis 77
- 4.5 Earnings Surprise as a Determinant of Dynamic Information Transfers 79
- 4.6 Community Detection 82
- 4.7 Conclusion 84

4.8 Figures and Tables 86

Chapter 5 Conclusion 102

5.1 Summary and Conclusion 102

Chapter 1

Introduction

An economic goal of a capitalistic society is to maximize wealth by allocating resources efficiently. Capital markets (such as the stock market) provide a method to help allocate resources efficiently. As a result, the prediction and inference of future financial events within a capital market are often critical to achieving the economic goal. Since firm-specific information contributes to resource allocation (i.e., labor, capital, and natural resources) in capital markets, the prediction of a future state for a firm (such as its profitability, financial condition, or revenue growth) is an essential factor contributing to resource allocation. In this introduction, we review several different methods that can be used to predict or inform about the future state of a firm.

1.1 Machine Learning

Machine Learning encompasses a vast set of statistical tools for understanding data and making predictions and inferences. These tools fall into supervised or unsupervised categories. For data-driven problems, typically, we have a response variable (or labels) referred to as Y, and p independent variables (or features) referred to as X. Given the assumption that there is some relationship between Y and X the relationship can be expressed as:

$$Y = f(X) + \epsilon \tag{1.1}$$

In Equation 1.1 f is an unknown function that models the relationship between X and Y with ϵ noise. In most cases, it is not possible to perfectly model f with real-world data, so we instead estimate the function f by an estimator ($\hat{f}$). If Y is a quantitative variable in Equation1.1, we use regression techniques to estimate f; however, if Y is a qualitative variable, we use classification techniques instead. $\hat{f}$ is sought for inference where one wants to estimate f to understand how $\hat{Y}$ (predicted labels) changes as one adjusts features. $\hat{f}$ is also sought for prediction where the only concern is the accuracy of predictions for Y. Machine learning provides a variety of approaches to estimate f for prediction and inference (see, e.g., James et al. (2013)).

Machine learning problems typically fall into either supervised learning, semi-supervised learning, or unsupervised learning. With supervised learning, one wants to fit a model on labels using a set of features. Semi-supervised learning consists of a domain with features where only a subset have corresponding labels. Typically, in this setting, one wishes to use the given features and partial labels to determine the remaining missing labels. Lastly, with unsupervised learning, one has features and no associated labels and seeks to find structure between the features. In this dissertation, the presented results will only utilize supervised learning.

1.1.1 Supervised Learning

In a supervised learning paradigm, it is not enough to only have $\hat{f}$ estimate f well on a dataset. Instead, we seek to estimate $\hat{f}$ sufficiently well so that this estimator will also perform well on new data that $\hat{f}$ has not seen before. To find a suitable $\hat{f}$, a standard methodology is to partition the data into a training set and testing set (see Figure 1.1). The training set consists of data to train the model. The testing set consists of data to assess $\hat{f}$'s ability to generalize well to new data. The researcher/practitioner typically determines the ratio of data used to train and assess the model. A specific [1] way to estimate $\hat{f}$'s performance for a regression model is to use an error metric such as mean squared error (MSE) given by:

[1] One can use many error metrics; however, for the sake of brevity MSE is the only error metric discussed.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2 \tag{1.2}$$

In Equation 1.2, y_i and x_i are the i^{th} observations of the labels and features respectively. $\hat{f}(x_i)$ is the prediction of $\hat{f}$ for the i^{th} observation and y_i is the label for the i^{th} observations. Training MSE indicates the mean standard error of the estimator's ability to learn the training data. Subsequently, there is a testing MSE associated with the mean standard error of the estimator on the withheld testing data. A model with a small testing MSE often generalizes well to the withheld data. Often, the training MSE and testing MSE differ where the training MSE underestimates the MSE for the testing MSE (see chapter 5 of James et al. (2013)). Given the importance of this task, a number of approaches have been developed that can produce a more accurate estimate of how well $\hat{f}$ will generalize to new data.

1.1.2 Validation Approaches

An alternative approach to training and assessing $\hat{f}$ on training and testing sets, respectively, is to create a validation dataset by randomly sampling the training dataset. The researcher/practitioner determines the sampling method. $\hat{f}$ trains on the training data, and the trained model assesses its ability to generalize with validation data. The validation MSE is more indicative of the test MSE. However, there are issues with this approach. For example, randomly sampled data is in the validation dataset. Given that the validation set is composed of random data, the validation MSE can be highly variable. A generally accepted solution to reducing the variability of the validation MSE is cross-validation.

A common cross-validation technique is k-fold cross-validation where the training data is split into k separate folds (see Figure 1.2). The first fold is withheld and used as the validation dataset to assess the model's performance. The remaining $k-1$ folds train the model. For the next iteration, the same procedure repeats where the second fold serves as the validation dataset to assess the model (see Figure 1.3) and the other $k-1$ folds train the model. This process repeats k times and produces k MSE scores $(MSE_1, MSE_2, ..., MSE_k)$. Averaging the MSE fold values yields the k-fold CV estimate:

$$CV = \frac{1}{k}\sum_{i=1}^{k} MSE_i \tag{1.3}$$

where k is the number of folds.

Selecting a large value k can be computationally expensive as the model must be retrained on k different folds. In addition to this, the selection of k can affect the model's ability to generalize to new data well. Typically with a smaller value of k you have more bias, and a larger value of k has more variance. For example consider when $k = 3$ and $k = 10$, the training folds will have n_{train} observations where $n_{train} = \frac{(k-1)n}{k}$ and the validation fold will have $n_{validation}$ observations where $n_{validation} = \frac{k-(k-1)n}{k}$. When $k = 10$, 90% of the data is used to train the model, and the remaining 10% assesses the model. The model trains on very similar datasets, and the predictions from the model on the cross-validation datasets are highly correlated. The highly correlated datasets can produce a model with a high variance. If $k = 3$ 66.7% of the data trains the model and the remaining 33.3% of the data assesses the model. In this case, cross-validation averages fewer predictions that are less correlated with each other since the overlap between the training sets are smaller. When $k = 3$ this results in a CV score with lower variance, but since each model is trained on a smaller set of the original training data there is potential for a higher bias.

1.2 Network Science

Network science is the study of networks, and a network (or graph) consists of a collection of nodes joined by connections (commonly referred to as edges). Networks capture interactions between nodes, and over the years, scientists have discovered that specific patterns of interactions have a significant effect on the network's behavior (see Newman (2010). This field has evolved, and a vast set of tools have been developed to analyze, model, and understand networks (again, see Newman (2010)). Researchers have used networks in financial markets to determine which market sectors are related around specific events. For example, Billio et al. (2012) and Diebold and Yilmaz (2011) found that banks play an essential role in transmitting shocks relative to the remaining set of financial firms, consistent with banks playing a more central role in systemic risk for the economy around the 2007-2009 financial crisis. Thus, network science can help us understand more about the probable future states of firms.

Networks can be formed from data to offer an alternative representation of the data, allowing many new analysis techniques. Network Theory involves the study of networks as a representation of relationships between nodes. Throughout this section, G denotes a network, m denotes the number of edges in a network, and n denotes the number of nodes. A graph with n nodes and m edges is denoted as $G(n, m)$.

Undirected graphs do not have to specify the direction of an edge. For example, if Node A and Node B are connected, an undirected graph will not distinguish whether Node "A" connects to Node "B" or vice-versa; an undirected graph will only retain information about nodes A and B having a connection. Figure 1.4 shows an example of an undirected graph $G(7, 16)$. The number of edges (or connections to other nodes) a particular node has is known as the degree of the node. Figure 1.4, the node "FB" has a degree of six because it connects to six other nodes. Degree is a standard metric used in network theory.

It is also possible for a graph to retain additional information about edges. Another source of information is the direction of the edges; graphs that store information about edges' directions are called directed graphs. For example, consider a directed graph with two nodes (Node "A" and Node "B"). If Node "A" connects to Node "B" a directed graph would contain information that Node "A" has an edge to Node "B" but Node "B" does not have an edge to Node "A". Figure 1.5 shows an example of a directed graph; in Figure 1.5 there is an edge from "JNJ" to "FB" however there is no edge from "FB" to "JNJ".

Weighted networks (or weighted graphs) assign weights to the edges in a network. Instead of having a binary connection to determine whether an edge is present or not, the weight can provide a measure of the strength between the connections among different nodes. For example, if one has an attribute to determine the strength between nodes, such as a statistic from the data, it can be used as the weight value of an edge to determine the strength of the connection between those nodes. Edge weights are applicable to both directed and undirected graphs. Ergo it is possible to have an undirected weighted graph or directed weighted graph. Figure 1.6 provides an example of a directed weighted network.

Under certain circumstances, it is more convenient to express a graph as an adjacency matrix. The graph in Figure 1.4, is converted to a table with column and row headers in Table 1.1. However, it is mathematically more convenient to express Table 1.1 as an adjacency matrix A (see Equation 1.4).

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{1.4}$$

Similar to Table 1.1, each element (A_{ij}) in the adjacency matrix A is assigned a 1 or 0 based on if there is a connection between a row index (i) and column index j. Note, the adjacency matrix in Equation 1.4 is symmetric because the direction of edges are not taken into account; ergo if node i and node j are connected A_{ij} and A_{ji} both have a value of 1. This approach can be extended to represent directed graphs by assigning a 1 to a matrix element A_{ij} where node i is connected to node j. Lastly, you can express the weight of edges in adjacency matrices by assigning the weight values, which are non-binary, as the matrix elements.

1.2.1 Analysis of Networks

After forming a network from data, common metrics are often sought to help describe the structure of the network. The first metric is the degree of nodes where the degree can be defined in Equation 1.5 as the sum of edges for a particular node. In Figure 1.7, the node sizes are scaled based on the degree.

$$m_i = \sum_{ij} A_{ij} \tag{1.5}$$

where A is an adjacency matrix. Subsequently, there are degree measures for directed networks, such as in-degree, which counts the number of edges connected to node i (see Equation 1.6), and out-degree, which counts the amount of edges that node i is connected to (see Equation 1.7).

$$m_i^{in} = \sum_{j=1}^{n} A_{ij} \tag{1.6}$$

$$m_i^{out} = \sum_{i=1}^{n} A_{ij} \tag{1.7}$$

The network's density provides a numerical value for the proportion of actual connections out of all potential connections in a network. If all potential connections in a network are present, the density will have a numerical value of 1, and if there are no connections in an undirected network, the density will be 0. The maximum number of potential connections in a network is $n(n-1)/2$. The degree summed for all nodes is m (for directed networks, the summed in-degree and out-degree for all nodes is m). The density for a directed graph is:

$$D = \frac{m}{n(n-1)} \tag{1.8}$$

Centrality is another metric that is used to measure the importance of individual nodes within a network. Measuring the degree of nodes in relation to other nodes is called degree centrality (see Equation 1.9).

$$C_i^d = m_i = \sum_{ij} A_{ij} \tag{1.9}$$

1.3 Information Transfer in Financial Markets

The price discovery process plays a vital role in achieving the economic goal for a society. In particular, how information transfers from one price to another is useful in the price discovery process. Information is embedded in price, and an efficient market changes rapidly (if not instantaneously) when new information arrives. For example, Apple Inc. recently announced that their new line of computers (Macs) would use central processing units (CPUs) developed in-house and transition away from Intel CPUs. This information will rapidly be reflected both in Apple's stock price and Intel's stock price in ideal settings.

In the Apple and Intel example, it is probable that additional information reflects in Intel's price change at a given moment. However, it is difficult to determine price changes for specific events. The entire set of information that reflects in the price change at a given time is unknown. Prior studies have examined the price changes over long windows such as days or weeks (see Foster (1981) and Baginski (1987)). In particular, Foster (1981) measured information transfers from firms that announce earnings and non-announcement firms in the same industry based on short-window price reactions. Subsequent studies use a similar approach to examine short-window price reactions to firm-level earnings announcements (see Clinch and Sinclair (1987), Pownall and Waymire (1989), Han and Wild (1990), WANG (2014), and Hann et al. (2019)).

1.3.1 Information Theory

Information theory provides a framework for studying the quantification, storage, and communication of information (see Stone (2015)). Information theory has been shown to be useful in the price discovery process (see Billio et al. (2012) or Diebold and Yilmaz (2011)), as it offers an alternative approach for understanding the impacts of information on price. In particular, it offers alternative methods to understand information transfer between random processes (such as equity prices).

In this dissertation, we measure information transfer in financial markets with the relatively new measure Transfer Entropy (TE) (see Schreiber (2000)). TE allows us to measure information transfers in a broader sense and avoids assumptions of prior works. We avoid the assumption of links between firms due to a shared characteristic (such as shared industry) or a known event (such as earnings announcements). TE quantifies the reduction in uncertainty in one random process from knowing past realizations of another random process.

For example, it is unlikely one could predict Intel's closing price tomorrow with no information very accurately. However, knowing Intel's price today reduces uncertainty in the prediction for its price tomorrow. In addition to this, knowing Apple's price today may help further reduce the uncertainty in predicting Intel's future price. If Apple's price today reduces the uncertainty further, TE will be positive, indicating an information transfer from Apple to Intel. The TE value will vary at any given time, given the information incorporated into Apple's and Intel's prices.

Schreiber (2000) discovered TE and coined the name "transfer entropy," although Paluš et al. (2001) also independently discovered the concept as well. To define TE, one must begin with the building blocks of information theory [2]. Shannon Information is:

$$h(x) = \log_2 \frac{1}{p(x)} \tag{1.10}$$

[2] For simplicity, we use discrete events to introduce these concepts.

where x is an event and $p(x)$ is probability of x occurring. Shannon's information definition implies that values of x with small probabilities contain more information. Consequently, values of x that are more probable (or common) contain less information. Entropy is the amount of uncertainty or disorder in a random process.

$$H(X) = \frac{1}{n}\sum_{i=1}^{n} \log_2 p(x) \tag{1.11}$$

In Equation 1.11, we define entropy as the weighted average of Shannon Information. Conditional entropy is the entropy after conditioning on another process (Y):

$$H(X|Y) = \sum_{y \in \Omega_y} p(y)H(X|y) \tag{1.12}$$

In Equation 1.12, $y \in \Omega_y$ represents the occurrence of y in a set of possible events for y, and $H(X|y)$ represents conditional entropy on a single event y (or $H(X|y) = \sum_{x\in\Omega_x} \frac{p(x|y)}{\log_2 p(x|y)}$). Mutual Information quantifies the amount of information shared across random variables:

$$I(X, Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{1.13}$$

In Equation 1.13 consider $H(X) - H(X|Y)$, $H(X)$ computes the entropy of X and $H(X|Y)$ computes the entropy of X conditioned on Y. The difference between $H(X)$ and $H(X|Y)$ captures the shared information between X and Y. Mutual Information is a symmetric measure ergo $I(X, Y) = I(Y, X)$.

Lagged mutual information $I(X_t : Y_{t-k})$ can be used as a time-asymmetric measure of information transfer from Y to X where X and Y are both random processes, k is a lag period, and t is the current time period. However, lagged mutual information is unsatisfactory as it does not account for a shared history between the processes X and Y (see Kaiser and Schreiber (2002)). While similar to lagged mutual information, TE also considers the dynamics of information and how these dynamics evolve in time (see Schreiber

(2000)). TE is also an asymmetric measure of information transfer. Thus, TE computed from process A to process B may yield a different result than TE computed from B to A. The information-theoretic framework and these measures have led to various applications in different research areas (see Bossomaier et al. (2016)).

TE considers the shared history between two processes via conditional mutual information. Specifically, TE conditions on the past of X_t to remove any redundant or shared information between X_t and its past. This also removes any information in the process Y about X at time t that is in the past of X (see Williams and Beer (2011)). Transfer entropy T (where the transfer of information occurs from Y to X) is:

$$T_{Y \to X}(t) \equiv I(X_t : Y_{t-k} | X_{t-k}) \tag{1.14}$$

Kraskov et al. (2004) shows that transfer entropy can be expressed as the difference between two conditional mutual information computations:

$$T_{Y \to X}(t) = I(X_t | X_{t-k}, Y_{t-k}) - I(X_t | X_{t-k}) \tag{1.15}$$

The intuition of this definition is that TE measures the amount of information in Y_{t-k} about X_t after considering the information in X_{t-k} about X_t. Put differently, TE quantifies the reduction in uncertainty about X_t from knowing Y_{t-k} after considering the reduction in uncertainty about X_t from knowing X_{t-k}.

1.4 Figures and Tables

Figure 1.1: This figure shows how a data set can be partitioned for a supervised machine learning paradigm. The "data-set" is partitioned into two unequal sets with the "Train Set" having more data assigned to it than the "Test Set". The "Train Set" and "Test Set" ratios can vary and is typically determined by the practitioner. There are many ways to partition the data for the train and test sets. For example the dataset can be partitioned sequentially from the data, or form partitions based on the data being randomly sampled.

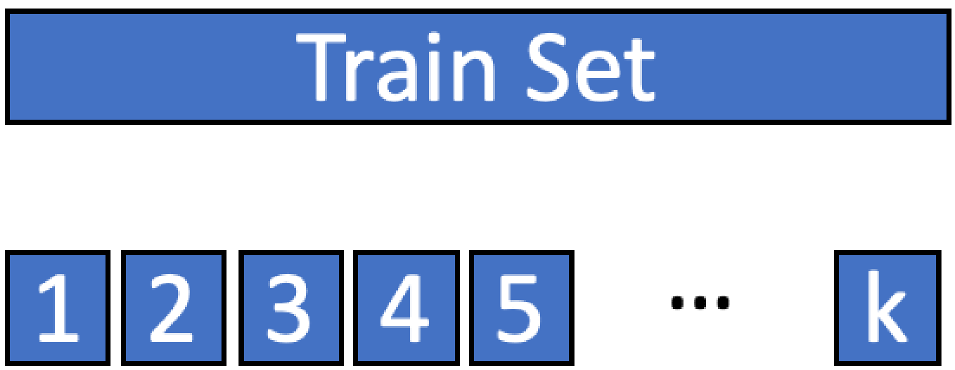

Figure 1.2: The "Train Set" here represents the "Train Set" in Figure 1.1. In this figure the data is split into k folds. How the k folds are created can vary. A common technique is to randomly divide the "Train Set" observations into k equal folds.

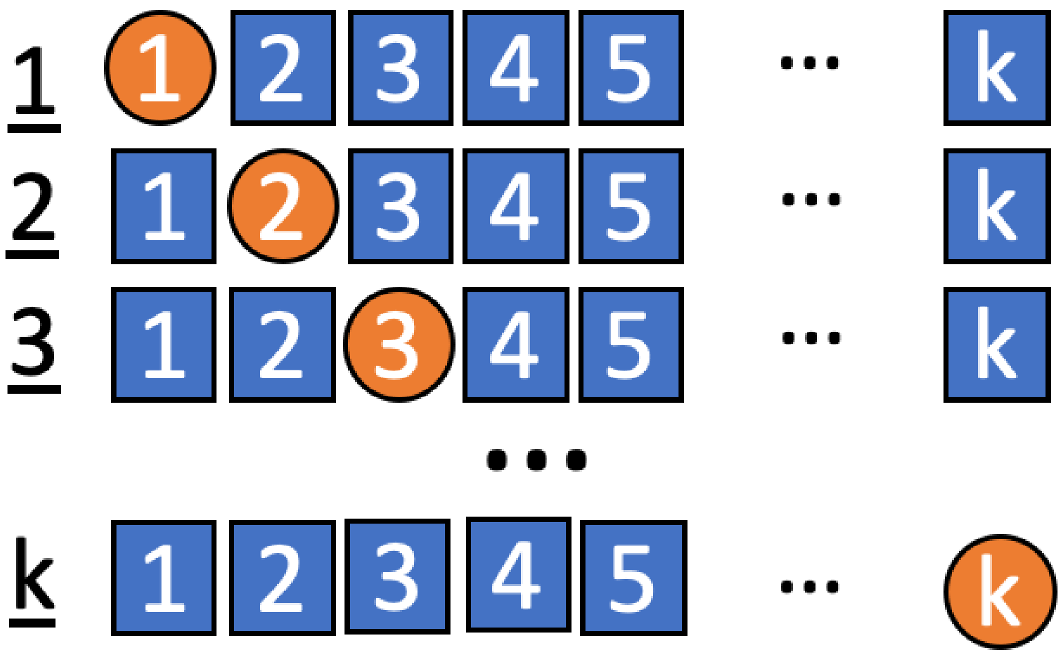

Figure 1.3: This figure breaks down how k-Fold Cross Validation works. These folds are created from "Train Set" in Figure 1.2. For the first iteration (where you see 1 underlined), the first fold (circle labeled 1) is used to assess the models ability and the remaining folds (squares) are used to train the model. The second iteration follows the same procedure as the first except only the second fold is withheld and the other folds are used. This repeats k times.

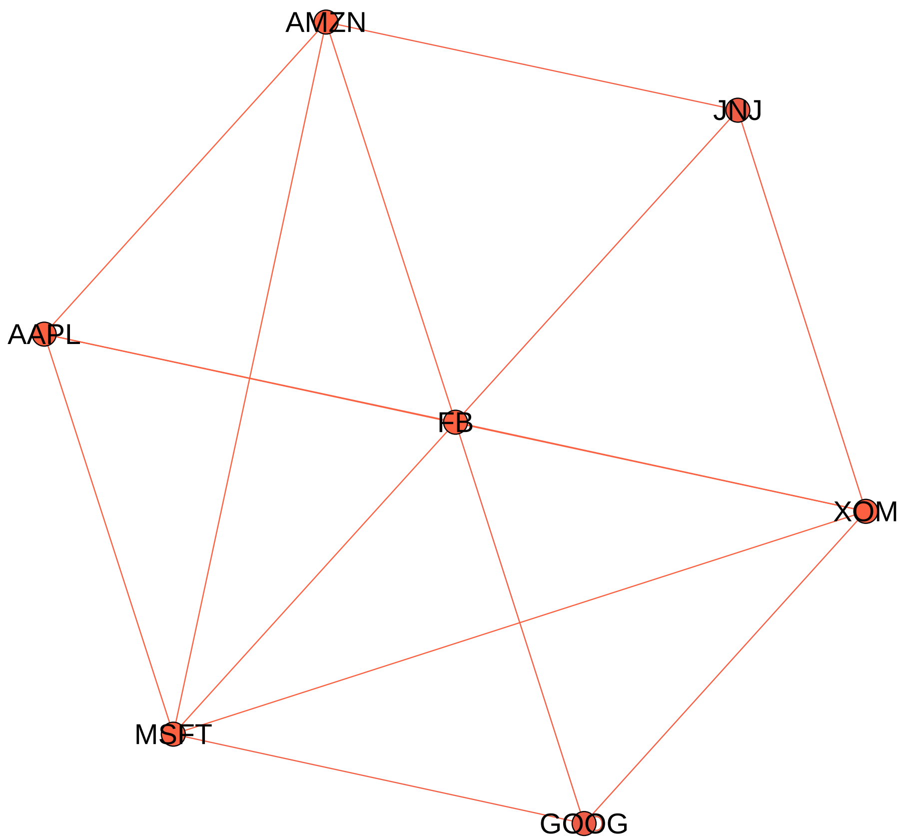

Figure 1.4: This figure contains an example of an undirected graph with seven nodes and sixteen edges. This graph is formed from randomly selecting stock symbol tickers from a set of tickers and forming connections randomly. The purpose of this data is solely to demonstrate the basics of network theory. The nodes are defined as circles and have ticker symbols overlaid on them. Edges are formed between nodes with lines between them. Since this is an undirected graph there is no directional information or no distinction as to whether one node is connected to another or vice-versa. The only information represented from an undirected graph is that they are connected.

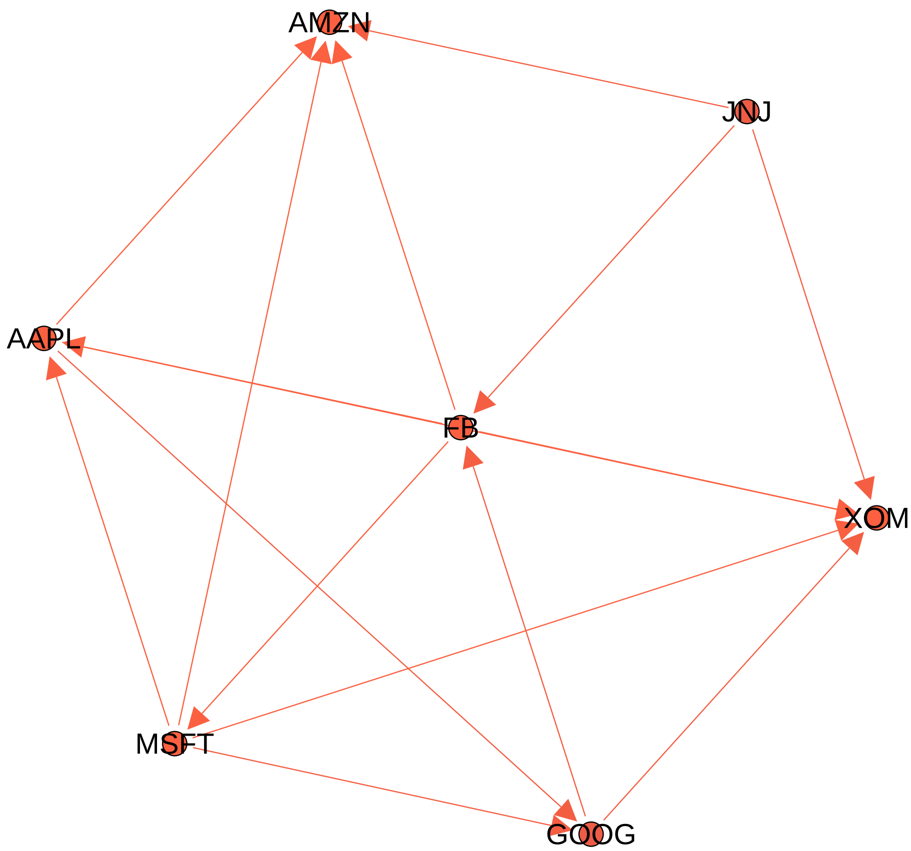

Figure 1.5: This graph is nearly identical to the graph in Figure 1.4 except this graph is a directed network. This means that it contains directional information between the nodes. This directional information is communicated based on the position of the arrow in a line. For example, the node FB has an edge to $MSFT$ however $MSFT$ does not have an edge to FB. The directional information allows other metrics to be used such as in-degree (which represents the amount of edges directed toward a node) and out-degree (which represents the amount of edges directed away from a node. In this example FB has an in-degree of two, an out-degree of four and a degree of six.

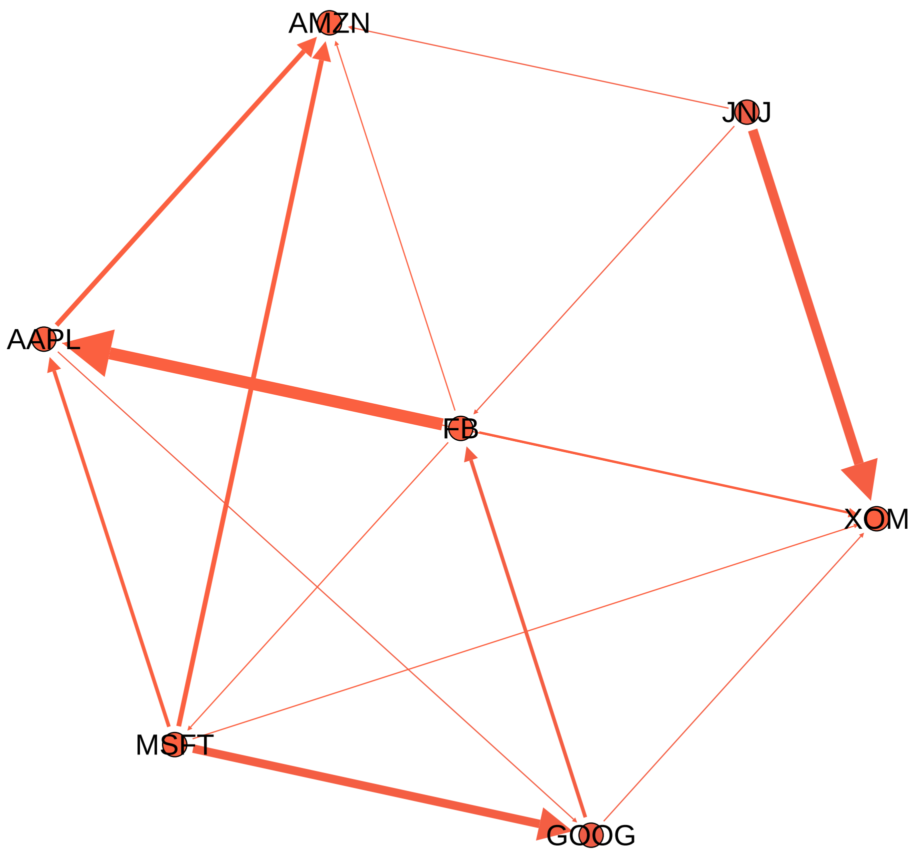

Figure 1.6: A nearly identical graph to the graph in Figure 1.5. This directed graph contains weighted edges where a thick edge represents a high weight value (or strong connection). The smaller the edge weight value the thinner the edge will be. Here FB has a strong connection to $AAPL$ whereas FB has a weak connection to $AMZN$.

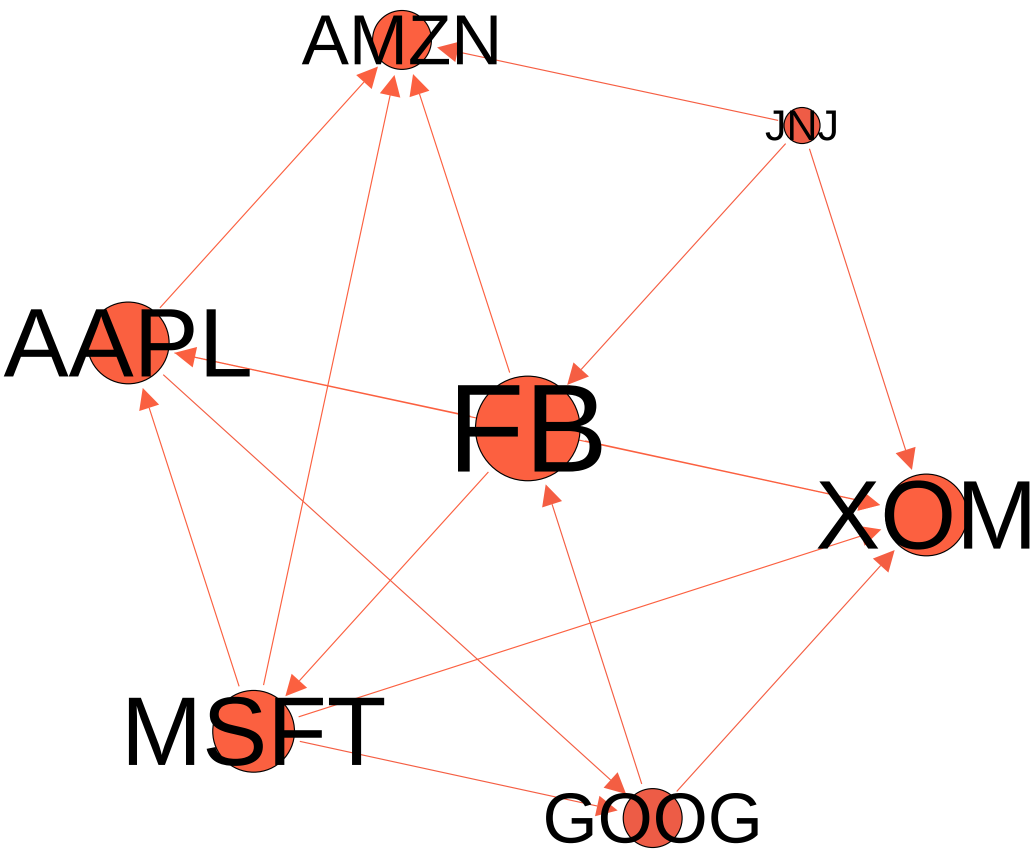

Figure 1.7: A nearly identical graph to the graph in Figure 1.5. Here the node sizes are scaled based on the degree. The higher the degree the larger the node size and consequently the smaller the degree the smaller the node size.

	MSFT	AAPL	AMZN	FB	JNJ	GOOG	XOM
MSFT	0	1	1	1	0	1	1
AAPL	1	0	1	1	0	1	1
AMZN	1	1	0	1	1	0	0
FB	1	1	1	0	1	1	1
JNJ	0	0	1	1	0	0	0
GOOG	1	1	0	1	0	0	1
XOM	1	1	0	1	0	1	0

Table 1.1: This table contains an example of simulated data. This table was formed from randomly selecting stock symbol tickers from a set of tickers. The purpose of this data is solely to demonstrate the basics of network theory. Here the columns and row indicies belong to the selected tickers. The values contain either a one to represent if there is a connection between the ticker at a particular row index i and the ticker of a particular column j or a zero if there is no connection.

Chapter 2

Directional Profitability Predictions with Random Forests

2.1 Introduction

An economic goal of a society is to maximize wealth. In order to maximize wealth, resources such as labor, capital, and natural resources must be allocated efficiently to firms. Capital markets (such as the stock market) allow capital to be efficiently allocated to said firms. Firm profitability can affect society, whether through job growth or loss, the offered services, the provision of goods, or personal gain or loss (see Spiceland et al. (2018)). Predicting the future profitability for firms is an essential endeavor as it is central to the valuation of the firm and the securities they issue (see Monahan (2018)).

A study by Bradshaw et al. (2012) provides evidence that analysts' predictions are not superior to random walk predictions, indicating that traditional methods provide little efficacy. Monahan (2018) conducts a review and discussion on the accounting research related to out-of-sample profitability predictions. He provides evidence that models in prior works have lower out-of-sample accuracy than random walk models. This provides motivation to seek an alternative method, given the shortcomings of traditional methods. Sarsam et al. (2020), Hang and Banks (2019), Choudhury et al. (2021), and Yang (2020) are relatively

This chapter contains material from the following working paper:

- Vic Anand, Robert Brunner, Kelechi Ikegwu, and Theodore Sougiannis. Predicting profitability using machine learning. *SSRN*, Oct 2019. URL `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3466478` recent examples that demonstrate the effectiveness of machine learning for prediction and inference in a variety of applications. Thus, we explore machine learning methods to predict the future profitability of firms.

An approach outlined in Monahan (2018) is to select five items: the labels (or what to predict), the features (or predictors), a regression model (where the functional form is determined by the researcher), and the training and testing datasets. One can use a parametric approach and assume the functional form of f (see Equation 1.1) and estimate a set parameters; however, it is likely that the functional form will not match the true unknown form of f which would provide poor results. Given our objective to find the best f in Equation 1.1 that predicts Y reasonably well, we use non-parametric methods. In addition, we employ cross-validation, described in section 1.1.2, to avoid overfitting the model. Lastly, we use machine learning to predict directional changes rather than the magnitude of changes in the firm's profitability. We make this choice to account for working with a smaller dataset. Ball and Brown (1968) and Ou and Penman (1989) have shown that the future directional change of profitability is a useful indicator.

Our stated research objective is to determine if a machine learning model can be constructed to outperform a random walk model. This chapter will describe the datasets used in this study and the methods used to predict firms' directional change of profitability. Finally, comparative analyses will be conducted between the proposed model and random walk model, followed by a discussion of results and potential future directions of this research.

2.2 Data

We follow the data selection process described in Hou et al. (2012) in this research. We use all observations from the fiscal years 1963 - 2016 from the Compustat fundamentals annual file. We merge this data set with the CRSP monthly returns file, which includes all securities listed on the NYSE, Amex, and Nasdaq markets. The securities selected have share codes 10 or 11, indicating that they are ordinary common shares of US companies and exclude ADR's and REITs. We remove observations if total assets, common equity, dividends, income before extraordinary items, or accruals are missing, resulting in 194,172 observations. Given the objective to predict the future direction of profitability for firms, each firm must have at least three observations; the rationale behind this choice is to compute two price changes for a firm. Excluding firms with fewer than four years of observations, the final dataset consists of 160,653 observations.

The following variables were retrieved and computed for this research: Return on common equity (ROE), Return on assets (ROA), return on net operating assets (RNOA), Cash flow from operations (CFO), and free cash flow (FCF). Table 2.1 contains more details about the variables used from Compustat and Table 2.2 shows how the profitability measures are computed.

Table 2.3 provides descriptive statistics on variables retrieved from Compustat. Outliers are present in most variables, and all of them have positively skewed distributions given that the mean is greater than the median. Table 2.4 provides the descriptive statistics on profitability measures. Since the raw variables construct the profitability measures, their values are also extreme. However, unlike the raw variables, the distributions of the targets are negatively skewed (the mean is smaller than the median). Table 2.5 shows the percentage of times the target variables increase in the holdout period or testing data (2012-2015). With the exception of CFO and FCF during 2013, there are no relatively significant class imbalances for directional changes in profitability measures.

Tables 2.6, 2.7, 2.8, 2.9, and 2.10 show the Pearson correlations between variables that will serve as the features in the random forest models. Each table shows the correlations between the profitability measure, the mean industry profitability measure, the median industry profitability measure, the standard deviation of the industry profitability measure, and the profitability measure at time $t-1$, and $t-2$. Since correlations of raw variables are extreme due to large outliers, we truncate each variable at 2.5% on each side of its distribution. Many of the features in these tables are correlated. Nevertheless, the proposed machine learning method performance is insensitive to multicollinearity.

2.3 Methods

2.3.1 Cross Validation for Time Series

For this work, the supervised learning paradigm described in section 1.1.1 will be employed. We use 90% of the data for training and the remaining 10% for testing. The first 90% of data contains data from the fiscal years 1963-2011, and the last 10% of data is from 2011-2016. Given the objective to predict the year-to-year change, we cannot predict using features for firms in the fiscal year 2016 because the fiscal year 2017 is not in the sample. Given the time-series nature of the data, we use blocked k-fold cross-validation as described in Cerqueira et al. (2019). The difference between blocked k-fold cross-validation and k-fold cross-validation is that the observations are not randomly shuffled and then divided into k folds. Instead, the observations are split into k blocks of contiguous observations.

2.3.2 Tree based Machine Learning Methods

Many machine learning models can provide high accuracy scores. However, there is a trade-off between interoperability and performance (see James et al. (2013), Chapter 2). Given that the research objective is more aligned with prediction than inference, we emphasize model performance. Tree-based models are flexible, simple to implement, and have high performance (see James et al. (2013), Chapter 8). In addition to this, tree-based methods can handle multicollinearity and non-linearity better than linear regression methods traditionally used in the financial accounting domain (see Monahan (2018)).

2.3.3 Decision Trees

Decision Trees are the building blocks of tree-based machine learning methods. Decision Trees are directed graphs with no cycles, meaning there are no nodes with edges to the same node, there is no more than one edge connecting two nodes, and there is only one path connecting two nodes. We refer to edges in trees as branches. Consider three nodes node A, B, and C, if node A has a branch to nodes B and C, B and C are the children of node A. In this case, node A is the parent of nodes B and C. The root node starts the decision tree and does not have a parent. Lastly, a node with no children signifies a stopping point within the decision tree and is referred to as a leaf node (see Figure 2.1 for an example).

In a supervised machine learning paradigm, decision trees are constructed from the training data set. Each node in the decision tree will represent the training features (or feature space, i.e. $X_1, ..., X_p$) partitioned into particular regions. Using the CART algorithm (see Breiman et al. (1984)) a decision tree will create m distinct non-overlapping regions R. The root node will contain a rule to partition the feature space into two non-overlapping regions (R_1 and R_2). The process will repeat recursively, in other words, both R_1 and $R2$ will be further divided by a splitting criteria to partition the data into two additional regions. $R1$ will be divided into R_3 and R_4 and R_2 into R_5 and R_6.

In a classification setting, a decision tree at a given region will determine the most optimal feature to select from the set of features to partition the data. The selection is typically based on the feature's ability to minimize error. A common metric is gini-impurity, where gini-impurity measures the variance across C classes. We can define gini-impurity as:

$$G_{index} = \sum_{i=1}^{C} p(i) \cdot (i - p(i)) \tag{2.1}$$

where C is the number of classes in a label and $p(i)$ is the probability of an observation labeled class i. A small value of gini-impurity indicates that a particular region has many observations from a single class.

Decision trees have stopping criteria. For example, we stop creating additional regions when a child has less than a defined number of observations at a particular region or when the difference of gini-impurity between the parent and its children is less than a specified amount. Lastly, to predict values of Y from another set of data (containing the same features), new observations can be passed into the trained tree. The trained tree will select the appropriate region for an observation based on the splitting criteria rules.

Decision Trees offer an intuitive explanation of the results it produces. However, there are some caveats to this method. A few additional observations or any modification to observations in a feature space can drastically change a decision tree because it is using a greedy search to build a tree (see Koning and Smith (2017)). Meaning that at each stage of the tree-building process, a node is split using a local optimal choice instead of a choice that will find the optimal choice on a global level.

2.3.4 Random Forests

Random forests can overcome some of the shortcomings of decision trees. Random forests are an ensemble of decision trees used to make predictions. However, a random forest is less interpretable than a single decision tree. A random forest will create B decision trees. For each decision tree, the random forest will assign a random dataset to each decision tree in the forest by sampling the data with replacement (or bootstrapping). In addition to this, the random forest will choose a subset of random features (with replacement) to construct each decision tree.

The rationale behind selecting only a subset of features is to decorrelate trees, meaning that if a feature is important, it will not always be used as a top-level split thereby producing unique decision trees. Each tree is built and predictions made by using the random data and features in the method outlined in the Decision Trees subsection at 2.3.3. To make predictions in a classification setting B, decision trees will make a prediction using the features from the bootstrapped data. The most common prediction class is used as the prediction for the forest of trees (which is referred to as the majority vote).

2.4 Random Forest Implementation to Predict Directional Profitability

For this research, annual profitability changes (ROE, ROA, RNOA, CFO, and FCF) are coded as +1 for an increase and -1 for a decrease for individual firms in the dataset described in section 2.2. The features used are described in section 2.6. With data sorted by fiscal year, for a change in a specific profitability measure that feature set (see Tables 2.6, 2.7, 2.8, 2.9, and 2.10) will be partitioned into training and testing sets. The first 90% of observations (fiscal years between 1963-2011) are used to train and build B trees. The remaining 10% of observations (2012-2015) are used to make predictions out of sample.

While building B decision trees, cross-validation, as described in 2.3.1, is employed. The training data will have k folds (where $k = 5$) grouped by fiscal year. Each particular fold will consist of observations for particular firm-years. Each decision tree in the forest will bootstrap from each fold and select random features. The decision tree will then train and build a tree based on the methodology outlined in 2.3.3. The random forest of trees will make predictions on the validation sets with the process outlined in section 2.3.4. Finally, the trained random forest will make predictions out-of-sample on observations from the test set.

2.5 Random Walk Benchmark Models

The random walk is a model of a stochastic process. Campbell et al. (1997) (hereafter CLM) described it by the following equation:

$$P_t = \mu + P_{t-1} + \epsilon_t \tag{2.2}$$

In Equation 2.2, P_t is the value of some state variable at time t, μ is a drift term that we will assume is 0, and ϵ_t is a disturbance, or increment. CLM note that a common assumption is that the disturbance term is independent and identically distributed, with mean zero and variance σ^2, i.e. $\epsilon_t \sim IID(o, \sigma^2)$.

The random walk is a mathematical description of a process, not a forecasting tool. However, some papers (i.e. Bradshaw et al. (2012)) that wish to use the random walk as a benchmark for their results treat it as a forecasting tool by taking the expected value of both sides of Equation 2.2. That yields the following:

$$E[P_{t+1}] = P_t \Leftrightarrow E[P_{t+1} - P_t] = 0 \tag{2.3}$$

These papers reason that since the expected value of P_{t+1} is P_t, the best forecast of P_{t+1} is also P_t. However, the random walk does not actually predict P_{t+1}. It simply states that the expected magnitude of change in P at any time is zero. In other words, if you observe changes in P over a long time series, the average change will be zero.

To be consistent with prior literature, we treat the random walk model as a forecasting tool. However, our work introduces a complication. Prior literature forecasts the magnitude of some variable. By contrast, we forecast the direction of change in our variables. Applying the random walk to our setting, therefore, requires some additional steps. CLM notes that the expected value of ϵ_t is zero, but without additional assumptions, we cannot know the proportion of increases and decreases in ϵ_t.

2.5.1 Random Walk: Benchmark 1

Rearranging Equation 2.2 and setting the drift term to zero yields $P_t - P_{t-1} = \epsilon_t$. Thus, the change in profitability is exactly equal to the disturbance term. That implies that the sign of the change in profitability is equal to the sign of the disturbance term. By assumption, $\epsilon_t \sim IID(0, \sigma^2)$, and CLM (p.32) state that "perhaps the most common distributional assumption for the innovations or increments ϵ_t is normality." Normality implies a 50-50 chance of an up or down movement since the normal distribution is symmetric. Therefore, our first random walk prediction should be that profitability is equally likely to increase or decrease.

Under the assumption that the error term is equally likely to increase or decrease, we find that the random walk model has an expected accuracy of 50%. To see this, assume the testing subsample contains N observations, and $0 \leq p \leq 1$ is the fraction of increases. Then p of those observations increases, and $(1-p)N$ decreases. For any observation, the random walk will predict an increase 50% of the time and a decrease 50% of the time. Thus, of the p increases, on average $0.5pN$ will be classified accurately. Similarly, of the $(1-p)N$ decreases, $0.5(1-p)N$ will be classified accurately. For the overall sample, the fraction that will be accurately classified is:

$$Accuracy = \frac{0.5pN + 0.5(1-p)N}{pN + (1-p)N} = \frac{1}{2}$$

Note that the classification accuracy of this interpretation of the random walk is invariant to the fraction of increases in the data. For any value of p, if we assume increases and decreases are equally likely, this random walk model will accurately predict on average the outcome 50% of the time.

2.5.2 Random Walk: Non-normal Error Term

The random walk model requires that the error term has zero mean and finite variance, $\epsilon_t \sim I(0, \sigma^2)$. This restriction does not imply normality of the error term: ϵ_t could have any arbitrary probability of increase as long as its expected value is zero. For example, if $\frac{2}{3}$ of the time ϵ_t increases by 1, and $\frac{1}{3}$ of the time it decreases by 2, it has mean zero. A distributional assumption other than normality may be appropriate if the actual fraction of increases in the data is not 50%.

Anand et al. (2019) showed in Appendix B that even if we assume ϵ_t will increase 40%-60% of the time, the classification accuracy ranges from 48%-52% which is very close to the 50% by assuming a normally distributed error term. Table 2.5 shows that, for most measures in most years, the actual fraction of increases in our sample are very close to 50%. Even in the most extreme case (in 2013, CFO and FCF increase 43.3% and 42.5% of the time).

2.5.3 Random Walk: Benchmark 2

As an alternative interpretation of applying the random walk model to forecasting, consider the following equations:

$$P_t = P_{t-1} + \epsilon \tag{2.4}$$

$$P_{t-1} = P_{t-2} + \epsilon_{t-1} \tag{2.5}$$

Subtracting Equation 2.5 from Equation 2.4 yields:

$$\begin{aligned}(P_t - P_{t-1}) &= (P_{t-1} - P_{t-2}) + (\epsilon_t - \epsilon_{t-1}) \\ \Delta P_t &= \Delta P_{t-1} + (\epsilon_t - \epsilon_{t-1}) \qquad (2.6)\end{aligned}$$

If we take the expected value of both sides, Equation 2.6 reduces to $\Delta P_t = \Delta P_{t-1}$ since, by assumption, $E[\epsilon_t] = 0, \forall t$. Under this interpretation, in expectation, an increase in a profitability measure in period $t-1$ will persist into period t. Thus, our second random walk prediction is that the sign of the change in a measure in one period will persist into the subsequent period.

2.6 Comparative Analyses

To answer the research objective, we conduct comparative analyses between the benchmark random walk models (described in 2.5) and the random forest models (described in 2.3.4). We employ multiple random forests to predict specific profitability measures and add additional features to each random forest for each subsequent analysis. These analyses provide insight as to whether specific features are informative and yield more accurate estimates.

We perform the analyses in two phases. In the first phase, we test whether Random Forests performs better with winsorized or unwinsorized data [1]. In the second phase, we employ multiple random forests to learn the incremental value of additional variables used in fundamental analysis.

2.6.1 Phase 1: Effect of Winsorization

The purpose of phase 1 is to test the effect of winsorization on predictive accuracy. Our prior is that winsorization will not affect the predictive accuracy of random forests. In phase 1, we use a minimal information set in which each firm-year observation consists of the firm identifier (PERMNO), fiscal year, the contemporaneous value of a measure (i.e., ROE), and the label or change that occurs in the subsequent year (i.e., increase or decrease). We create three random forests for this phase. We train the first random forest on un-winsorized data and refer to this as analysis 1-1. For analysis 1-2, we train the second random forest on data where the measure is winsorized at 1% on each side. For analysis 1-3 we repeat the same methodology as analysis 1-2 where the measure is winsorized at 2.5% on each side.

[1] See Dixon and Yuen (1974) for a detailed explanation and review of winsorization.

2.6.2 Phase 2: Effect of Adding Additional Fundamental Information

In phase 2, we progressively add information about each firm's industry. We begin with analysis 2-1, and add SIC codes as a feature. Analysis 2-2 adds industry information to the feature set. For each firm-year observation in analysis 2-2 we add the industry mean, median, and standard deviation for the measure. For example, consider predicting the change in annual RNOA for IBM during the fiscal year 2000. The industry mean, median, and standard deviation of RNOA for IBM's Industry are used as features during the year 2000. The firm's 4-digit SIC code determines the industry.

Analysis 2-3 repeats analysis 2-2 while including the lag of the profitability measure. For example, if the measure is ROE, then for IBM in the year 2000 its ROE from 1999 is included as a feature in addition to other features from analysis 2-2. For the final analysis 2-4, we repeat analysis 2-3 while including the 2nd lag of the profitability measure. Following the IBM example for analysis 2-3, its ROE from 1998 is included as a feature.

2.7 Results

2.7.1 Phase 1 Results and Discussion

Phase 1 tests the predictive power of random forests with minimal information. It also tests the effect of winsorizing the input data, a common practice in accounting research. In phase 1, we create three random forests per profitability measure and use a minimal set of features. For each random forest, the only predictors (features) in each observation are a firm identifier (PERMNO), fiscal year, and the contemporaneous value of the profitability measure. The label is a categorical variable that indicates whether profitability increased or decreased in the subsequent year. The input data's non-categorical measures are winsorized at three levels, 0%, 1%, and 2.5% on each side. Analysis 1-1 uses a random forest to make predictions on the non-winsorized data, 1-2 uses the 1% winsorized data, and 1-3 uses the 2.5% winsorized data.

We begin with the default values for model parameters in sci-kit learn version 0.22 (see Pedregosa et al. (2011)). We varied the number of trees used in a forest for each measure to predict the change in profitability. We found for all measures that there was no significant improvement in performance after using 30 decision trees in a random forest, see Figure 2.2. In Figure 2.2, all subplots share the same x-axis, which represents the number of trees used in a forest. The y-axis for each subplot represents the accuracy score for a particular measure.

In each subplot, the dashed line with the circle markers represents the mean training scores during cross-validation. The shaded region surrounding the dashed line with circle markers represents the variance of the training accuracy scores during cross-validation. This shaded region is not visible because the variance of the $k-1$ training scores is very small. The solid line with star markers represents the mean cross-validation accuracy score, and the shaded-in region represents the variance of the cross-validation scores for each fold.

Given the large gap between the mean training and cross-validation accuracy scores, it is probable that the random forests are over-fitting the data. With the current model parameters (while varying the number of trees used in the random forest), the model is learning information from the training data that is noise. The current parameters impede the model's ability to generalize to the cross-validation data well. To decrease the model's variance (or flexibility), we vary the depth of each decision tree in the forest.

Figure 2.3 shows the mean training and cross-validation scores for a random forest with 30 trees as the maximum depth is varied. As the depth increases, the model becomes more flexible. Across all profitability measures, one can see that the model can learn from the training data fairly well with high training accuracy scores. However, the highly flexible models cannot generalize well to the cross-validation data and perform worst than other models with a smaller maximum depth. Thus, we select the maximum depth that yields a model with relatively low variance and bias for each profitability measure. We forgo displaying similar plots for analyses 1-2 and 1-3 because the effect and accuracy scores are similar.

Table 2.11 shows the model maximum depth selected for analyses 1-1, 1-2, and 1-3. Tables 2.12, 2.13, 2.14, 2.15, and 2.16 show the mean training, mean cross-validation, and testing scores for the ROE, ROA, RNOA, CFO, and FCF measures respectively. In addition to showing the cross-validation scores, we show the random walk model performance on the data, and the feature importance of each variable.

The results show that the random forest can outperform the random walk models for each measure in predicting the change of profitability with a minimum set of features. In addition to this, when excluding the ROE measure we find no significant difference in mean testing accuracies between winsorized and non-winsorized data. Analysis 1-1 (no winsorization) and 1-3 (2.5% per side winsorization) had a mean test accuracy of 55.2%; however, analysis 1-2 (1% per side winsorization) had a mean test accuracy of 56.8%. The remaining analyses for the other profitability measures have mean testing accuracies within 1% of each other. These results provide evidence that winsorization has a negligible effect on performance out of sample.

To summarize, traditional approaches have struggled in the past to out-perform a random walk model. With a minimum amount of features (company identifier, profitability measure, and fiscal year), we find that a random forest model is able to out-perform a random walk model in predicting annual profitability changes. In addition, we find that winsorizing the data (which is a traditional method used to handle outliers in financial accounting) has a negligible effect on the performance of a random forest model.

2.7.2 Phase 2 Results and Discussion

Given that there is no significant effect of winsorization on the random forest model's performance, we forgo using the winsorized datasets. We create additional random forests where each successive forest incrementally adds features. In analysis 2-1, we include the firm's industry as a feature. Analysis 2-2 includes descriptive statistics about the firm's industry at a given year. The specific statistics used are mean, median, and standard deviation. Analysis 2-3 adds the previous lag value of the profitability measure (i.e., the profitability measure at $t-1$), and analysis 2-4 includes the two previous lag values of the profitability measure (i.e., $t-1, t-2$). Tables 2.17, 2.18, 2.19, 2.20, and 2.21 report the results from the second set of analyses.

Similar to the first set of analyses, all random forest models outperform the random walk models. Comparing the second set of analyses to analysis 1-1 (non-winsorized data with a minimum set of features), we find mixed results for each profitability measure. For CFO and FCF, we find improvements across all sets of analyses. However, for ROE, ROA, and RNOA there is little to no improvement from analysis 1-1. We find that CFO and FCF random forest models utilize the feature CFO_t and FCF_t the most throughout all analyses. This finding provides evidence that the best feature to predict the future directional change of a measure is the current value of the measure for CFO and FCF.

The RNOA, ROA, and ROE analyses still utilize $RNOA_t$, ROA_t, and ROE_t respectively more than any other feature; however, they utilize it significantly less than the random forest models that predict CFO and FCF. For example, in analysis 2-3 for RNOA, ROA, ROE, CFO, and FCF the previous lagged measures are the second most used by the model. The feature importance of CFO and FCF are 69.2% and 78.5% more than the lagged value (CFO_{t-1} and FCF_{t-1}). However, the ROE, ROA, and RNOA measures use their respective current values at t 45.7%, 50.3%, and 45.8% more than the lagged values. In addition to this, we find that ROE's performance improves more than ROA and RNOA. This finding further provides evidence that the most information about future profitability changes is in the profitability measure's current value.

2.8 Conclusion and Future Work

In this chapter, we explored the utility of random forest trees in a supervised learning paradigm to predict the annual direction of profitability for firms with minimal information. This study's motivation stems from the fact that the accounting literature shows that traditional regression methods cannot produce models superior to random walk models when predicting out of sample. We also explore the advantages of following a supervised learning paradigm to predict directional change. For example, random forests are insensitive to multicollinearity, and properly utilizing the supervised paradigm discovers a functional form that generalizes well to new data.

We implement and train random forests in a classification setting using a large sample of US firms over the period 1963-2011. We generate out-of-sample predictions of directional changes (increases or decreases) in five profitability measures: return on equity (ROE), return on assets (ROA), return on net operating assets (RNOA), cash flow from operations (CFO), and free cash flow (FCF) from 2011-2016. We found that the classification accuracy for each measure outperformed the random walk models. The accuracy does not tend to decline with a four-year forecast horizon significantly for the out of sample predictions. In addition to these findings, models with a strong reliance on a profitability measure's current value provided more accurate estimates of the future directional change in profitability.

We can improve these results further by selecting a model that yields better performance but offers less interpretability. Given that we focus on a minimal set of features in this research, we can incorporate additional features for additional performance gains. Utilizing more data, we can extend the methodology outlined in this paper to a regression setting and make predictions about the magnitudes of profitability.

The results shown in the paper are promising since we outperform a random walk model for all of the profitability measures used in this study. The methods used in this chapter are robust to econometric issues such as multicollinearity and outliers. When used correctly, the methodology outlined in this paper can produce models that generalize well to out of sample data. Machine learning has simplified and solved problems in many aspects of our lives. The results show the benefit that it can have on predicting the profitability of firms in a financial accounting research setting.

2.9 Figures and Tables

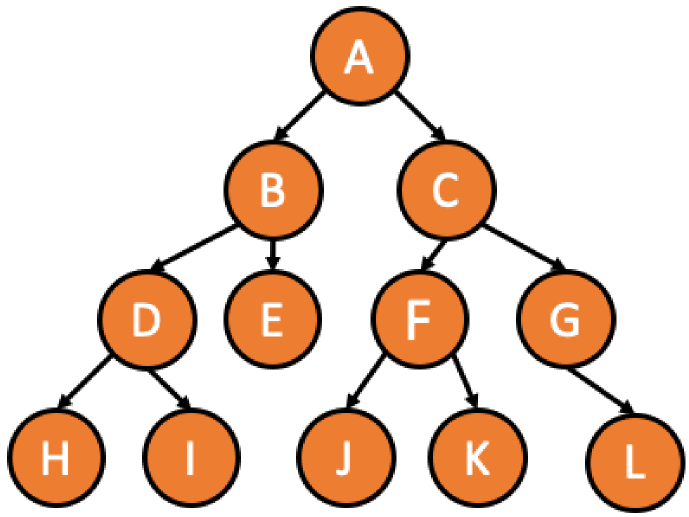

Figure 2.1: This figure shows an example of a Decision Tree. This is essentially a graph with no cycles. Each circle represents a node (the label of the node is a letter inside of the node) and the arrows represent directional edges (branches) to another node. Node A is the root node of this tree and is where the decision tree begins. Potential outcomes from the root node lead to different states that are represented in nodes B and C. B and C are the children of A. Nodes B and C have children and their children has children. Nodes H, I, E, J, K, and L are the leaf nodes of this tree because they have no children.

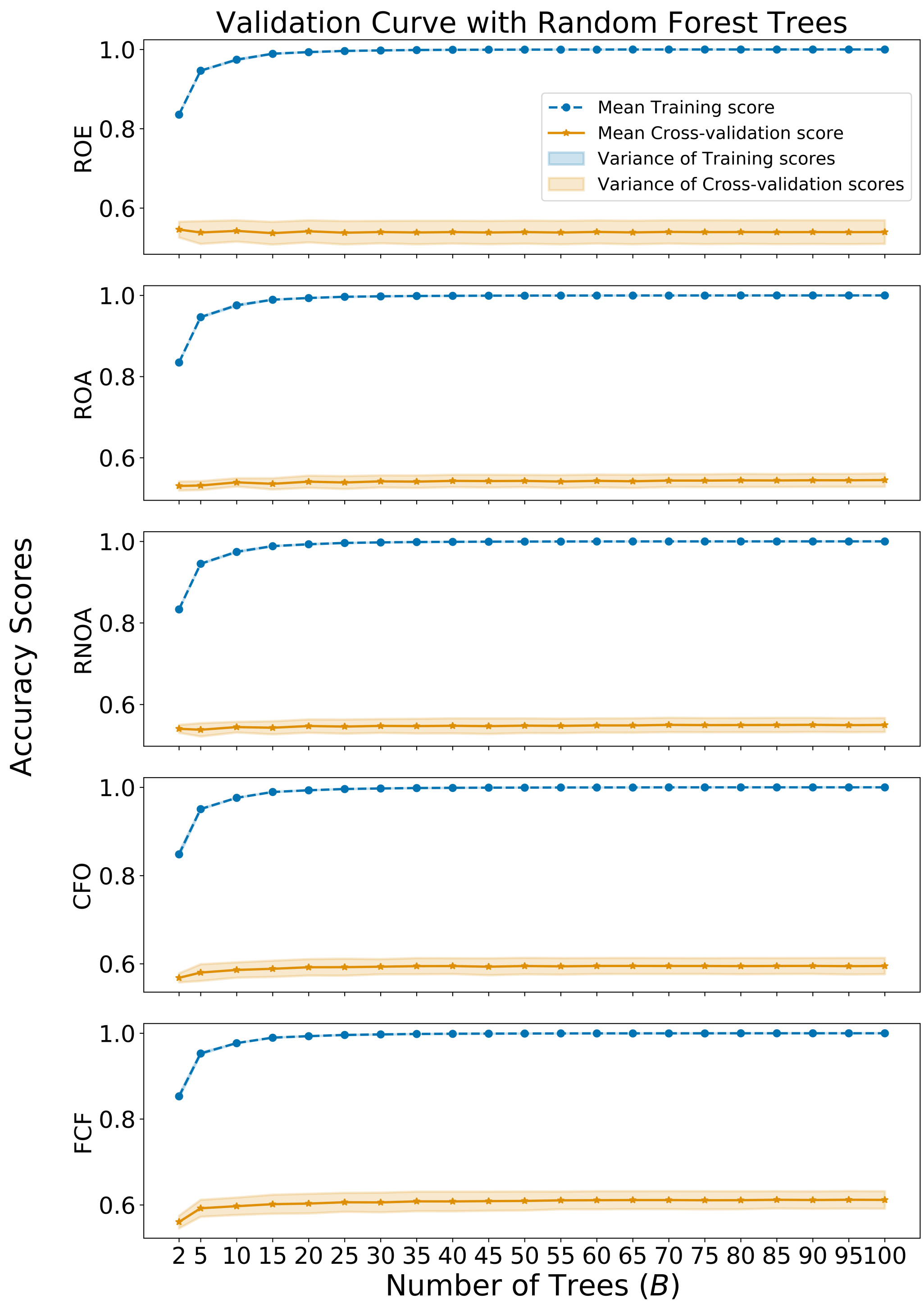

Figure 2.2: This figure shows validation curves for each measure in Analysis 1-1, varying the number of trees.

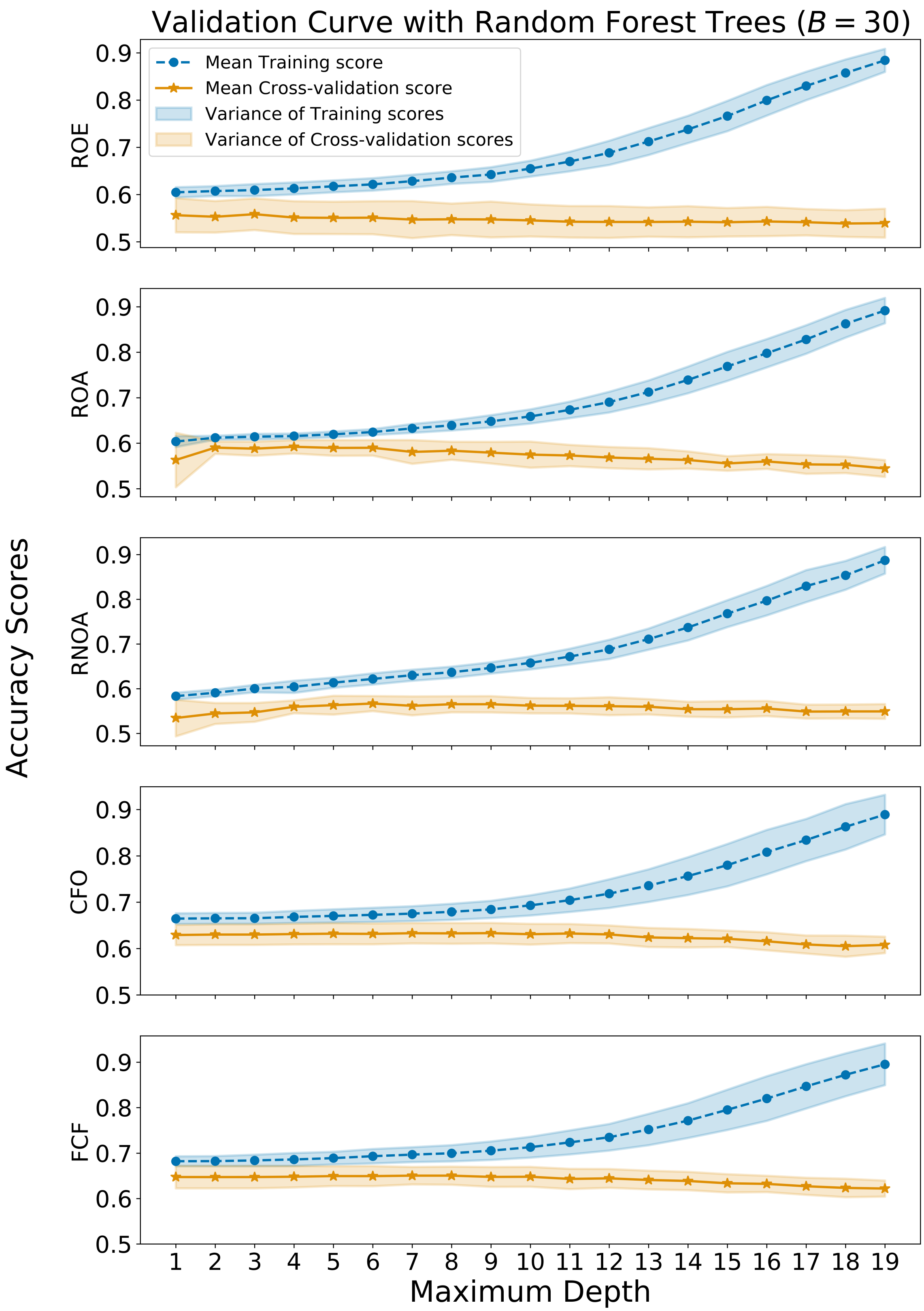

Figure 2.3: This figure shows the validation curves for each measure in Analysis 1-1 varying the max depth using 30 trees in a forest.

Variable	**Code**
Current Assets – Total	ACT
Assets – Total	AT
Capital Expenditures	CAPX
Common Equity – Total	CEQ
Cash and Short-Term Investments	CHE
Debt in Current Liabilities (short-term debt)	DLC
Long-Term Debt – Total	DLTT
Depreciation and Amortization	DP
Data year – fiscal	FYEAR
Earnings (income before extraordinary items)	IB
Earnings (income before extraordinary items, from statement of cash flows)	IBC
Interest and Related Income – Total	IDIT
Short-term investments – Total	IVST
Current Liabilities – Total	LCT
Liabilities – Total	LT
Operating Activities – Net Cash Flow	OANCF
Operating Income before Depreciation	OIBDP
CRSP permanent number	PERMNO
Property, Plant, and Equipment – Total (Net)	PPENT
Standard Industry Classification	SIC
Income Taxes Payable	TXP
Extraordinary Items and Discontinued Operations (Statement of Cash Flows)	XIDOC
Interest Expense	XINT

Table 2.1: Table showing the Computstat variables used in this study.

Variable	Definition
ROE_t	$\frac{IB_t}{0.5(CEQ_t+CEQ_{t-1})}$
ROA_t	$\frac{IB_t+XINT_t}{0.5(AT_t+AT_{t-1})}$
$RNOA_t$	$\frac{OI_t}{0.5\times(NOA_t+NOA_{t-1})}$
CFO_t	$\frac{OANCF_t}{0.5(AT_t+AT_{t-1})}$
FCF_t	$\frac{OANCF_t-CAPX_t}{0.5(AT_t+AT_{t-1})}$

Table 2.2: Table showing the profitability measure definitions.

Variable	Count	Mean	Std	Min	1%	25%	50%	75%	99%	Max
AT	167,718.00	3,344.65	37,775.67	0	2.08	32.21	139.24	754.55	45,009.92	2,573,126.00
CAPX	167,213.00	103.64	665.28	-401.61	0	0.99	5.29	30.46	1,833.88	37,985.00
CEQ	167,718.00	733.70	4,746.54	-59,640.00	-46.55	13.94	59.19	272.44	11,881.49	255,550.00
DP	166,456.00	73.70	465.94	-7.91	0.02	0.95	4.20	23.01	1,250.90	22,016.00
IB	167,718.00	85.78	859.57	-99,289.00	-227.97	-0.32	3.38	25.86	1,844.83	53,394.00
OANCF	167,718.00	181.91	1,590.52	-110,560.00	-81.17	0.10	6.70	51.69	3,353.03	129,731.00
OIBDP	167,026.00	264.67	1,742.21	-76,735.00	-53.35	1.57	13.19	81.83	4,516.75	81,730.00
PPENT	166,826.00	620.94	3,822.74	0	0.05	4.86	24.81	154.47	12,305.75	252,668.00
XINT	167,718.00	49.38	639.00	-0.88	0	0.12	1.28	10.41	648.83	57,302.00

Table 2.3: This table shows the descriptive statistics for the Compustat variables in the sample.

Variable	Count	Mean	Std	Min	1%	25%	50%	75%	99%	max
ROE	160,653.00	0.00	0.50	-4.47	-2.52	-0.01	0.10	0.17	1.10	2.97
ROA	160,653.00	0.02	0.18	-1.33	-0.85	0.01	0.06	0.10	0.26	0.35
RNOA	160,653.00	0.07	1.07	-12.03	-4.00	0.02	0.10	0.17	3.64	10.66
CFO	160,653.00	0.04	0.16	-0.91	-0.68	0.01	0.07	0.12	0.34	0.41
FCF	160,653.00	-0.03	0.17	-1.07	-0.76	-0.07	0.01	0.06	0.27	0.34

Table 2.4: This table shows the descriptive statistics for profitability measures.

Fisical Year	ROE	ROA	RNOA	CFO	FCF
2012	47.6%	48.4%	47.9%	49.4%	51.9%
2013	48.9%	49.7%	47.6%	43.3%	42.5%
2014	46.6%	45.5%	44.7%	49.1%	51.5%
2015	48.4%	51.0%	50.2%	50.8%	53.6%

Table 2.5: This table shows the percentage increases of annual profitability in our test sample.

	ROE	Ind ROE Mean	Ind ROE Median	Ind ROE Std	ROE_L1	ROE_L2
ROE	1.00	0.20	0.39	-0.11	0.53	0.39
Ind ROE Mean		1.00	0.35	-0.63	0.14	0.12
Ind ROE Median			1.00	-0.19	0.33	0.29
Ind ROE Std				1.00	-0.12	-0.10
ROE_L1						0.54
ROE_L2						1.00

Table 2.6: This table shows the Pearson correlations for ROE features in our sample.

	ROA	Ind ROA Mean	Ind ROA Median	Ind ROA Std	ROA_L1	ROA_L2
ROA	1.00	0.53	0.49	-0.42	0.74	0.63
Ind ROA Mean		1.00	0.89	-0.84	0.48	0.43
Ind ROA Median			1.00	-0.60	0.45	0.42
Ind ROA Std				1.00	-0.38	-0.36
ROA_L1					1.00	0.74
ROA_L2						1.00

Table 2.7: This table shows the Pearson correlations for ROA features in our sample.

	RNOA	Ind RNOA Mean	Ind RNOA Median	Ind RNOA Std	RNOA_L1	RNOA_L2
RNOA	1.00	0.10	0.24	0.01	0.36	0.22
Ind RNOA Mean		1.00	0.25	0.26	0.06	0.03
Ind RNOA Median			1.00	0.09	0.18	0.14
Ind RNOA Std				1.00	0.01	0.01
RNOA_L1					1.00	0.36
RNOA_L2						1.00

Table 2.8: This table shows the Pearson correlations for RNOA features in our sample.

	FCF	Ind FCF Mean	Ind FCF Median	Ind FCF Std	FCF_L1	FCF_L2
FCF	1.00	0.49	0.45	-0.31	0.62	0.53
Ind FCF Mean		1.00	0.89	-0.71	0.39	0.36
Ind FCF Median			1.00	-0.46	0.37	0.33
Ind FCF Std				1.00	-0.28	-0.26
FCF_L1					1.00	0.61
FCF_L2						1.00

Table 2.9: This table shows the Pearson correlations for FCF features in our sample.

	CFO	Ind CFO Mean	Ind CFO Median	Ind CFO Std	CFO_L1	CFO_L2
CFO	1.00	0.50	0.47	-0.32	0.67	0.60
Ind CFO Mean		1.00	0.90	-0.70	0.42	0.40
Ind CFO Median			1.00	-0.47	0.40	0.38
Ind CFO Std				1.00	-0.29	-0.28
CFO_L1					1.00	0.66
CFO_L2						1.00

Table 2.10: This table shows the Pearson correlations for CFO features in our sample.

Analysis (Degree of Winsorization)	**1-1 (None)**	**1-2 (1% per side)**	**1-3 (2.5% per side)**
ROE	3	2	3
ROA	2	2	2
RNOA	4	2	4
CFO	2	2	2
FCF	2	2	2

Table 2.11: This table shows the maximum depth selected for each analysis and measure combination with $B = 30$.

Return on Equity (ROE)

	Analysis (Degree of Winsorization)		
	1-1 (None)	**1-2 (1% per side)**	**1-3 (2.5% per side)**
Mean Training Accuracy	61.1%	61.4%	61.1%
Mean Cross Validation Accuracy	55.5%	58.5%	55.5%
Mean Testing Accuracy	55.2%	56.8%	55.2%
Random Walk 1	50%		
Random Walk 2	48%		
Test Scores by Year			
2012	57.5%	60.4%	57.5%
2013	53.8%	57.1%	53.8%
2014	55.2%	54.2%	55.2%
2015	54.1%	55.4%	54.1%
Feature Importance			
PERMNO	2.7%	2.7%	2.7%
Fisical Year	19%	30%	19%
ROE	78.2%	67.2%	78.2%

Table 2.12: This table shows the results of analyses 1-1, 1-2, and 1-3 results for the profitability measure ROE.

Return on Equity (ROA)

	Analysis (Degree of Winsorization)		
	1-1 (None)	**1-2 (1% per side)**	**1-3 (2.5% per side)**
Mean Training Accuracy	61.1%	61.2%	61.1%
Mean Cross Validation Accuracy	57.9%	58.1%	57.9%
Mean Testing Accuracy	58.5%	58.5%	58.5%
Random Walk 1	50%		
Random Walk 2	48.1%		
Test Scores by Year			
2012	60.6%	60.4%	60.6%
2013	57.6%	58.2%	57.6%
2014	58.9%	58%	58.9%
2015	56.8%	57.2%	56.8%
Feature Importance			
PERMNO	2.6%	2.5%	2.6%
Fisical Year	32.2%	32.2%	32.2%
ROA	65.2%	65.2%	65.2%

Table 2.13: This table shows the results of analyses 1-1, 1-2, and 1-3 results for the profitability measure ROA.

	Return on Net Operating Assets (RNOA)		
	Analysis (Degree of Winsorization)		
	1-1 (None)	**1-2 (1% per side)**	**1-3 (2.5% per side)**
Mean Training Accuracy	61.2%	62.2%	61.2%
Mean Cross Validation Accuracy	56.5%	59.8%	56.5%
Mean Testing Accuracy	54.1%	53.3%	54.1%
Random Walk 1	50%		
Random Walk 2	47.6%		
Test Scores by Year			
2012	59%	55.6%	59%
2013	50%	53.3%	50%
2014	52.7%	50.1%	53%
2015	54.1%	54.3%	54.2%
Feature Importance			
PERMNO	3.2%	1.8%	3.3%
Fisical Year	16.2%	28.4%	16.2%
RNOA	80.5%	69.8%	80.5%

Table 2.14: This table shows the results of analyses 1-1, 1-2, and 1-3 results for the profitability measure RNOA.

Cash Flow from Operations (CFO)

	Analysis (Degree of Winsorization)		
	1-1 (None)	**1-2 (1% per side)**	**1-3 (2.5% per side)**
Mean Training Accuracy	66.5%	66.5%	66.5%
Mean Cross Validation Accuracy	63%	63.1%	0.63%
Mean Testing Accuracy	58.7%	58.5%	58.6%
Random Walk 1	50%		
Random Walk 2	38.2%		
Test Scores by Year			
2012	62.5%	62.2%	62.5%
2013	55.2%	54.9%	55.2%
2014	59%	58.9%	59%
2015	57.9%	0.58%	57.8%
Feature Importance			
PERMNO	7.6%	7.6%	7.6%
Fisical Year	19.6%	19.6%	19.5%
CFO	72.8%	72.8%	72.8%

Table 2.15: This table shows the results of analyses 1-1, 1-2, and 1-3 results for the profitability measure CFO.

	Free Cash Flow (FCF)		
	Analysis (Degree of Winsorization)		
	1-1 (None)	**1-2 (1% per side)**	**1-3 (2.5% per side)**
Mean Training Accuracy	68.3%	68.4%	68.3%
Mean Cross Validation Accuracy	64.7%	64.8%	64.7%
Mean Testing Accuracy	59.1%	59.1%	59.1%
Random Walk 1	50%		
Random Walk 2	39%		
Test Scores by Year			
2012	63.2%	63.2%	63%
2013	55.6%	55.7%	55.6%
2014	60%	60%	59.8%
2015	57.9%	57.6%	57.9%
Feature Importance			
PERMNO	3.7%	3.7%	3.7%
Fisical Year	20.4%	20.4%	20.5%
FCF	75.8%	75.8%	75.8%

Table 2.16: This table shows the results of analyses 1-1, 1-2, and 1-3 results for the profitability measure FCF.

	Return on Equity (ROE)			
	Analysis			
	2-1	**2-2**	**2-3**	**2-4**
Mean Train Accuracy	60.5%	62.4%	62.4%	62.5%
Mean Cross Validation Accuracy	55.4%	56.0%	57.2%	56.4%
Mean Test Accuracy	55.0%	56.6%	56.8%	56.5%
Random Walk 1	50%			
Random Walk 2	48%			
Test Scores by Year				
2012	57.2%	58.2%	60.0%	60.0%
2013	53.2%	55.4%	54.8%	55.3%
2014	55.4%	57.4%	56.5%	54.2%
2015	53.9%	55.5%	55.9%	56.4%
Feature Importance				
PERMNO	10.8%	2.8%	2.8%	2.1%
FYEAR	32.8%	13.5%	8.8%	10.8%
ROE	50.1%	68.7%	61.1%	52.5%
SIC	6.3%	1.2%	1.0%	1.0%
Industry ROE Mean		5.2%	5.0%	4.3%
Industry ROE Median		5.3%	3.3%	3.0%
Industry ROE std.		3.3%	2.6%	2.5%
ROE_{t-1}			15.4%	18.7%
ROE_{t-2}				5.4%

Table 2.17: This table shows the results of analyses 2-1, 2-2, 2-3, and 2-4 for the profitability measure ROE.

Return on Assets (ROA)

	Analysis			
	2-1	**2-2**	**2-3**	**2-4**
Mean Training Accuracy	62.2%	61.9%	61.4%	61.4%
Mean Cross Validation Accuracy	59.0%	58.3%	58.7%	57.3%
Mean Testing Accuracy	58.6%	58.1%	58.1%	58.3%
Random Walk 1	50%			
Random Walk 2	48.1%			
Test Scores by Year				
2012	60.8%	60.7%	60.4%	60.5%
2013	58.2%	58.5%	57.8%	57.7%
2014	58.5%	57.6%	58.3%	58.4%
2015	56.8%	55.7%	55.9%	56.8%
Feature Importance				
PERMNO	4.0%	1.1%	0.8%	0.7%
FYEAR	15.9%	13.3%	5.0%	7.4%
ROA	78.3%	70.1%	66.7%	56.0%
SIC	1.8%	1.1%	0.8%	0.7%
Industry ROA Mean		5.0%	3.6%	3.5%
Industry ROA Median		7.4%	5.6%	4.0%
Industry ROA std.		2.1%	1.0%	0.8%
ROA_{t-1}			16.5%	21.3%
ROA_{t-2}				5.7%

Table 2.18: This table shows the results of analyses 2-1, 2-2, 2-3, and 2-4 for the profitability measure ROA.

Return on Net Operating Assets (RNOA)

	Analysis			
	2-1	**2-2**	**2-3**	**2-4**
Mean Train Accuracy	60.8%	60.8%	61.9%	61.9%
Mean Cross Validation Accuracy	55.9%	56.3%	57.1%	56.6%
Mean Testing Accuracy	52.4%	52.7%	52.5%	52.8%
Random Walk 1	50%			
Random Walk 2	47.6%			
Test Scores by Year				
2012	59.0%	57.0%	57.4%	57.8%
2013	49.2%	49.8%	52.2%	52.1%
2014	47.5%	49.6%	46.9%	47.8%
2015	53.8%	54.4%	53.5%	53.4%
Feature Importance				
PERMNO	3.6%	1.0%	1.4%	0.9%
FYEAR	15.3%	12.2%	6.7%	7.8%
$RNOA$	79.6%	63.0%	61.7%	56.6%
SIC	1.5%	0.6%	0.9%	0.8%
Industry RNOA Mean		6.7%	4.5%	4.1%
Industry RNOA Median		14.3%	6.2%	6.8%
Industry RNOA std.		2.1%	2.7%	2.2%
$RNOA_{t-1}$			15.9%	16.2%
$RNOA_{t-2}$				4.5%

Table 2.19: This table shows the results of analyses 2-1, 2-2, 2-3, and 2-4 for the profitability measure RNOA.

Cash Flow from Operations (CFO)

	Analysis			
	2-1	**2-2**	**2-3**	**2-4**
Mean Training Accuracy	67.5%	67.8%	68.8%	69.2%
Mean Cross Validation Accuracy	63.9%	64.0%	64.5%	64.6%
Mean Testing Accuracy	60.2%	59.8%	60.2%	60.1%
Random Walk 1	50%			
Random Walk 2	38.2%			
Test Scores by Year				
2012	62.6%	63.2%	64.1%	63.2%
2013	62.6%	58.5%	57.5%	58.1%
2014	59.0%	60.1%	60.4%	60.3%
2015	56.5%	57.3%	58.7%	58.7%
Feature Importance				
PERMNO	2.4%	1.6%	1.8%	1.4%
FYEAR	4.9%	3.5%	3.2%	2.8%
CFO	91.1%	82.1%	75.8%	75.4%
SIC	1.6%	1.7%	1.6%	1.4%
Industry CFO Mean		3.8%	4.5%	3.7%
Industry CFO Median		5.4%	4.4%	4.5%
Industry CFO std.		1.9%	2.1%	1.9%
CFO_{t-1}			6.6%	5.1%
CFO_{t-2}				3.8%

Table 2.20: This table shows the results of analyses 2-1, 2-2, 2-3, and 2-4 for the profitability measure CFO.

Free Cash Flow (FCF)

	Analysis			
	2-1	**2-2**	**2-3**	**2-4**
Mean Training Accuracy	69.2%	69.5%	69.5%	69.6%
Mean Cross Validation Accuracy	65.2%	65.2%	65.5%	65.4%
Mean Testing Accuracy	60.8%	60.7%	61.3%	61.7%
Random Walk 1	50%			
Random Walk 2	39%			
Test Scores by Year				
2012	62.3%	64.1%	62.2%	63.9%
2013	63.4%	60.4%	63.9%	63.3%
2014	60.6%	60.6%	61.1%	61.5%
2015	56.9%	57.8%	57.9%	58.2%
Feature Importance				
PERMNO	2.1%	1.0%	1.0%	0.9%
FYEAR	5.0%	4.1%	2.5%	2.4%
FCF	91.5%	82.1%	82.2%	81.1%
SIC	1.3%	1.1%	1.0%	0.8%
Industry FCF Mean		3.9%	4.4%	2.8%
Industry FCF Median		6.0%	3.8%	5.0%
Industry FCF std.		1.9%	1.6%	1.6%
FCF_{t-1}			3.6%	3.2%
FCF_{t-2}				2.1%

Table 2.21: This table shows the results of analyses 2-1, 2-2, 2-3, and 2-4 results for the profitability measure FCF.

Chapter 3

Information Transfer Estimation on Large Data

3.1 Introduction

Transfer entropy (TE) is an information measure that quantifies the transfer of information between processes evolving in time (see Chapter 1.3.1). Transfer entropy has many potential applications in neuroscience, social media, and financial markets. Prior TE work concerning financial applications has typically used long windows in their research. For example, Marschinski and Kantz (2002) measured information transfer between two financial time series (the DAX stock index and the Dow Jones) to determine to what extent one index determined the other's behavior. The data used by Marschinski and Kantz (2002) sampled data every minute between May 2000 and June 2001; after cleaning the data, 63,867 observations were used in Marschinski and Kantz (2002)'s study. More recently Sandoval (2014) used daily price data from 2003 to 2012 to detect causal relationships between 197 largest firms (globally) with Transfer Entropy.

Given the assumption in Chapter 1.3 that information is reflected in price and in an efficient market changes rapidly (if not instantaneously), then slower frequencies of observations (or long time windows) such as monthly, daily, hourly, or even by the minute may be

This chapter contains material from the following publication:

- K. M. Ikegwu, J. Trauger, J. McMullin, and R. J. Brunner. Pyif: A fast and light weight implementation to estimate bivariate transfer entropy for big data. In *2020 SoutheastCon*, pages 1–6, 2020. doi: 10.1109/SoutheastCon44009.2020.9249650 insufficient in capturing the price change process. Given that we may not capture the entire price change process with long time windows between each observation, we examine price change with shorter windows. However, shorter time windows will have a finer resolution of data, requiring more observations in a dataset.

In Figure 3.1, we present the number of observations for a univariate dataset over 30 days for varying frequency size. If the frequency of observations (time windows) in 30 days is daily, there are 30 observations in the dataset. If the time windows are hourly, we can expect 720 observations in 30 days, and if the time window between observations occurs on a minute level, then we have 43,200 observations. If we continue these calculations down to the 1-second level, then there are 2,592,000 observations in a 30-day window. Following this trend, we find that existing open-source, TE computation implementations are not suited to estimate information flow for datasets with large numbers of observations. Given the need to examine information transfers over finer time resolutions, we propose a new, open-source, software implementation to estimate TE for large datasets.

3.2 Estimating Transfer Entropy

Prior research has developed a few approaches for estimating TE. There is a requirement to specify the following parameter choices: the time between periods, length of the time series, number of past observations that inform the future observations, and the direction of information transfer in the approaches. The correct approach depends on the underlying data. There are many techniques for estimating mutual information. Khan et al. (2007) explored the utility of different methods for mutual information estimation, and many of the methods are applicable to estimate TE.

3.2.1 Kraskov Estimator

Kraskov et al. (2004) outlined a widely used method to estimate Transfer Entropy by using a k-nearest neighbor algorithm. Note, to estimate entropy:

$$\hat{H}(X) = -\frac{1}{n}\sum_{i=1}^{n} ln(\hat{p}(x_i)) \tag{3.1}$$

Kraskov et al. expanded this definition to estimate entropy to:

$$\hat{H}(X) = -\frac{1}{n}\sum_{i=1}^{n}\psi(n_x(i)) - \frac{1}{k} + \psi(n) + ln(c_{d_x}) + \frac{d_x}{n}\sum_{i=1}^{n} ln(\epsilon(i)) \tag{3.2}$$

where n are the number of data points, k is a parameter to select the number of nearest neighbors, d_x is the dimension of x, and c_{d_x} is the volume of the d_x-dimensional unit ball. For two random variables X and Y, let $\frac{\epsilon(i)}{2}$ be the distance between (x_i, y_i) and it's k$^{\text{th}}$ neighbor be denoted by (kx_i, ky_i). Let $\frac{\epsilon_x(i)}{2}$ and $\frac{\epsilon_y(i)}{2}$ be defined as $||x_i - kx_i||$ and $||y_i - ky_i||$ respectively. $n_x(i)$ is the number of points x_j such that $||x_i - x_j|| \leq \epsilon_x(i)/2$, $\psi(x)$ is the digamma function where:

$$\psi(x) = \Gamma(x)^{-1}\frac{d\Gamma(x)}{dx} \tag{3.3}$$

and $\Gamma(x)$ is the ordinary gamma function. Lastly $\psi(1) = -C$ where $C = 0.5772156649$ and is the Euler-Mascheroni constant.

To estimate Joint entropy between X and Y:

$$\hat{H}(X,Y) = -\psi(k) - \frac{1}{k} + \psi(n) + ln(c_{d_x} c_{d_y}) + \frac{d_x + d_y}{n} \sum_{i}^{n} ln(\epsilon(i)) \tag{3.4}$$

where d_y is the dimension of y, and c_{d_y} is the column of the d_y-dimensional unit ball. Using $\hat{H}(X), \hat{H}(Y)$, and $\hat{H}(X,Y)$, Kraskov shows that mutual information can be estimated as:

$$\hat{I}(X,Y) = \psi(k) - \frac{1}{k} - \frac{1}{n} \sum_{i=1}^{n} [\psi(n_x(i)) + \psi(n_y(i))] + \psi(n) \tag{3.5}$$

where $n_y(i)$ is the number of points y_j such that $||y_i - y_j|| \leq \frac{\epsilon_y(i)}{2}$.

The basic idea of mutual information estimation with the Kraskov's estimator is for each observation i to count the number $n_x(i)$ (or $n_y(i)$) of points less than the distance between x_i (or y_i) and the k^{th} neighbor. The distance or $\epsilon(i)$ fluctuates for each observation, consequently, $n_x(i)$ and $n_y(i)$ fluctuate. The counts are averaged over all observations, which results in mutual information as defined in Equation 3.5.

To estimate TE with Kraskov's estimator, recall from Equation 1.15 that TE is expressed as the difference between two mutual information computations. To estimate TE with Equation 1.15, the first mutual information is between Y's future value, Y's lagged values, and the X's lagged values. The second mutual information is the mutual information between X and its lagged values. The difference between the first and second mutual information provide the TE. With an embedding of 1, TE can be estimated:

$$TE_{Y\rightarrow X} = \left[\psi(k) - \frac{1}{k} - \frac{1}{n}\sum_{i=1}^{n}[\psi(n_{xy}^{(t)}(i)) + \psi(n_{xy}^{(t-1)}(i))] + \psi(n)\right] -$$
$$\left[\psi(k) - \frac{1}{k} - \frac{1}{n}\sum_{i=1}^{n}[\psi(n_{x}^{(t)}(i)) + \psi(n_{x}^{(t-1)}(i))] + \psi(n)\right] \tag{3.6}$$

Here n_{xy} is the number of points y_j such that $||y_i - y_j|| \leq \epsilon_y(i)/2$ and $||x_i - y_j|| \leq \epsilon_y(i)/2$.

3.2.2 Additional Estimators

Khan et al. (2007) also explored the utility of Kernel Density Estimation, Edgeworth approximation of differential entropy to calculate Mutual Information, and adaptive partitioning of the XY plane to estimate the joint probability density, which can estimate mutual information. Ultimately Khan et al. (2007) found that a KDE estimator and Kraskov estimator outperform other methods for their ability to capture the dependence structure of random processes. Anderson and McMullin (2018) examined the properties of the Kraskov estimator between equities and their underlying options. They ultimately provided evidence that incorrect parameter choices lead to smaller TE estimates and that Kraskov's method has a downward bias indicating that it underestimates information transfer. Nevertheless, it is relatively insensitive to parameter selection.

3.3 PyIF

PyIF [1] is our proposed, open-source software implementation to estimate bivariate TE on large data. PyIF currently only supports using the Kraskov estimator to estimate TE. PyIF utilizes recent advancements in hardware to parallelize & optimize operations across CPUs and CUDA compatible GPUs (see NVIDIA et al. (2020)). Given the iterative nature

[1] PyIF can freely be downloaded from: `https://github.com/lcdm-uiuc/PyIF`

of Equation 3.6 it is possible to speed up the estimation by performing computations in parallel. In particular, we focus our efforts on the parallelization & optimization across operations to obtain n_x & n_{xy} in Equation 3.6 faster.

PyIF is a Python implementation that utilizes five well-known and actively supported Python libraries: SciPy (see Virtanen et al. (2020)), NumPy (see Harris et al. (2020)), scikit-learn (see Pedregosa et al. (2011)), nose (see Pellerin (2021)), and numba (see Lam et al. (2015)). SciPy is an open-source Python library used for a variety of STEM applications. NumPy is a part of SciPy's ecosystem and is an open-source package that provides convenient ways to perform matrix manipulations and useful linear algebra capabilities. Scikit-learn is a popular open-source library for machine learning, and nose is another open-source library that is useful for testing code to ensure that it will produce the correct outcome. Lastly, numba is a Python tool that can compile Python code to execute multicore CPUs and CUDA-capable GPUs.

PyIF's interface requires one to supply X and Y, two NumPy arrays with Nx1 dimensions. PyIF supports optional arguments such as k, which controls the number of neighbors used in KD-tree nearest neighbor searches, *embedding*, which controls how many lagged periods are in the transfer entropy computation, and a boolean argument GPU , which allows one to use a CUDA compatible GPU for transfer entropy estimation. Lastly, $safetyCheck$ is a boolean argument that can check for duplicate rows in the dataset. This boolean argument helps prevent a more subtle error when multiple data points in a bivariate dataset have identical coordinates. For duplicate observations, several points have an identical distance to a query point during k nearest neighbors search, which violates assumptions of the Kraskov estimator. A solution used in practice and that we recommend is to add a small amount of noise to the dataset to avoid this error.

3.4 Comparative Analysis

We compare PyIF's ability to estimate Transfer Entropy against existing implementations with respect to computational performance. We present all of the data and code used to estimate TE for all implementations [2]. Each implementation in this comparative analysis estimates TE on four simulated bivariate datasets of different sizes. The estimated TE values are roughly the same for each implementation, and we forgo comparing the actual values since this is random simulated data. Thus, we assume that there is relatively little to no information transfer between the random processes. We run each of the implementations (excluding Transfer Entropy Toolbox) on nano, a cluster of eight SuperMicro servers with Intel Haswell/Broadwell CPUs and NVIDIA Tesla P100/V100 GPUs hosted by the National Center of Super Computing Applications at the University of Illinois at Urbana-Champaign. We used one node containing two E5-2620 v3 Intel Xeon CPUs and two NVIDIA P100 GPUs with 3584 cores. We refer to this as the first analysis.

We conduct the same analysis on different hardware to compare PyIF to Transfer Entropy toolbox because of MATLAB licensing issues with the National Center of Super Computing Applications. We use an Engineering Workstation with an Intel Xeon Processor E5-2680 v4 hosted by Engineering IT shared services at the University of Illinois at Urbana-Champaign. We use a single CPU core and up to 16GB of RAM to estimate TE with Transfer Entropy toolbox and PyIF. This workstation does not offer CUDA compatible GPUs for either PyIF or Transfer Entropy Toolbox, so we forgo comparing the GPU implementations. This workstation has a CPU time limit of 60 minutes, meaning that if any process uses 100% of a CPU core for more than 60 minutes, the process is terminated. We refer to this as the second analysis.

[2] The data and code can freely be downloaded from:
`https://github.com/lcdm-uiuc/Publications/tree/master/2020_Ikegwu_Traguer_McMullin_Brunner`

3.4.1 IDTxl

The first implementation that we compare PyIF to is the Information Dynamics Toolkit xl (IDTxl). IDTxl is an open-source Python toolbox for network inference (see Wollstadt et al. (2019)). Currently, IDTxl relies on NumPy, SciPy, CFFI (which is another open source library that provides a C interface for Python code; see Rigo and Fijalkowski (2018)), H5py which is a Python package that is used to interface with HDF5 binary data format (see Collette (2013)), JPype (see Menard and Nell (2018)) which is a Python module that provides a Java interface for Python code, and Java JDK, which is a developer kit to develop Java applications and applets. IDTxl has additional functionality besides estimating TE; however, we only use IDTxl's capability to estimate TE on a bi-variate dataset.

3.4.2 TransEnt

TransEnt is a R package that estimates transfer entropy (see Mount et al. (2015)). Currently, TransEnt relies on Rcpp, which acts as an interface to C++ from R. TransEnt also relies on a C++ library called Approximate Nearest Neighbors (ANN; see Arya and Mount (1998)), which performs exact and approximate nearest neighbor searches. Currently, the package is removed from CRAN. However, this software can be used and installed from Mount et al. (2015)'s Github repository [3].

3.4.3 RTransferEntropy

RTransferEntropy is a R package that estimates transfer entropy between two time-series Simon et al. (2019). Currently, the RTransferEntropy package relies on Rcpp, and the future package supports performing computations in parallel to decrease the wall time. We include both the parallel implementation of RTransferEntropy and the default implementation for completeness in the results.

[3] Mount et al. (2015)'s Github Repo: `https://github.com/Healthcast/TransEnt`

3.4.4 Transfer Entropy Toolbox

Transfer Entropy Toolbox is an open-source MATLAB toolbox for transfer entropy estimation (see Lindner et al. (2011)). This code's dependencies include: the Statistics & Machine Learning toolbox, which provides functions to analyze and model data; the FieldTrip toolbox, which is used for EEG, iEEG, MEG, and NIRS analysis; the parallel computing toolbox, which performs parallel computations of multicore CPUs and GPUs; the signal processing toolbox, which provides functions to analyze, preprocess, and extract features from sampled signals; and the TSTOOL toolbox, which is a toolbox for nonlinear time series analysis. TSTOOL no longer exists and cannot be download from its official homepage [4]. Nevertheless, the Transfer Entropy toolbox developers include pre-compiled mex files of TSTOOL that will work with this implementation. At the time of writing, Transfer Entropy toolbox's last update was in the year 2017.

3.4.5 Data

We create four bivariate datasets for this comparative analysis. Each dataset contains two time series with randomly generated values between 0 and 1. The first dataset contains 1000 observations, the second dataset contains 10,000 observations, the third dataset contains 100,000 observations, and the fourth dataset contains 1,000,000 observations. We used the seed number 23 for the pseudo-random number generator for reproducibility. We will refer to the first dataset, second dataset, third dataset, and fourth dataset as the micro dataset, small dataset, medium dataset, and the large dataset respectively.

3.5 Results

We report the results for Analysis 1 in Tables 3.1, 3.2, 3.3, and 3.4. Each table contains the wall time to estimate TE using the previously deinfed implementations on the different data

[4] `http://www.dpi.physik.uni-goettingen.de/tstool/`

sets described in section 3.4. The higher the wall time, the longer it took for the specific implementation to estimate TE. The number in the relative performance column indicates how many times slower (or faster) PyIF (CPU) is to a particular implementation. After estimating TE by using all of the implementations outlined in the comparative analysis section, we found that PyIF scales better on larger data.

Excluding the TransEnt implementation, the CPU implementation of PyIF (or PyIF (CPU)) takes less time to estimate TransferEntropy than all other implementations. The R package TransEnt has a better performance in terms of speed than PyIF (CPU) for the micro dataset and the small dataset. However, PyIF (CPU) can estimate transfer entropy in less time then all other implementations for the medium dataset and large dataset. PyIF (GPU) outperforms PyIF (CPU) for the small, medium, and large datasets. Figure 3.2 visualizes this explanation. We suspect that the optimizations performed by Numba contribute to PyIF having a larger wall time than TransEnt on the micro and small datasets.

The results for Analysis 2 are in Table 3.5. Although the Transfer Entropy Toolbox exceeds the CPU time limit for the large dataset, the results show that PyIF can scale better than Transfer Entropy Toolbox for the other three datasets. PyIF's wall times are less than Transfer Entropy toolbox's wall times, excluding the Micro Dataset. Figure 3.3 visualizes this result.

3.6 Conclusion

We address an important issue regarding large datasets with respect to the estimation of bi-variate TE. We introduce a fast solution to estimate TE with a small number of software dependencies. On large data, our implementation, PyIF, is up to 1072 times faster utilizing GPUs and up to 181 times faster utilizing CPUs than existing implementations that estimate bi-variate TE. PyIF is also open sourced and publicly available on Github for anyone to use. For future work, we plan to improve the existing code base to increase the computational performance of PyIF even further. We plan to implement additional estimators outlined in section 3.2 to estimate bi-variate TE. This boost in computational performance will enable researchers to estimate bi-variate TE much faster for various research applications. In particular, PyIF will allow us to estimate information transfer between firms with small windows in a reasonable amount of time.

3.7 Figures and Tables

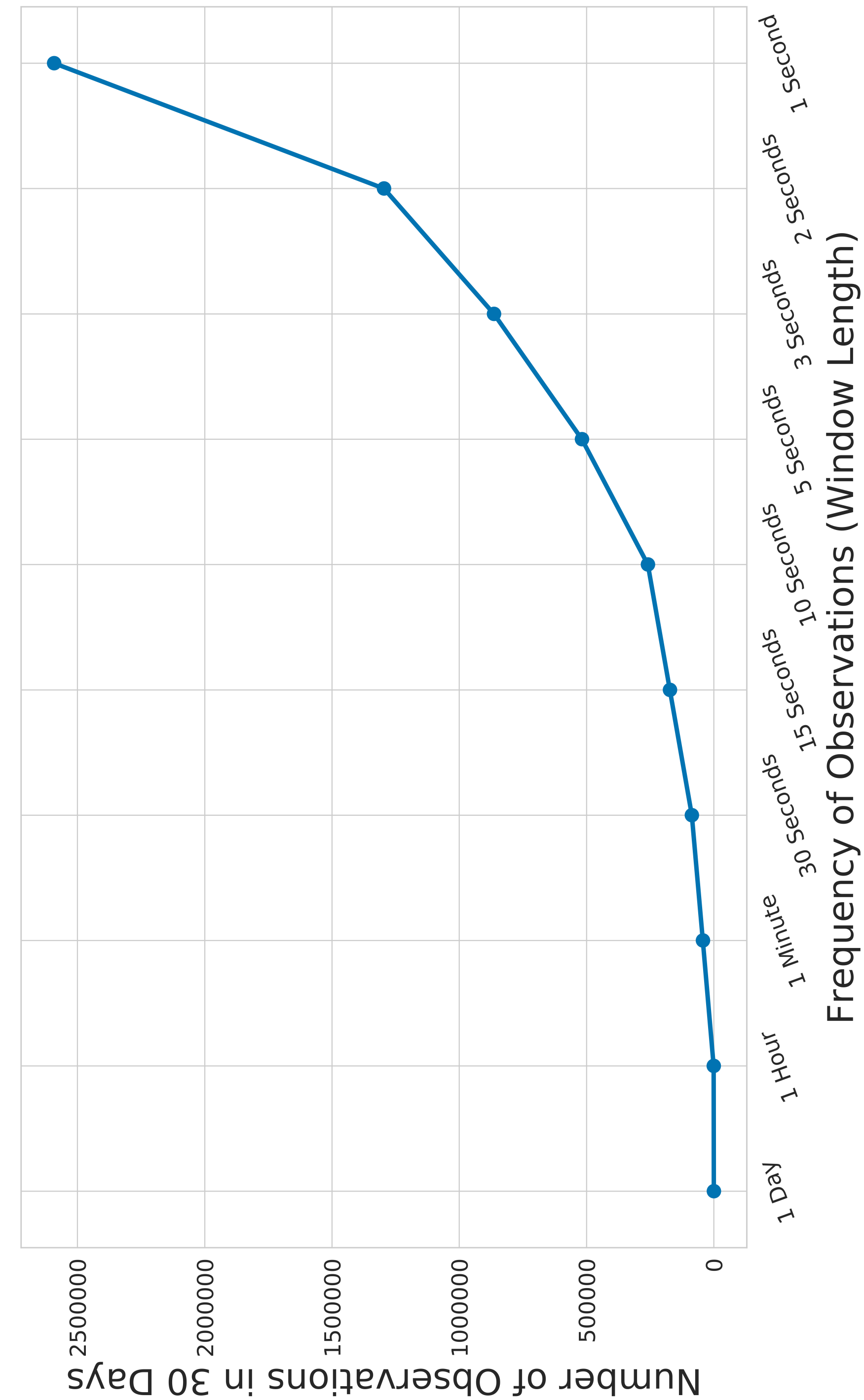

Figure 3.1: This figure shows the number of observations for 30 days based on the frequency of observations.

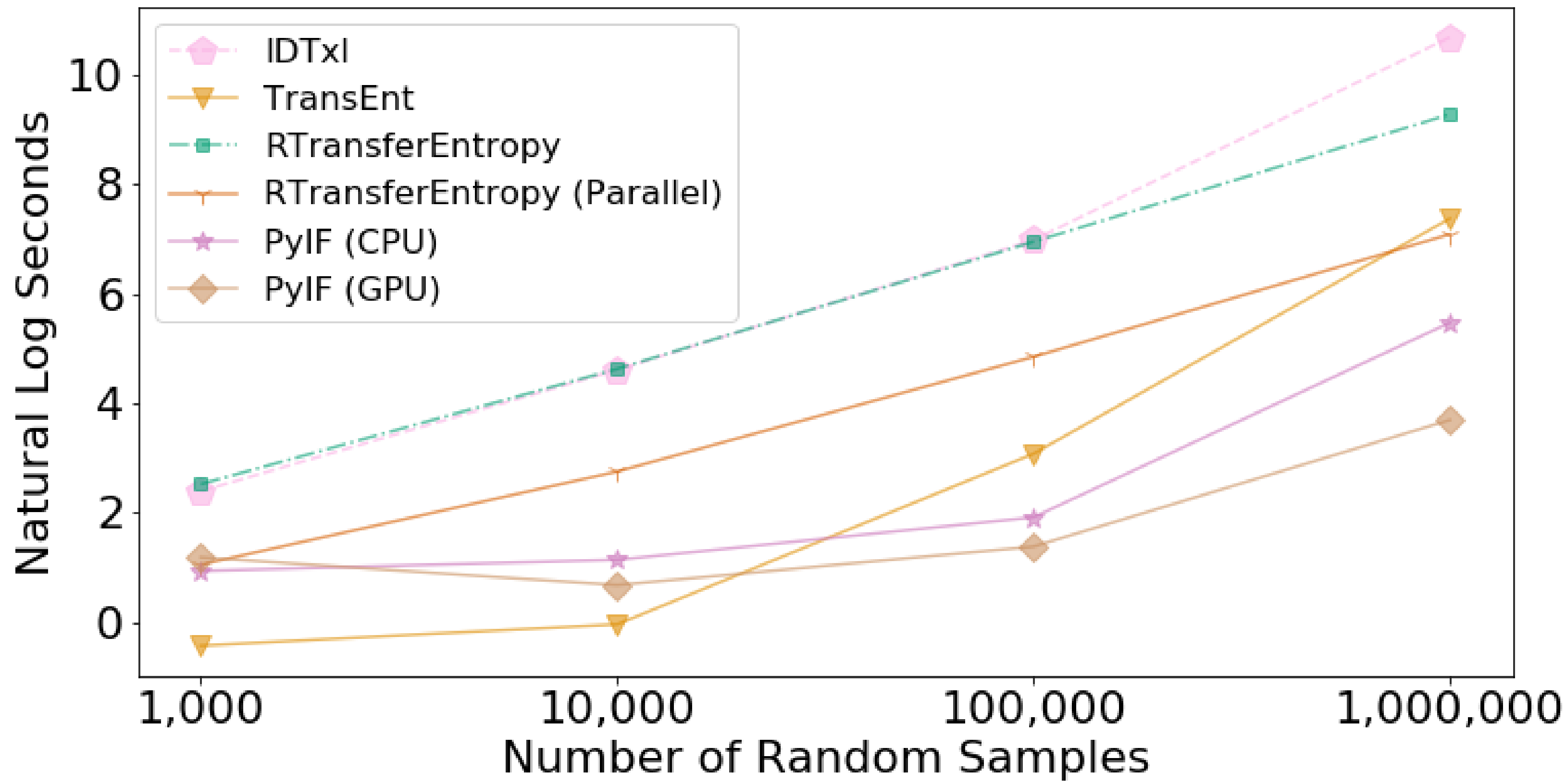

Figure 3.2: This figure shows the natural log time (in seconds) to estimate Transfer Entropy for each implementation (excluding Transfer Entropy Toolbox) for each dataset used in this study.

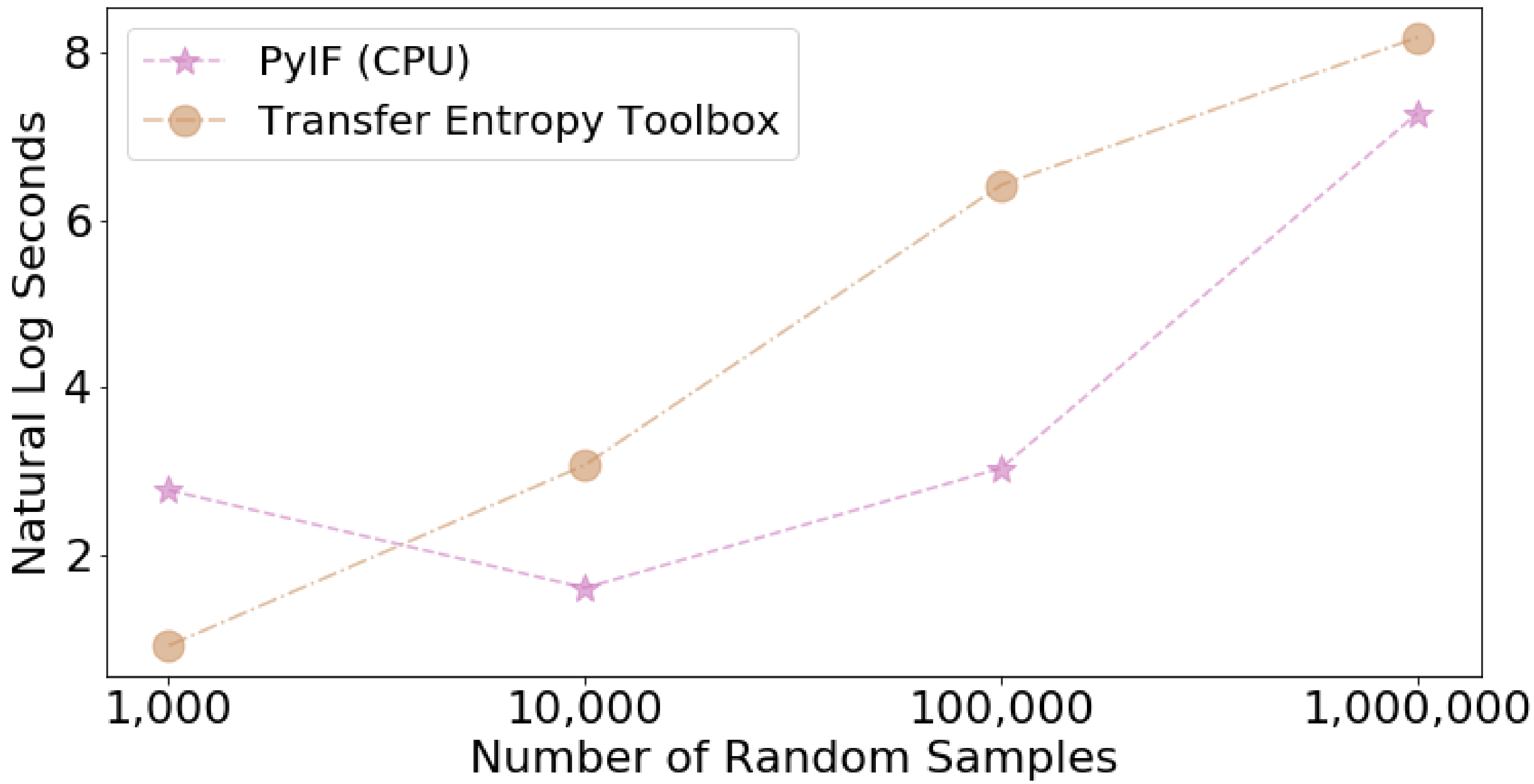

Figure 3.3: This figure shows the natural log time (in seconds) to estimate Transfer Entropy between PyIF and Transfer Entropy Toolbox on an Engineering Workstation described in the Comparative Analysis section. Transfer Entropy Toolbox exceeded the maximum allowable CPU runtime for the Large Dataset. (1,000,000 observations).

Implementation	Wall Time (in seconds)	Relative Performance to PyIF (CPU)
IDTxl	10.98	4.28
TransEnt	0.656	0.25
RTransferEntropy	12.492	4.87
RTransferEntropy (Parallel)	2.876	1.12
PyIF (CPU)	2.564	1
PyIF (GPU)	3.282	1.28

Table 3.1: Micro Data results for the first analysis.

Implementation	Wall Time (in seconds)	Relative Performance to PyIF (CPU)
IDTxl	100.23	31.94
TransEnt	0.968	0.308
RTransferEntropy	102.228	32.57
RTransferEntropy (Parallel)	15.703	5
PyIF (CPU)	3.138	1
PyIF (GPU)	1.98	0.63

Table 3.2: Small Data results for the first analysis.

Implementation	**Wall Time (in seconds)**	**Relative Performance to PyIF (CPU)**
IDTxl	1070.749	152.89
TransEnt	21.708	3.03
RTransferEntropy	1036.661	152
RTransferEntropy (Parallel)	127.281	18.66
PyIF (CPU)	6.82	1
PyIF (GPU)	3.996	0.58

Table 3.3: Medium Data results for the first analysis.

Implementation	**Wall Time (in seconds)**	**Relative Performance to PyIF (CPU)**
IDTxl	43150.129	181.97
TransEnt	1585.942	6.68
RTransferEntropy	10592.77	44.67
RTransferEntropy (Parallel)	1188.636	5.01
PyIF (CPU)	237.122	1
PyIF (GPU)	40.231	0.16

Table 3.4: Large Data results for the first analysis.

Implementation	Wall Time (in seconds)	Relative Performance to PyIF (CPU)
Micro Dataset Results (1000 Obs.)		
PyIF (CPU)	16.049	1.00
Transfer Entropy Toolbox	2.5012	0.15
Small Dataset Results (10,000 Obs.)		
PyIF (CPU)	4.989	1.00
Transfer Entropy Toolbox	21.6880	4.347
Medium Dataset Results (100,000 Obs.)		
PyIF (CPU)	20.915	1.00
Transfer Entropy Toolbox	616.8712	29.49
Large Dataset Results (1,00,000 Obs.)		
PyIF (CPU)	1455.725	1.00
Transfer Entropy Toolbox	> 3600	> 2.47

Table 3.5: Results for the second analysis.

Chapter 4

Cross-Firm Information Transfers During Earnings Season: A Network Approach

4.1 Introduction

Major public-firms announce their annual earnings during the first quarter of the year. The information embedded in these announcements affects the share price of the announcing firm and is incorporated in the share prices of other firms. Existing literature in finance and accounting focuses on measuring information transfers as effects stemming from firm-specific information releases, such as earnings announcements (see Foster (1981)). This chapter explores how firm information releases influence other firms in the economy without prior assumptions. We make two simplifying decisions when testing for informational links: an ex-ante source of fundamental linkage between firms, and a time window over which an information transfer will occur. On the first point, most previous studies assume that links between firms are both persistent and readily observable by a characteristic such as a shared industry (see Foster (1981)) or a customer-supplier relationship (see Olsen and Dietrich (1985); R.Ahren and Harford (2014)). On the second point, most studies examine an event window, typically several days long, around specific firms' announcements, such as announcing firms in an industry (see Foster (1981); Thomas and Zhang (2008)).

This chapter contains material from the following working paper:

- Robert Brunner, Kelechi Ikegwu, Bryce Schonberger, and Jeff McMullin. Cross-firm information transfers during earnings season: A network approach, April 2021

Research by Billio et al. (2012) uses a network approach to examine cross-firm transfers implicit in monthly equity returns for large firms in the financial services sector. Consistent with a dynamic network of links among these financial firms, these authors find that the extent of cross-firm information transfers increases in recent decades and is associated with significant systemic risk in the financial sector, particularly around the recent 2007-2009 Great Recession. Evidence presented by Billio et al. (2012) raises the natural question of whether dynamic cross-firm information transfers are present in a broad-based sample of firms and what factors explain variation within this network.

We construct a novel network approach during Q1 2018, where the network relies on pairwise TE estimates between firms as a proxy for information transfer. Our approach contributes to extant accounting and finance literature. We provide novel evidence on the network of cross-firm links implied by the lead-lag structure implicit in high-frequency U.S. equity prices observed around releases of earnings information. By tracing the network of information transfers directly, we can document several features of cross-firm links that go beyond industry. Finally, we use a community detection algorithm to uncover latent clusters of firms with shared information links. Ultimately we provide evidence that the presence and size of these communities significantly differ depending on the presence of earnings releases. We hope to encourage future work designed to broaden our understanding of the complex system of information flows inherent in modern equity markets.

For the remainder of this chapter, we first discuss the datasets used in this study. Next, we discuss the methodology for estimating cross-firm information transfers and discuss the network science used to create dynamic cross-firm information transfer networks. We also make observations about the structural properties of the networks. Then, we conduct an analysis to determine the effect of earnings surprise on dynamic cross-firm information transfers, and we discuss the results of the analysis. Finally we present an analysis involving community detection within the network of cross-firm transfers and conclude.

4.2 Data

The data in the study comes from Wharton Research Data Services (WRDS) (see Wharton School (1993)). We obtain security prices from the Trade and Quote (TAQ) dataset, which contains all trades and quotes that occurred at a sub-second level. We use Holden and Jacobsen (2014) SAS code to measure the national best bid/offer(NBBO) price. The price is updated as trades or quotes occur. This means that the timing of the prices is dependent upon when these events occur.

We obtain the NBBO price data for firms in the S&P 500 in Q1 2018. We construct ten datasets for each firm at different sampling rates. The sampling rates used in this study are: 1 second, 2 seconds, 3 seconds, 4 seconds, 5 seconds, 10 seconds, 15 seconds, 30 seconds, 60 seconds, and 120 seconds. For example, at 1 second sampling rate, there will be 60 NBBO price observations per minute for a particular firm. CRSP and Compustat are used to obtain firm-specific information. We also use the Institutional Brokers' Estimate System (IBES) to obtain earnings surprise for firms in the S&P 500 during Q1 2018. Throughout the text, we describe how variables are used from IBES, CRSP, and Compustat.

4.3 Estimating Dynamic Information Transfer Between Firms

We estimate the amount of information transfer between every unique pair of firms in the S&P 500 during Q1 2018 (the first 61 days of 2018) by using TE. To detect information transfer at the minute/sub-minute level between firms in the S&P 500, we first obtain the NBBO prices from the TAQ data for all firms in the S&P 500 in Q1 2018. We create ten datasets from the NBBO data for each date and firm. We sample the NBBO price data with the following sampling rates: 1, 2, 3, 5, 6, 10, 15, 30, 60, and 120 seconds. We select these sampling frequencies to allow for a sufficiently large sample of pricing observations in each time window to compute TE. These sampling frequencies also help determine at what speed information transfer peaks.

Next for each date, firm, and sampling rate, bi-variate TE will be estimated with the NBBO sampled price data using three 130 minute windows throughout the trading day. We use these three windows to determine if there are any consistent information flow patterns during the beginning (9:30am-11:40am), middle (11:40am-1:50pm), or end (1:50pm-4pm) of the trading day. To summarize TE will be estimated to a specific firm from all other firms in the S&P 500 for that particular date, sampling rate, and interval. Given that there are 10 sampling rates, 3 windows, 61 days, and 499 other firms, this amounts to 913,170 TE calculations per firm for Q1 2018 (or 456,585,000 TE calculations in total).

Data from small sample rates will yield a high number of observations. The 1-second sampling rate has the most observations and is the most costly to estimate TE. For a particular trading day, running TE computations with PyIF (see Chapter 3) on the HAL cluster at the National Center for Supercomputing Applications using a single NVIDIA V100 GPU to 1 firm from the other 499 firms for the three windows will take roughly 4 minutes. For all 500 firms, this took about 2000 minutes. For all days in Q1 2018, this took about 122,000 minutes (or about 85 days) at the 1-second sampling rate sequentially. Subsequently, for the 2-second sampling rate, the data is reduced by half. Thus, the wall time was reduced by roughly half. Given that we can run up to five jobs in parallel on HAL, 1-second TE estimates took 17 days to compute.

4.3.1 Algorithmic Process for Estimating Information Transfer Between firms

In this section, we outline the algorithmic process to estimate information transfer between firms using the Q1 2018 data (see Algorithm 1). In Algorithm 1, we create a dictionary to map combinations of dates and sampling rates with matrices of computed information transfers. Given the dates of interest in line 2 and the sampling rates in line 3, we iterate through each date in line 4. For a particular date, we iterate through each sampling rate in line 5. For each unique combination of date and sampling rate, we select firms with observations in that date and sampling rate. Next, we create an empty matrix of size $\text{Firms}_i{}^2$ by 3, and we use a counter variable to keep track of the observations in line 9.

Next, we iterate through each firm in the set of firms_i. In Algorithm 1 the trading day is split into three, two hour and ten minute windows that represent the beginning (9:30am-11:40am), middle (11:40am-1:50pm), and end of the trading day (1:50pm-4pm). In lines 10-12, we filter the NBBO prices for firm i to the beginning of the trading day, middle of the trading day, and end of the trading day for a single firm from the set of firm_i . We repeat this process for all of the firms in the set firm_j and estimate TE from firm_j to firm_i. The TE estimate is assigned to row cnt_{ij} and column zero of the dateSR_InfoFlow matrix for the particular key of the dictionary for the TE computation with the morning prices. Subsequently, the TE estimate is assigned to row cnt_{ij} and column one or two for the middle of the trading day or end of the trading day, respectively. Finally, we increment cnt_{ij} by one and repeat this process for all pair of firms, sampling rates, and days. Following this algorithm produces a dictionary of computed information transfers for each unique pair of date and sampling rate to perform analyses on.

Algorithm 1: Estimating Information Transfers Between Firms

```
dateSR_InfoFlow := { } ;
Dates := All Dates in Q1 2018 ;
SampleRates := [1, 2, 3, 4, 6, 10, 30, 60, 120] secs ;
for t ∈ Dates do
    for SR ∈ SampleRates do
        Firms_i := SelectFirms(t, SR) ;
        Firms_j := SelectFirms(t, SR) ;
        dateSR_InfoFlow[(t,SR)] := Matrix(length(Firms_i)^2, 3) ;
        cnt_ij: = 0 ;
        for Firm_i ∈ Firms_i do
            MoPrices_i := Filter(Firm_i, "9:30am-11:40am") ;
            MidPrices_i := Filter(Firm_i, "11:40am-1:50pm") ;
            AfterPrices_i := Filter(Firm_i, "1:50pm-4:00pm") ;
            for Firm_j ∈ Firms_j do
                MoPrices_j := Filter(Firm_j, "9:30am-11:40am") ;
                MidPrices_j := Filter(Firm_j, "11:40am-1:50pm") ;
                AfterPrices_j := Filter(Firm_j, "1:50pm-4:00pm") ;
                dateSR_InfoFlow[(t,SR)] [cnt_ij,0] := ComputeTE(MoPrices_i,
                  MoPrices_j) ;
                dateSR_InfoFlow[(t,SR)] [cnt_ij,1] := ComputeTE(MidPrices_i,
                  MidPrices_j) ;
                dateSR_InfoFlow[(t,SR)] [cnt_ij,2] := ComputeTE(AfterPrices_i,
                  AfterPrices_j) ;
                cnt_ij += 1
            end
        end
    end
end
```

4.3.2 Network Creation

After we compute all bi-variate TE estimates, we conduct exploratory data analysis to find patterns or characteristics of the information transfer between firms to explore further. We employ network analysis techniques (see Chapter 1.2) to create an information transfer network for all firms at date t, sampling rate s, and window w. Figure 4.1 shows a single network for January 2nd, 2018, during the morning window (9:30am-11:40am). Each circle in the network is a node that represents a firm. The lines (or edges) between nodes have a thickness determined by the information transfer estimate as computed via transfer entropy.

The information transfer network in Figure 4.1 has been filtered with a Disparity Filter (see Serrano et al. (2009)) to reduce the amount of edges in a network. However, it is still challenging to gather insight from this network. In addition, there is an information transfer network for each day, t, s, and w. Creating networks for all days, sampling rates, and windows will produce roughly 1,830 networks, which will make it more difficult to discover general insights from the data by viewing them. An alternative is to look at network measures to gain additional insights from the information transfer networks.

4.4 Exploratory Data Analysis

We now turn to an exploratory analysis of the computed information measures. Figure 4.2 shows the distribution of information transfers for all days in Q1 2018 for all windows at each sampling rate. Each subplot is for a particular sample rate, the x-axis represents the bin value, and the y-axis represents the percentage of observations in a bin. Faster sampling rates have lower variance and higher means. Slower sampling rates have higher variance and smaller means. A possible explanation is that the frequency of observations decreases due to fewer non-overlapping observations during each 130-minute window. For example, there are 7,800 observations per firm at a 1-second sample rate and 130 observations per firm at a 1-minute sample rate. The 1-minute yields fewer data to compute TE and produces a noisier information transfer estimate.

Table 4.1 show additional summary statistics of the information transfers at each sampling rate. The 75th and 99th percentile values increase while the median percentile (and lower) values decay as the sample rate becomes slower. Table 4.2 contains the average TE values computed for each of the 130-minute trading windows during the first quarter of 2018 for pairs of firms in the S&P 500.

Across sampling rates, Table 4.2 mean values exhibit similar patterns to Table 4.1 with a decrease in means as the sampling rates become slower. Across the three 130-minute windows on each trading day, the 9:30 am-11:40 am (morning) window displays the highest average information transfer. This latter result is consistent with a surge in trading at the market open each day.

In Table 4.3, we investigate the mean TE values at narrower time windows to determine when information transfers tend to peak during the trading day. Table 4.3 contains the average TE values computed for thirteen, non-overlapping, thirty-minute trading windows. Across all sampling rates, average TE values are highest during the first thirty minutes of the trading day. Mean TE values then decay throughout the trading day at all sampling rates. Sampling rates faster than 6 seconds have a slight burst of information transfer within the last hour of the trading day, consistent with information-based trading as part of the daily settlement (closing) trade.

Table 4.4 shows the average weighted out-degree of the information transfer network at various sampling rates across 61 morning trading periods (9:30am-11:40am). We scale each degree measure by the number of firms in the network at each measurement period. Since all firms in the information transfer networks are connected, the average weighted out-degree values across all firms in the first quarter of 2018 are equivalent to the mean weighted in-degree values. Stated differently, the average of all incoming information transfers from a set of firms to a particular firm is equivalent to the average of all outgoing connections from a set of firms to a firm. The average weighted incoming and outgoing information transfer spikes at a sampling rate of 5 – 6 seconds.

Tables 4.5 and 4.6 present results of ordinary least squares models examining auto-correlation in daily measures of weighted network degree to determine whether firms' centrality in the network of information transfers displays persistence. In particular, Table 4.5 (Table 4.6) presents auto-correlations for incoming (outgoing) information transfers based on the weighted in-degree (out-degree) computed from 9:30am-11:40am for each trading day in our 2018 sample period. The R^2 values for both tables exhibit similar behaviors across sampling rates. In particular, at faster sampling rates we see more explained variation from the morning information transfers with a gradual decay as the sample rates become slower. The R^2 values for the auto-correlation models for outgoing information transfers during morning trading windows in Table 4.6 are substantially higher than the incoming morning information transfers in Table 4.5. This provides evidence that each firm's position in the information transfer network is persistent.

4.5 Earnings Surprise as a Determinant of Dynamic Information Transfers

Prior literature finds that more informative news events elicit more robust stock price responses among peer firms (see Foster (1981); Brochet et al. (2018)). Given this, we examine whether firms with larger absolute earnings surprises (measured relative to the consensus analyst forecast) display stronger information transfers in response to their earnings announcement. This analysis allows us to shed light on the dynamics of the information transfer network by focusing on variation in the timing of firm's earnings announcements during earnings season and on variation in the news contained in the announcement.

In particular, we estimate an ordinary least squares regression model of the form:

$$
\begin{aligned}
Y_{iwt} = \alpha + \beta_1 EA_{iwt} * Morning_{iwt} + \beta_2 EA_{iwt} * Morning_{iwt} * Abs\_Suprise_{iwt} + \\
\beta_3 EA_{iwt} * Afternoon_{iwt} + \beta_4 EA_{iwt} * Afternoon_{iwt} * Abs\_Suprise_{iwt} + \\
\beta_5 EA_{iwt} * Evening_{iwt} + \beta_6 EA_{iwt} * Evening_{iwt} * Abs\_Suprise_{iwt} + \\
\beta_7 Share\_Turnover_{it} + \beta_8 Abs\_RET_{it} + \\
\beta_9 Morning_{it} + \beta_{10} Evening_{it} + \eta_i + \lambda_t + \epsilon_i
\end{aligned}
\tag{4.1}
$$

Y_{iwt} is the weighted out-degree (or in-degree) for the i^{th} firm at the day t and window w (i.e, morning, afternoon, or evening). EA_{iwt} is an indicator variable for the i^{th} firm at day t and window w which is set equal to one for trading days with an earnings announcement for the i^{th} firm made after the prior market close or during the current trading day and to zero otherwise. $Morning$, $Afternoon$, and $Evening$ are indicators variables set equal to one if the current observation is during the 9:30 am - 11:40 am, 11:40 am-1:50 pm, and 1:50 pm-4:00 pm trading windows, respectively.

$Abs\_Surprise$ is the absolute value of the quarterly earnings surprise measured as the difference between actual earnings in I/B/E/S and the last available consensus analyst earnings forecast from the I/B/E/S Summary file, scaled by the quarter-end stock price available from Compsutat. The $Abs\_Surprise$ variable is set to zero for all trading windows other than the first Morning, Afternoon, and Evening trading windows following the announcement. $Share\_Turnover$ controls the number of shares traded during the trading day, scaled by the number of shares outstanding on CRSP. Abs_RET and the absolute value of the close-to-close stock return from CRSP. Equation 4.1 treats each 130-minute trading window for each firm as a distinct observation. To focus on within-firm variation in centrality we include firm-fixed effects η_1 and trading day fixed-effects λ_t. This analysis excludes four outlying observations with absolute earnings surprises larger than 5% of price.

We present results of estimating Equation 4.1 in Tables 4.7 and 4.8, where Table 4.7 (Table 4.8) presents results of OLS models where weighted out-degree (in-degree) is the dependent variable. Results in Table 4.7 show that a firm's weighted out-degree in the network of information transfers is significantly higher in the trading periods immediately following its earnings announcement when the firm announces a larger earnings surprise. In particular, the positive coefficient on the $EA * Morning * Abs\_Surprise$ term across the sample rates in models (1) – (10) ranges from 1.672 in model (1) for the 1-second sample rate to 3.117 in model (4) for the 5-second sample rate. All estimates are statistically significant at a 5% two-tailed level. Further, the pattern in coefficients on the $EA * Morning * Abs\_Surprise$ term across models (1) – (10) in Table 4.7 suggests an information transfer that peaks at 5 seconds, consistent with our transfer entropy estimates presented in Table 4.1. A significant relation between weighted_out-degree and earnings news persists into the afternoon trading window at faster sample rates, with significant positive coefficients on the $EA * Afternoon * Abs\_Surprise$ term for sample rates ranging from 1 second to 3 seconds in models (1) – (3) (two-tailed p-values < 0.05).

Turning to results in Table 4.8 for weighted network in-degree shows that in contrast to results in Table 4.7, weighted_in-degree displays limited variation with the magnitude of earnings news released. Except for the $EA * Morning * Abs\_Surprise$ term in models (1) and (2) for the fastest sample rates, coefficients on the interaction terms with $Abs\_Surprise$ are generally insignificant at conventional levels and display no clear pattern across sample rates. We interpret these results as suggesting that transfers out to other firms are significant in response to the magnitude of earnings news released, while transfers in are limited in response to the news released.

4.6 Community Detection

We attempt to use community detection algorithms on the information transfer networks to uncover dynamic community formation across morning trading windows. To limit computational time, we select three morning trading periods for analysis: two periods with a large number of earnings announcements (January 31 and February 1) and one period with no earnings announcements (March 13). We further focus on information transfers at faster speeds by taking the average information transfer across 1 – 6 second sampling rates for each of the three morning trading windows selected and reconstruct the information transfer network.

In network science literature, a standard definition of a community is to have more nodes grouped where there is a higher density of edges within a group than between groups. Given that the information transfer networks are relatively large, we utilize the Clauset-Newman-Moore greedy modularity maximization algorithm to find communities (see Clauset et al., 2004). While the Clauset-Newman-Moore algorithm aims to find communities in vast networks, it cannot find communities in the information transfer networks.

The density in our information transfer networks are 1, whereas Clauset et al. (2004) benchmark network had a density of 0.0000125. Another way to think of this is that the ratio of edges to nodes in our network is 100 times greater than their benchmark network. While we would ideally employ a community detection algorithm able to scale to the size of these information transfer networks, we are unaware of an such an algorithm. Thus, we take an alternative approach by reducing the nodes-to-edges ratios for our information transfer networks. This allows us to use standard community detection algorithms.

With the Clauset-Newman-Moore algorithm to combat the density issue, we reduce the density (or the node to edges ratio). We apply a disparity filter (see Serrano et al. (2009)) to the information transfer networks. This method locally identifies statistically relevant weighted edges and can filter out relevant connections across all scales of interactions between firms. This method locally identifies statistically relevant weighted edges by removing any edge that is weaker than a statistical cutoff designed to measure the importance of a given connection to each node. As a result, the key input for the disparity filter is a researcher-selected value for α used to identify sufficiently strong connections between nodes

We first take the average information transfer across $1 - 6$ second sampling rates for the morning trading windows and reconstruct the information transfer networks. Next, we apply the disparity filter to find the network backbone, which requires a value for α. The selection of α is more of an art than science. Table 4.9 shows the amount of nodes and edges remaining in a filtered network at a particular α value. A smaller α will yield an information transfer network that is too sparse to detect communities. A large α will yield an information transfer network that is too dense to detect communities.

We select α based on the Clauset-Newman-Moore algorithm's ability to detect communities for the most firms in the sample information transfer networks. We found that when $\alpha = 0.375$, we can detect communities for 100% of the firms on February 1st and 97% of the firms on January 31st and March 13th. We find fewer communities formed during announcement days (January 31st and February 1st) than the trading day with no announcements (March 13th). However, most of the community's sizes during the announcement days are larger than the communities in the March 13th information transfer network (see Figure 4.3). On this point, Figure 4.3 shows that most of the firms in the March 13th information transfer network are in communities with 1, 2, or 3 other firms. During the announcement days, it is less likely to see communities with 1, 2, or 3 other firms; instead, we find more communities with larger sizes.

On the announcement days, the communities typically have only one firm that announced. There are three cases where a community has two firms that announced and one case where three firms announced. We see a diverse set of industries connected within and across the detected communities. Figure 4.4 presents the full set of nodes and resulting communities for the information transfer network for the March 13th morning trading window. This figure shows the presence of several larger communities with a handful of central (larger) nodes. In addition, Figure 4.4 shows a number of firms (in gray) appearing with small (or no) communities. To aid in graphically presenting these communities, Figures 4.5, 4.6, and 4.7 presents only communities with at least 9 firms in the resulting community.

For all the information transfer networks, there is a dynamic formation of hubs. Some hubs reoccur, and others form during a particular trading day. For example, in Figures 4.5, 4.6, and 4.7 Amazon (AMZN), National Weather Service (NWS), and Mettler-Toledo International Inc (MTD) are hubs during the announcement days and the non-announcement day despite the different connections that form to each of these hubs. However, when Boeing (BA) announced on January 31st, it temporarily appears as a hub in Figure 4.5 for that morning information transfer network. The networks' characteristics provide evidence that the networks are not random, which indicates that we are capturing valid cross-firm information transfers via our measurement of transfer entropies.

4.7 Conclusion

We introduce a new approach to examine information transfers around earnings season. We study the network effect of earnings announcements by constructing daily networks of pairwise cross-firm information transfers. Our approach to construct these networks relies on non-parametric estimates in equity prices with measures of transfer entropy drawn from information theory. While it is known that earnings information produced by a single firm is incorporated into the firm's equity price, we provide evidence that this information flows to the equity prices of other firms across industries. In particular our tests show that cross-firm links are substantially stronger for firms on days with releases of earnings information and firms with more unexpected earnings news, consistent with the network of cross-firm information transfers dynamically responding to shifts in the information landscape. Further, we find that communities form between firms with links that are not adequately captured by characteristics that are the focus of existing literature. These analyses also demonstrate that the prior literature's focus on measuring information connections between firms in the same industry or along the supply chain likely result in missing important information linkages across more diverse firms.

4.8 Figures and Tables

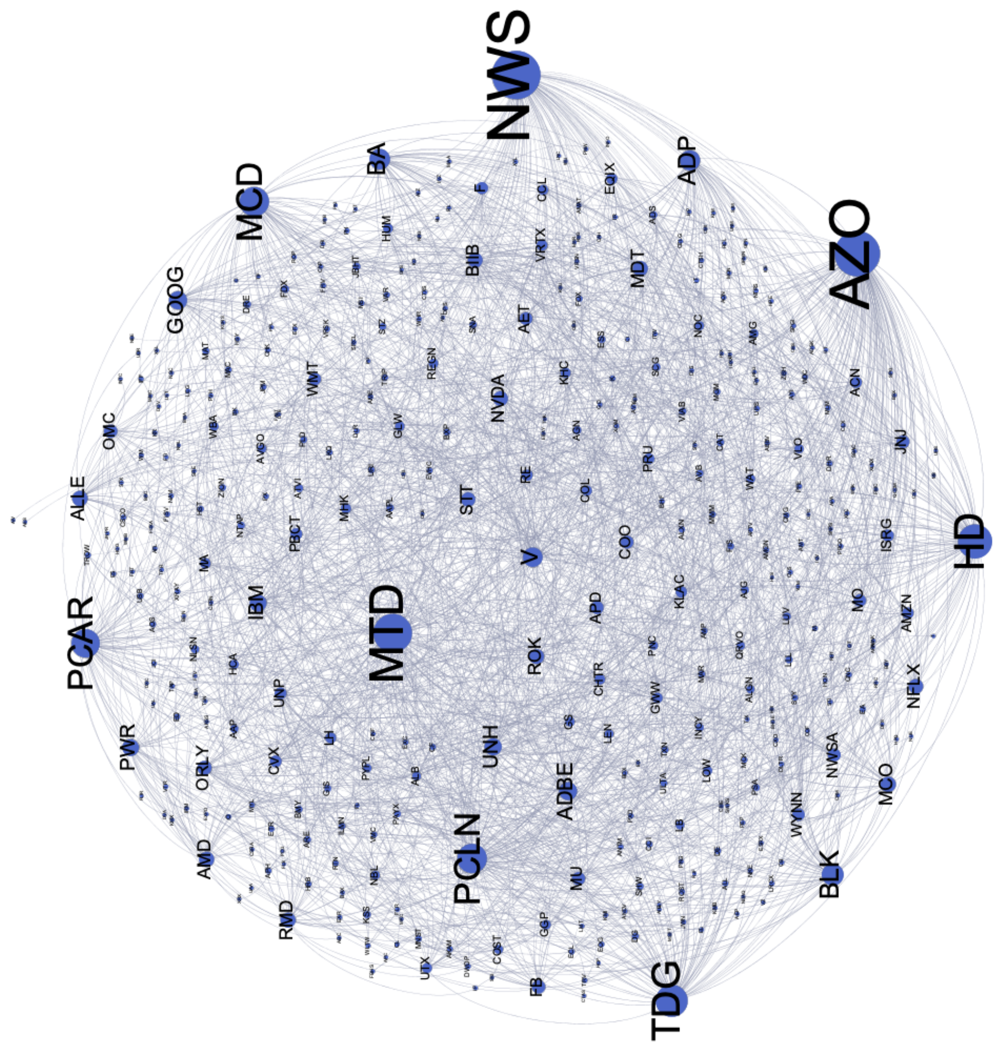

Figure 4.1: This figure shows an example network on January 2nd during the morning window (9:30 am-11:40 am) at the 1-second sampling rate.

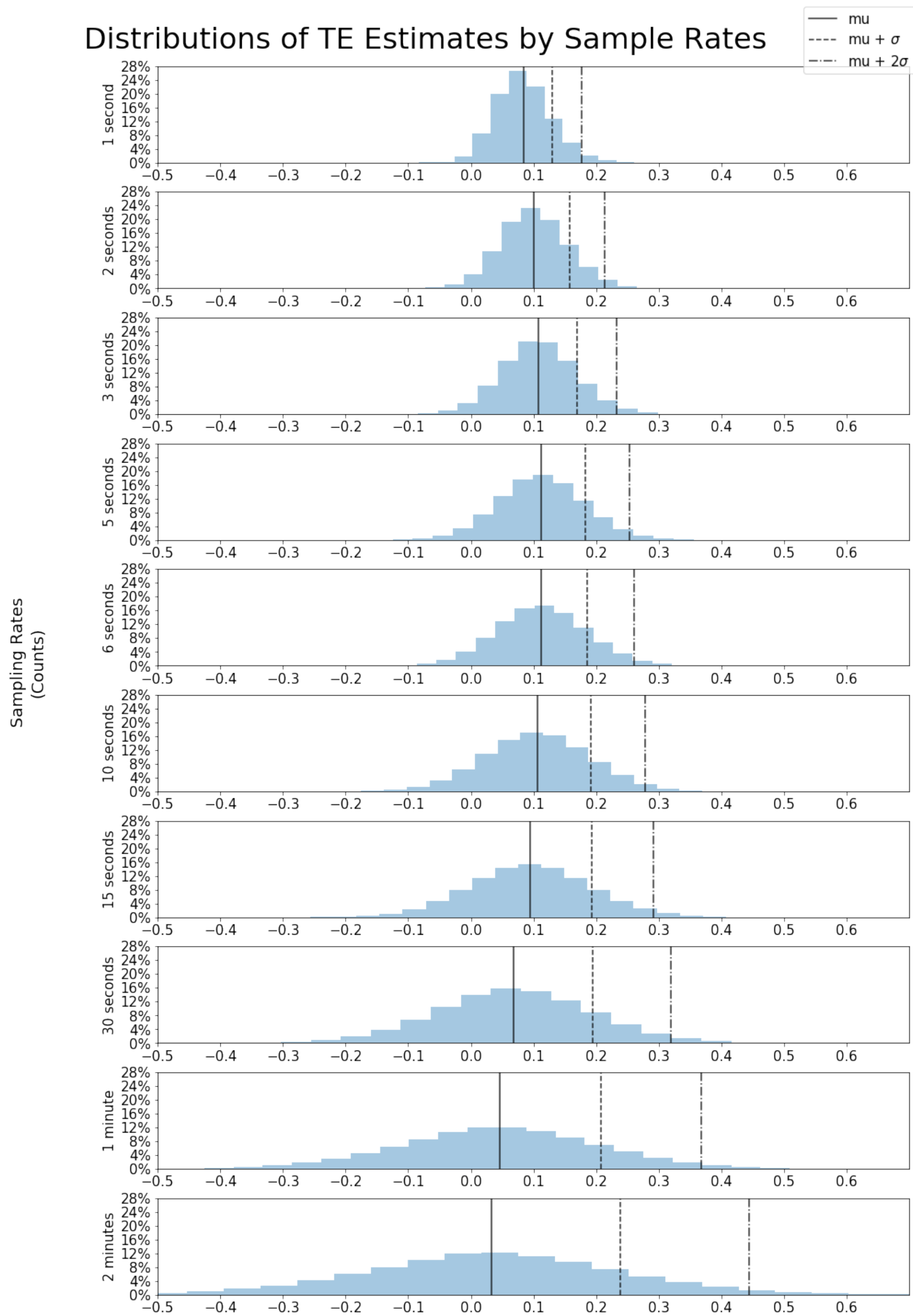

Figure 4.2: This figure shows the distributions of information transfers between firms during Q1 2018 for all sampling rates.

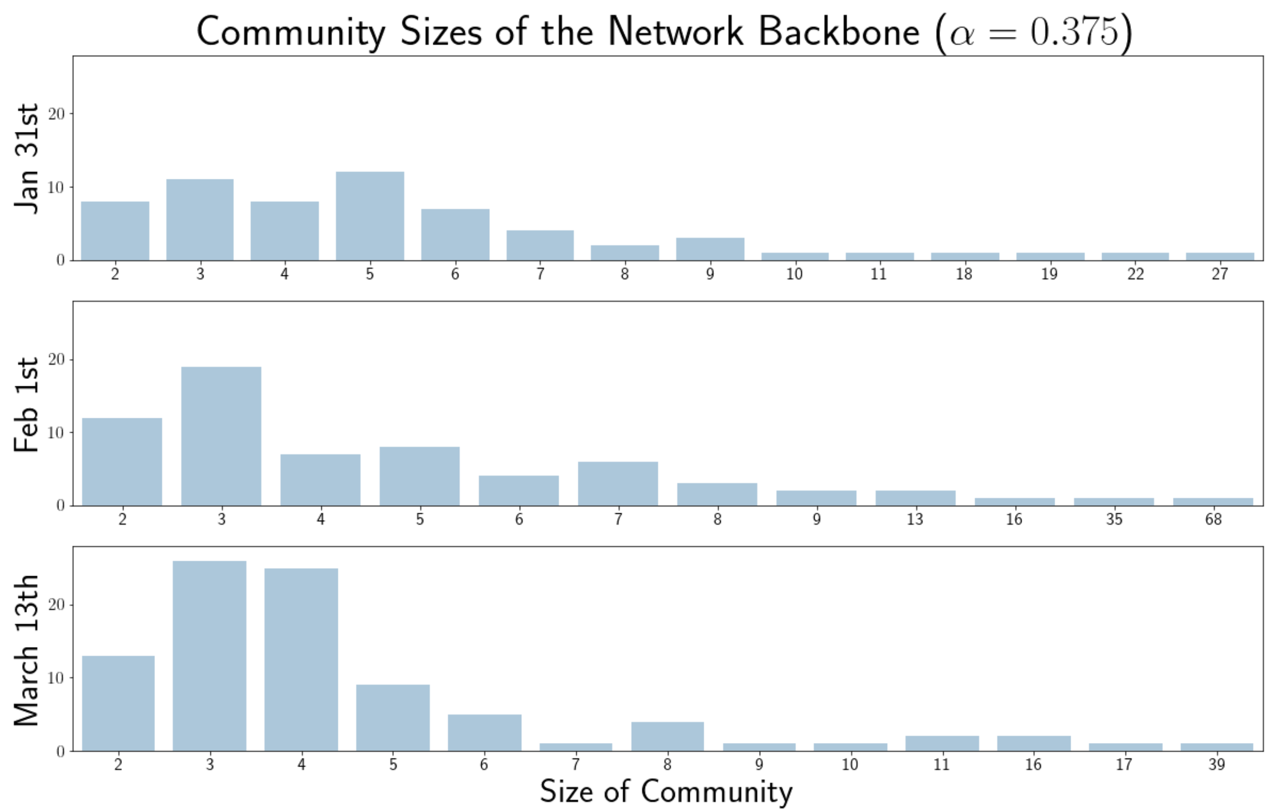

Figure 4.3: Community sizes of the network backbone when $\alpha = 0.375$ for January 31st, February 1st, and March 13th

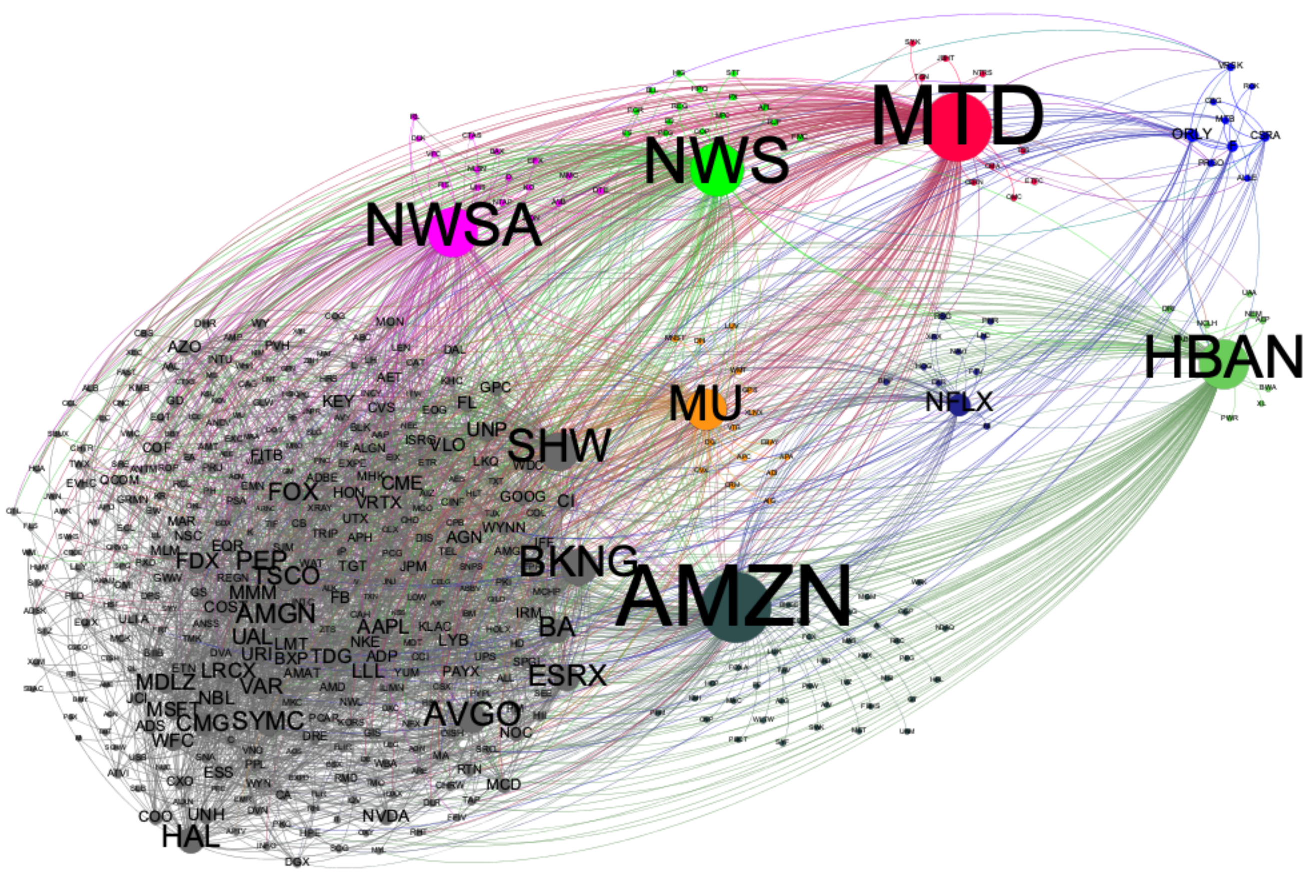

Figure 4.4: This figure shows the backbone network displaying communities for March 13th. Each color represents a specific community, and the node's size represents the degree of the node. All communities and nodes with at least 9 firms appear with a color other than gray for each community.

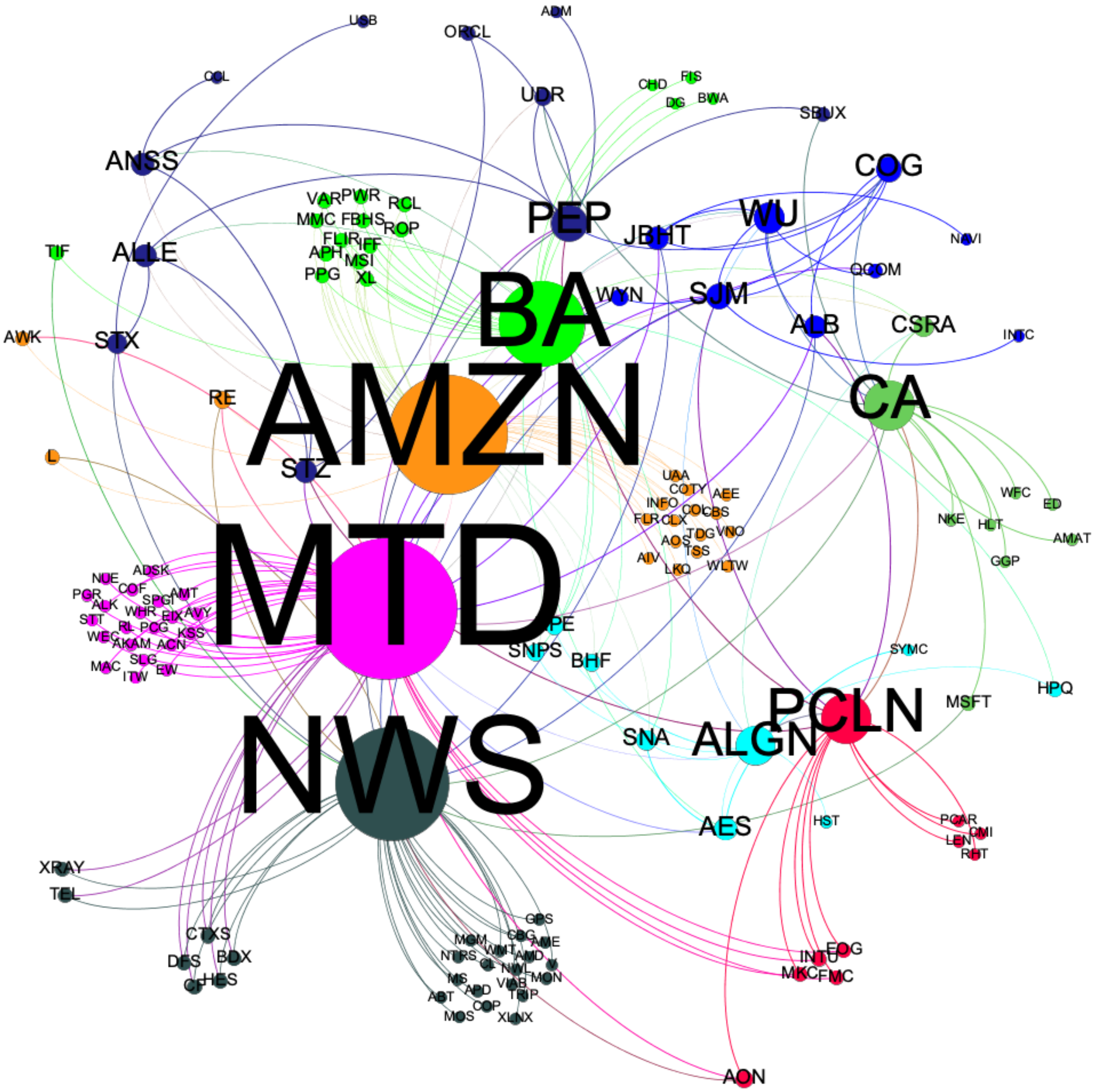

Figure 4.5: This figure shows the backbone network displaying communities for January 31st, where at least nine firms are in the community. Each color represents a specific community, and the node's size represents the degree of the node.

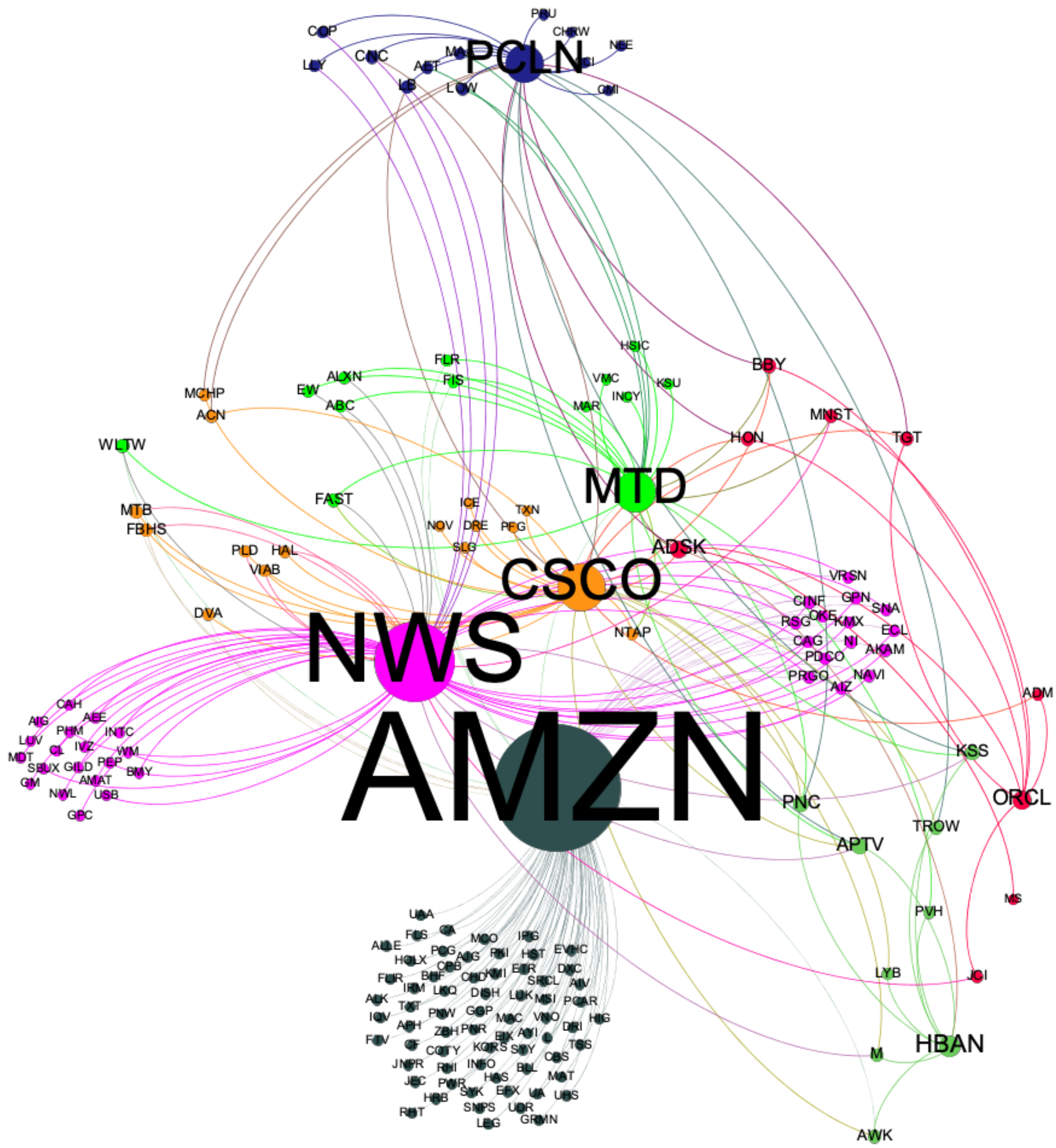

Figure 4.6: This figure shows the backbone network displaying communities for February 1st, where at least nine firms are in the community. Each color represents a specific community, and the node's size represents the degree of the node.

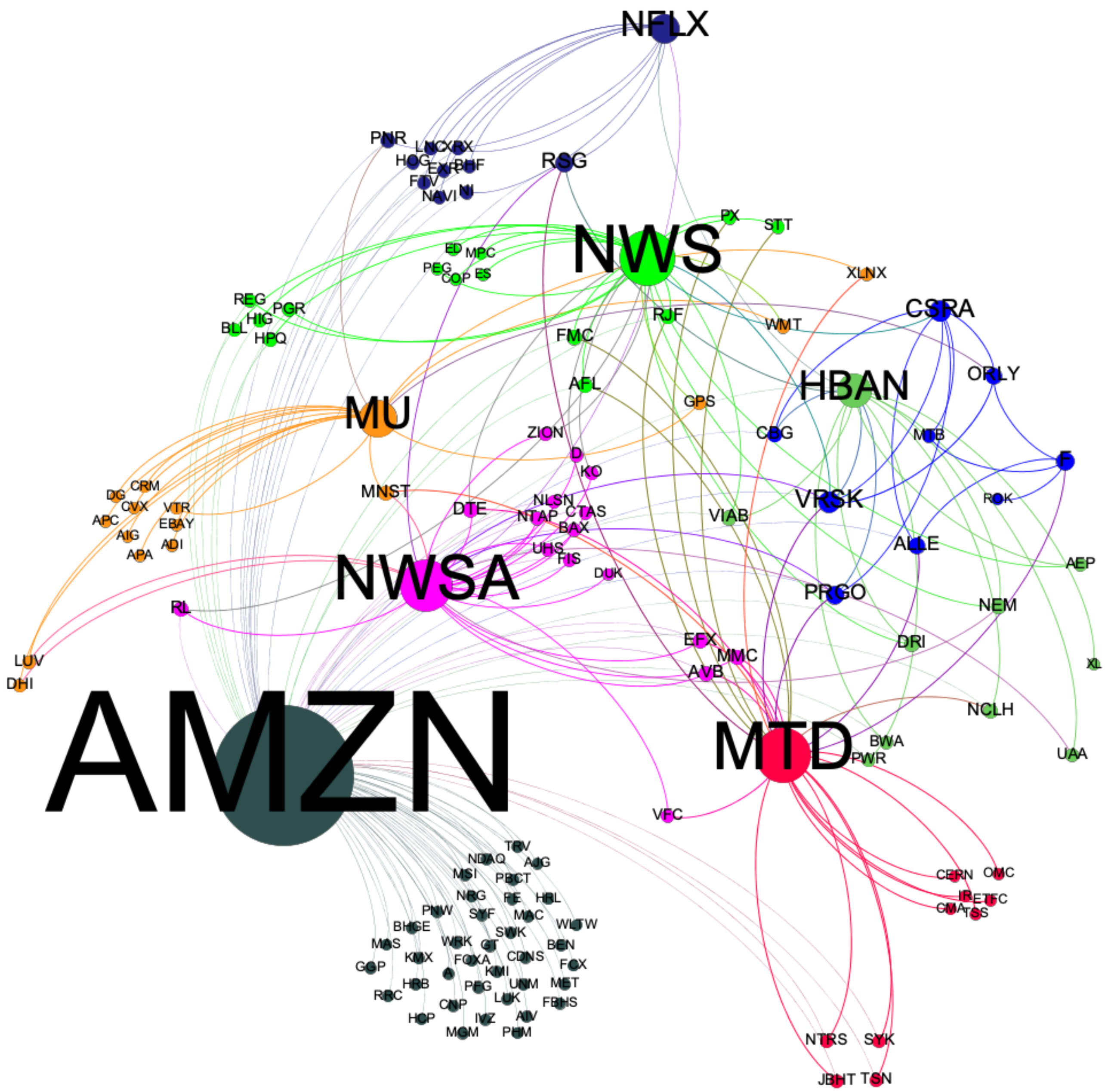

Figure 4.7: This figure shows the backbone network displaying communities for March 13th where at least nine firms are in the community. Each color represents a specific community, and the node's size represents the degree of the node.

Sample Rate	**1 second**	**2 seconds**	**3 seconds**	**5 seconds**	**6 seconds**	**10 seconds**	**15 seconds**	**30 seconds**	**1 minute**	**2 minutes**
Mean	0.084	0.101	0.108	0.112	0.112	0.106	0.094	0.069	0.046	0.033
St. Deviation	0.046	0.056	0.062	0.070	0.074	0.086	0.099	0.125	0.161	0.206
1st percentile	-0.014	-0.025	-0.033	-0.052	-0.061	-0.099	-0.141	-0.233	-0.344	-0.472
Q1	0.054	0.065	0.069	0.067	0.064	0.049	0.030	-0.013	-0.058	-0.098
Median	0.082	0.100	0.108	0.112	0.112	0.106	0.094	0.069	0.047	0.034
Q3	0.112	0.136	0.148	0.158	0.160	0.163	0.159	0.152	0.153	0.167
99th percentile	0.201	0.234	0.253	0.275	0.283	0.306	0.323	0.359	0.422	0.517

Table 4.1: This table shows the summary statistics for transfer entropy estimates during non-overlapping 130-minute trading windows.

Sample Rate	**1 second**	**2 seconds**	**3 seconds**	**5 seconds**	**6 seconds**	**10 seconds**	**15 seconds**	**30 seconds**	**1 minute**	**2 minutes**
9:30am-11:40am	0.097	0.117	0.125	0.130	0.130	0.123	0.109	0.082	0.058	0.044
11:40am-1:50pm	0.071	0.088	0.096	0.102	0.103	0.100	0.091	0.067	0.044	0.030
1:50pm-4pm	0.083	0.098	0.103	0.105	0.103	0.095	0.082	0.057	0.037	0.024

Table 4.2: Mean transfer entropy estimates for non-overlapping 130-minute daily trading windows at each sample rate.

Sample Rate	1 second	2 seconds	3 seconds	5 seconds	6 seconds	10 seconds	15 seconds	30 seconds
09:30AM-10:00AM	0.099	0.118	0.127	0.130	0.129	0.118	0.105	0.078
10:00AM-10:30AM	0.085	0.101	0.107	0.109	0.108	0.098	0.084	0.058
10:30AM-11:00AM	0.077	0.093	0.099	0.101	0.100	0.092	0.082	0.058
11:00AM-11:30AM	0.073	0.087	0.093	0.096	0.095	0.089	0.077	0.056
11:30AM-12:00PM	0.069	0.085	0.091	0.095	0.095	0.090	0.080	0.059
12:00PM-12:30PM	0.065	0.080	0.086	0.091	0.091	0.087	0.080	0.058
12:30PM-01:00PM	0.061	0.076	0.084	0.089	0.089	0.086	0.078	0.057
01:00PM-01:30PM	0.061	0.076	0.082	0.086	0.088	0.085	0.076	0.057
01:30PM-02:00PM	0.062	0.076	0.083	0.088	0.089	0.085	0.076	0.058
02:00PM-02:30PM	0.067	0.081	0.087	0.091	0.091	0.085	0.076	0.053
02:30PM-03:00PM	0.066	0.079	0.086	0.089	0.088	0.082	0.072	0.052
03:00PM-03:30PM	0.072	0.084	0.090	0.090	0.088	0.079	0.067	0.043
03:30PM-04:00PM	0.087	0.093	0.091	0.084	0.080	0.067	0.055	0.041

Table 4.3: Mean transfer entropy estimates for non-overlapping 30-minute daily trading windows at each sample rate.

	1 SECOND	2 SECONDS	3 SECONDS	5 SECONDS	6 SECONDS	10 SECONDS	15 SECONDS	30 SECONDS	1 MINUTE	2 MINUTES
MEAN	0.253	0.305	0.328	0.341	0.342	0.331	0.309	0.273	0.268	0.294
STD	0.037	0.031	0.024	0.015	0.014	0.020	0.027	0.033	0.030	0.026
MIN	0.205	0.260	0.291	0.317	0.312	0.271	0.227	0.193	0.200	0.235
1%	0.207	0.260	0.294	0.318	0.315	0.272	0.233	0.200	0.201	0.239
25%	0.224	0.281	0.309	0.330	0.332	0.323	0.296	0.257	0.250	0.278
50%	0.251	0.302	0.326	0.338	0.339	0.336	0.313	0.280	0.268	0.296
75%	0.266	0.318	0.340	0.351	0.350	0.343	0.331	0.300	0.290	0.315
99%	0.367	0.385	0.383	0.379	0.376	0.367	0.345	0.325	0.321	0.337
MAX	0.373	0.389	0.384	0.380	0.383	0.369	0.345	0.326	0.324	0.337

Table 4.4: This table presents summary statistics across the 61 morning trading periods (9:30 am - 11:40 am) during the first-quarter of 2018 for weighted out-degree scaled by the amount of firms in the network for measures of transfer entropy at each sample rate.

Sample Rate	**1 second**	**2 seconds**	**3 seconds**	**5 seconds**	**6 seconds**	**10 seconds**	**15 seconds**	**30 seconds**	**1 minute**	**2 minutes**
Intercept	0.039***	0.060***	0.077***	0.091***	0.094***	0.091***	0.085***	0.071***	0.055***	0.043***
	(0.001)	(0.002)	(0.002)	(0.002)	(0.002)	(0.002)	(0.002)	(0.001)	(0.001)	(0.001)
Wtd_InDegree$_{t-1}$	0.602***	0.484***	0.385***	0.296***	0.277***	0.251***	0.222***	0.124***	0.039***	0.020***
	(0.012)	(0.012)	(0.013)	(0.014)	(0.013)	(0.011)	(0.010)	(0.008)	(0.006)	(0.006)

Observations	29,642	29,642	29,642	29,642	29,642	29,642	29,642	29,642	29,642	29,642
Adjusted R2	0.361	0.234	0.148	0.087	0.077	0.063	0.049	0.015	0.001	0.000

Table 4.5: This table presents the results of ordinary least squares models examining auto-correlation in weighted daily in-degree (Wtd_InDegree) scaled by the number of firms in the network. We use all 130-minute morning trading windows (9:30 am to 11:40 am) during the 2018 earnings season to measure persistence from trading day t to $t+1$. Standard errors clustered by firm appear in parentheses below each coefficient. *, **, and *** indicate significance at the 10%, 5%, and 1% levels, respectively.

Sample Rate	**1 second**	**2 seconds**	**3 seconds**	**5 seconds**	**6 seconds**	**10 seconds**	**15 seconds**	**30 seconds**	**1 minute**	**2 minutes**
Intercept	0.015***	0.019***	0.023***	0.029***	0.033***	0.039***	0.037***	0.032***	0.027***	0.025***
	(0.004)	(0.005)	(0.006)	(0.008)	(0.008)	(0.009)	(0.009)	(0.006)	(0.004)	(0.003)
Wtd_OutDegree$_{t-1}$	0.847***	0.838***	0.819***	0.773***	0.748***	0.680***	0.659***	0.605***	0.535***	0.419***
	(0.038)	(0.043)	(0.047)	(0.057)	(0.061)	(0.074)	(0.076)	(0.073)	(0.069)	(0.061)
Observations	29,642	29,642	29,642	29,642	29,642	29,642	29,642	29,642	29,642	29,642
Adjusted R2	0.702	0.689	0.658	0.587	0.551	0.456	0.43	0.366	0.287	0.177

Table 4.6: This table presents the results of ordinary least squares models examining auto-correlation in weighted daily out-degree (Wtd_OutDegree) scaled by the number of firms in the network. We use all 130-minute morning trading windows (9:30 am to 11:40 am) during the 2018 earnings season to measure persistence from trading day t to $t+1$. Standard errors clustered by firm appear in parentheses below each coefficient. *, **, and *** indicate significance at the 10%, 5%, and 1% levels, respectively.

Sample Rate	1sec	2sec	3sec	5sec	6sec	10sec	15sec	30sec	1min	2min
Model	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
EA_{iwt} * $Morning_{wt}$	-0.013***	-0.024***	-0.030***	-0.032***	-0.033***	-0.031***	-0.026***	-0.021***	-0.015***	-0.014***
	(0.002)	(0.003)	(0.003)	(0.003)	(0.003)	(0.003)	(0.003)	(0.002)	(0.002)	(0.002)
EA_{iwt} * $Morning_{wt}$ * $Abs\_Surprise_{iwt}$	1.672**	2.627***	2.995***	3.117***	3.094***	2.920***	2.510***	2.525***	2.379***	1.686**
	(0.660)	(0.802)	(0.853)	(0.899)	(0.922)	(0.923)	(0.858)	(0.744)	(0.660)	(0.762)
EA_{iwt} * $Afternoon_{wt}$	0.006***	0.006***	0.006***	0.007***	0.006***	0.007***	0.005***	0.002	-0.002	-0.007***
	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.002)	(0.002)	(0.002)	(0.002)	(0.002)
EA_{iwt} * $Afternoon_{wt}$ * $Abs\_Surprise_{iwt}$	0.672**	0.882**	0.946**	0.784*	0.755	0.715	0.795	0.649	1.019*	1.058
	(0.321)	(0.408)	(0.480)	(0.476)	(0.533)	(0.594)	(0.571)	(0.596)	(0.595)	(0.823)
EA_{iwt} * $Evening_{wt}$	0.008***	0.009***	0.008***	0.007***	0.007***	0.005***	0.003**	-0.001	-0.001	-0.006***
	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.002)	(0.002)
EA_{iwt} * $Evening_{wt}$ * $Abs\_Surprise_{iwt}$	-0.045	0.0001	0.031	0.174	-0.022	0.102	0.008	0.829*	0.382	0.716
	(0.366)	(0.375)	(0.389)	(0.389)	(0.417)	(0.436)	(0.493)	(0.499)	(0.587)	(0.711)
$Share\_Turnover_{it}$	-0.171***	-0.304***	-0.360***	-0.395***	-0.409***	-0.394***	-0.393***	-0.345***	-0.246***	-0.242***
	(0.066)	(0.074)	(0.075)	(0.075)	(0.076)	(0.072)	(0.070)	(0.063)	(0.057)	(0.050)
Abs_RET_{it}	0.065***	0.094***	0.120***	0.150***	0.159***	0.176***	0.177***	0.179***	0.120***	0.200***
	(0.012)	(0.014)	(0.015)	(0.016)	(0.016)	(0.016)	(0.016)	(0.015)	(0.014)	(0.013)
$Morning_{wt}$	0.026***	0.029***	0.030***	0.029***	0.027***	0.023***	0.019***	0.015***	0.014***	0.014***
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
$Evening_{wt}$	0.012***	0.010***	0.007***	0.003***	-0.0002	-0.005***	-0.009***	-0.010***	-0.007***	-0.005***
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
Fixed Effects	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week
Observations	89,322	89,322	89,322	89,322	89,322	89,322	89,322	89,322	89,322	89,322
Adjusted R^2	0.761	0.774	0.760	0.722	0.703	0.646	0.620	0.600	0.537	0.417

Table 4.7: This table presents results of ordinary least squares regressions where the dependent variable is the weighted out-degree scaled by number of firms in the network. We test for earnings surprise as a determinant of daily outgoing information transfers. Standard errors clustered by firm appear in parentheses below each coefficient. *, **, and *** indicate significance at the 10%, 5%, and 1% levels, respectively.

Sample Rate	1sec	2sec	3sec	5sec	6sec	10sec	15sec	30sec	1min	2min
Model	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
EA_{iwt} * $Morning_{wt}$	0.014***	0.013***	0.012***	0.011***	0.013***	0.008**	0.008**	0.014***	0.019***	0.006
	(0.002)	(0.002)	(0.003)	(0.003)	(0.003)	(0.004)	(0.004)	(0.005)	(0.006)	(0.008)
EA_{iwt} * $Morning_{wt}$ * $Abs\_Surprise_{iwt}$	1.619**	1.531*	1.066	1.134	0.121	0.213	1.907	-1.06	2.655	-0.816
	(0.803)	(0.852)	(1.084)	(0.942)	(1.202)	(1.487)	(1.627)	(2.530)	(2.390)	(3.893)
EA_{iwt} * $Afternoon_{wt}$	0.006***	0.010***	0.014***	0.018***	0.020***	0.019***	0.017***	0.012**	0.001	-0.003
	(0.002)	(0.002)	(0.002)	(0.003)	(0.003)	(0.003)	(0.004)	(0.005)	(0.006)	(0.008)
EA_{iwt} * $Afternoon_{wt}$ * $Abs\_Surprise_{iwt}$	0.821	0.789	0.365	-0.287	0.922	0.18	1.932	2.072	2.737	4.522
	(0.635)	(0.710)	(1.067)	(1.121)	(1.033)	(1.289)	(1.598)	(2.087)	(2.145)	(3.070)
EA_{iwt} * $Evening_{wt}$	0.008***	0.013***	0.014***	0.012***	0.015***	0.017***	0.018***	0.009*	0.013**	0.011
	-0.002	-0.002	-0.002	-0.002	-0.003	-0.003	-0.004	-0.005	-0.006	-0.008
EA_{iwt} * $Evening_{wt}$ * $Abs\_Surprise_{iwt}$	0.826	0.1	0.731	2.201**	-0.392	-0.72	0.7	5.270***	0.974	0.91
	(0.613)	(0.679)	(0.932)	(1.078)	(1.141)	(1.335)	(1.608)	(1.960)	(2.684)	(3.591)
$Share\_Turnover_{it}$	0.538***	0.462***	0.354***	0.148**	0.064	-0.075	-0.196***	-0.087	0.084	0.115
	(0.079)	(0.079)	(0.074)	(0.065)	(0.060)	(0.052)	(0.063)	(0.058)	(0.075)	(0.094)
Abs_RET_{it}	0.026**	0.007	-0.023	-0.060***	-0.081***	-0.126***	-0.138***	-0.143***	-0.196***	-0.159***
	(0.011)	(0.013)	(0.015)	(0.017)	(0.018)	(0.021)	(0.024)	(0.030)	(0.035)	(0.050)
$Morning_{wt}$	0.025***	0.029***	0.029***	0.028***	0.027***	0.023***	0.018***	0.015***	0.013***	0.014***
	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)
$Evening_{wt}$	0.012***	0.009***	0.007***	0.003***	-0.0002	-0.006***	-0.009***	-0.010***	-0.008***	-0.006***
	(0.000)	(0.000)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)
Fixed Effects	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week	Firm, Week
Observations	89,322	89,322	89,322	89,322	89,322	89,322	89,322	89,322	89,322	89,322
Adjusted R2	0.521	0.42	0.338	0.259	0.244	0.216	0.182	0.107	0.042	0.016

Table 4.8: This table presents results of ordinary least squares regressions where the dependent variable is the weighted out-degree scaled by number of firms in the network. We test for earnings surprise as a determinant of daily incoming information transfers. Standard errors clustered by firm appear in parentheses below each coefficient. *, **, and *** indicate significance at the 10%, 5%, and 1% levels, respectively.

α **levels**	**1**	**0.5**	**0.4**	**0.375**	**0.35**	**0.325**	**0.3**	**0.2**
Jan 31st	(498, 247506)	(498, 41802)	(450, 3082)	(361, 1353)	(257, 592)	(153, 240)	(85, 110)	(3, 2)
Feb 1st	(498, 247506)	(498, 41244)	(464, 3181)	(414, 1574)	(324, 804)	(219, 418)	(145, 219)	(8, 6)
Mar 13th	(494, 243542)	(494, 47139)	(487, 5010)	(459, 2568)	(402, 1335)	(306, 714)	(215, 396)	(34, 40)

Table 4.9: This table shows the number of nodes and edges in network backbones at various α levels for January 31st (Jan 31st), February 1st (Feb 1st), and March 13th (Mar 13th). Each table cell has the number notes followed by the number of edges. For example, Jan 31st has a node count of 498 and an edge count of 247,506 when $\alpha = 1$.

Chapter 5

Conclusion

5.1 Summary and Conclusion

In Chapter 2, we used random forest trees in a supervised learning paradigm to predict the annual direction of profitability for firms with minimal information. We generated out-of-sample predictions of directional changes (increases or decreases) in five profitability measures: return on equity (ROE), return on assets (ROA), return on net operating assets (RNOA), cash flow from operations (CFO), and free cash flow (FCF) from 2011-2016. We found that the classification accuracy for each measure outperformed random walk models.

We can further improve the classification accuracies by selecting a model that yields better performance but offers less interpretability. Given that we focus on a minimal set of features in this research, we can incorporate additional features for additional performance gains. With more observations, we can also extend the methodology outlined in this paper to a regression setting and make predictions about the magnitudes of profitability.

In Chapter 3, we develop a new software package to estimate bi-variate transfer entropy (TE). Our method scaled better to larger data than existing implementations. On large data, our implementation, PyIF, is up to 1072 times faster utilizing GPUs and up to 181 times faster utilizing CPUs than existing implementations that estimate bi-variate TE. For future work, we plan to improve the existing code base to improve the computational performance of PyIF. In addition to this, we plan to implement additional estimators to estimate bi-variate TE.

In Chapter 4, we introduced a new approach to examine information transfers around earnings announcements. We studied the network effect of earnings announcements by constructing daily networks of pairwise cross-firm information transfers. Our approach to construct this network relies on non-parametric estimates in equity prices with measures of transfer entropy drawn from information theory. We provide evidence that earnings information produced by a single firm flows to other firms across industries. Results from our tests showed that cross-firm links are substantially stronger for firms on days with releases of earnings information and for firms with more unexpected earnings news.

Further, we found that communities of firms in the network also form between firms with links that are not adequately captured by characteristics as defined by the existing literature. We plan to develop community detection algorithms to work in very dense weighted networks such as our information transfer networks as a future work.

References

Vic Anand, Robert Brunner, Kelechi Ikegwu, and Theodore Sougiannis. Predicting profitability using machine learning. *SSRN*, Oct 2019. URL `https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3466478`.

Matthew Anderson and Jeff McMullin. Detecting information flows in markets, Nov 2018.

S. Arya and D. Mount. Ann: library for approximate nearest neighbor searching. 1998.

Sp Baginski. Intraindustry information transfers associated with management forecasts of earnings. *Journal of Accounting Research*, 25(2):196–216, 1987. URL `https://EconPapers.repec.org/RePEc:bla:joares:v:25:y:1987:i:2:p:196-216`.

Ray Ball and P Brown. Empirical evaluation of accounting income numbers. *Journal of Accounting Research*, 6(2):159–178, 1968.

Monica Billio, Mila Getmansky, Andrew Lo, and Loriana Pelizzon. Econometric measures of connectedness and systemic risk in the finance and insurance sectors. *Journal of Financial Economics*, 104(3):535–559, 2012. URL `https://EconPapers.repec.org/RePEc:eee:jfinec:v:104:y:2012:i:3:p:535-559`.

Terry Bossomaier, Lionel Barnett, Michael Harré, and Joseph Lizier. *An Introduction to Transfer Entropy*. 01 2016. doi: 10.1007/978-3-319-43222-9.

Mark T. Bradshaw, Michael S. Drake, James N. Myers, and Linda A. Myers. A re-examination of analysts' superiority over time-series forecasts of annual earnings. *Review of Accounting Studies*, 2012. URL `https://doi.org/10.1007/s11142-012-9185-8`.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

Francois Brochet, Kalin Kolev, and Alina Lerman. Information transfer and conference calls. *Review of Accounting Studies*, 23(3):907–957, 2018. URL `https://EconPapers.repec.org/RePEc:spr:reaccs:v:23:y:2018:i:3:d:10.1007_s11142-018-9444-4`.

Robert Brunner, Kelechi Ikegwu, Bryce Schonberger, and Jeff McMullin. Cross-firm information transfers during earnings season: A network approach, April 2021.

John Y. Campbell, Andrew W. Lo, and A.Craig MacKinlay. *The Econometrics of Financial Markets*. Princeton University Press, 1997. ISBN 9780691043012. URL http://www.jstor.org/stable/j.ctt7skm5.

Vitor Cerqueira, Luis Torgo, and Igor Mozetic. Evaluating time series forecasting models: An empirical study on performance estimation methods, 2019.

Prithwiraj Choudhury, Ryan T. Allen, and Michael G. Endres. Machine learning for pattern discovery in management research. *Strategic Management Journal*, 42(1):30–57, 2021. doi: https://doi.org/10.1002/smj.3215. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/smj.3215.

Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111, Dec 2004. doi: 10.1103/PhysRevE.70.066111. URL https://link.aps.org/doi/10.1103/PhysRevE.70.066111.

Greg J. Clinch and Norman A. Sinclair. Intra-industry information releases: A recursive systems approach. *Journal of Accounting and Economics*, 9(1):89–106, 1987. ISSN 0165-4101. doi: https://doi.org/10.1016/0165-4101(87)90018-8. URL https://www.sciencedirect.com/science/article/pii/0165410187900188.

Andrew Collette. *Python and HDF5*. O'Reilly, 2013.

Francis X. Diebold and Kamil Yilmaz. On the Network Topology of Variance Decompositions: Measuring the Connectedness of Financial Firms. Koç University-TUSIAD Economic Research Forum Working Papers 1124, Koc University-TUSIAD Economic Research Forum, October 2011. URL https://ideas.repec.org/p/koc/wpaper/1124.html.

Wilfrid J Dixon and Kareb K Yuen. Trimming and winsorization: A review. *Statistische Hefte*, 15(2-3):157–170, 1974.

George Foster. Intra-industry information transfers associated with earnings releases. *Journal of Accounting and Economics*, 3(3):201–232, 1981. ISSN 0165-4101. doi: https://doi.org/10.1016/0165-4101(81)90003-3. URL https://www.sciencedirect.com/science/article/pii/0165410181900033.

Jerry C. Y. Han and J. Wild. Unexpected earnings and intraindustry information transfers - further evidence. *Journal of Accounting Research*, 28:211–219, 1990.

Weiqiang Hang and Timothy Banks. Machine learning applied to pack classification. *International Journal of Market Research*, 61(6):601–620, 2019. doi: 10.1177/1470785319841217.

Rebecca N. Hann, Heedong Kim, and Yue Zheng. Intra-industry information transfers: evidence from changes in implied volatility around earnings announcements. *Review of Accounting Studies*, 24(3):927–971, September 2019. doi: 10.1007/s11142-019-9487-1. URL https://ideas.repec.org/a/spr/reaccs/v24y2019i3d10.1007_s11142-019-9487-1.html.

Charles R. Harris, K. Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020. doi: 10.1038/s41586-020-2649-2.

Craig W. Holden and Stacey Jacobsen. Liquidity measurement problems in fast, competitive markets: Expensive and cheap solutions. *Journal of Finance*, 69(4):1747–1785, 2014. URL `https://EconPapers.repec.org/RePEc:bla:jfinan:v:69:y:2014:i:4:p:1747-1785`.

Kewei Hou, Mathijs A. van Dijk, and Yinglei Zhang. The implied cost of capital: A new approach. *Journal of Accounting and Economics*, 53(3):504 – 526, 2012. ISSN 0165-4101. doi: https://doi.org/10.1016/j.jacceco.2011.12.001. URL `http://www.sciencedirect.com/science/article/pii/S0165410111000966`.

K. M. Ikegwu, J. Trauger, J. McMullin, and R. J. Brunner. Pyif: A fast and light weight implementation to estimate bivariate transfer entropy for big data. In *2020 SoutheastCon*, pages 1–6, 2020. doi: 10.1109/SoutheastCon44009.2020.9249650.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R.* Springer, 2013. URL `https://faculty.marshall.usc.edu/gareth-james/ISL/`.

A. Kaiser and T. Schreiber. Information transfer in continuous processes. *Physica D, v.166, 43-62 (2002)*, 166, 06 2002. doi: 10.1016/S0167-2789(02)00432-3.

Shiraj Khan, Sharba Bandyopadhyay, Auroop Ganguly, Sunil Saigal, David Erickson, Vladimir Protopopescu, and George Ostrouchov. Relative performance of mutual information estimation methods for quantifying the dependence among short and noisy data. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 76:026209, 09 2007. doi: 10.1103/PhysRevE.76.026209.

M. Koning and C. Smith. *Decision Trees and Random Forests: A Visual Introduction for Beginners.* Amazon Digital Services LLC - Kdp Print Us, 2017. ISBN 9781549893759. URL `https://books.google.com/books?id=Hi_CtAEACAAJ`.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Phys. Rev. E*, 69:066138, Jun 2004. doi: 10.1103/PhysRevE.69.066138.

Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pages 1–6, 2015.

Michael Lindner, Raul Vicente, Viola Priesemann, and Michael Wibral. Trentool: A matlab open source toolbox to analyse information flow in time series data with transfer entropy. *BMC Neuroscience*, 12(1), 01 2011. doi: 10.1186/1471-2202-12-119. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3287134&tool=pmcentrez&rendertype=abstract. Citations 56.

R. Marschinski and H. Kantz. Analysing the information flow between financial time series. *The European Physical Journal B - Condensed Matter and Complex Systems*, 30:275–281, 2002.

Steve Menard and Luis Nell. "jpype documentation". 2018. URL https://jpype.readthedocs.io/en/latest/.

S. Monahan. Financial statement analysis and earnings forecasting. 2018.

ANN Library: David Mount, Sunil Arya. Transfer Entropy Packge: Ghazaleh Haratinezhad Torbati, and Glenn Lawyer. *TransferEntropy: The Transfer Entropy Package*, 2015. URL https://CRAN.R-project.org/package=TransferEntropy. R package version 1.5.

M. E. J. Newman. *Networks: an introduction.* Oxford University Press, Oxford; New York, 2010. ISBN 9780199206650 0199206651. URL http://www.amazon.com/Networks-An-Introduction-Mark-Newman/dp/0199206651/ref=sr_1_5?ie=UTF8&qid=1352896678&sr=8-5&keywords=complex+networks.

NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020. URL https://developer.nvidia.com/cuda-toolkit.

Chris Olsen and J. Richard Dietrich. Vertical information transfers: The association between retailers' sales announcements and suppliers' security returns. *Journal of Accounting Research*, 23:144–166, 1985. ISSN 00218456, 1475679X. URL http://www.jstor.org/stable/2490695.

Jane A. Ou and Stephen H. Penman. Financial statement analysis and the prediction of stock returns. *Journal of Accounting and Economics*, 11(4):295 – 329, 1989. ISSN 0165-4101. doi: https://doi.org/10.1016/0165-4101(89)90017-7. URL http://www.sciencedirect.com/science/article/pii/0165410189900177.

Milan Paluš, Vladimír Komárek, Zbyn ěk Hrnčíř, and Katalin Štěrbová. Synchronization as adjustment of information rates: Detection from bivariate time series. *Phys. Rev. E*, 63:046211, Mar 2001. doi: 10.1103/PhysRevE.63.046211. URL https://link.aps.org/doi/10.1103/PhysRevE.63.046211.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jason Pellerin. “nose. 2021. URL https://nose.readthedocs.io/en/latest/.

Grace Pownall and Gregory Waymire. Voluntary disclosure choice and earnings information transfer. *Journal of Accounting Research*, 27:85–105, 1989. ISSN 00218456, 1475679X. URL `http://www.jstor.org/stable/2491066`.

Kenneth R.Ahren and Jarrad Harford. The importance of industry links in merger waves. *The Journal of Finance*, 69(2):527–576, 2014. ISSN 00221082, 15406261. URL `http://www.jstor.org/stable/43611161`.

Armin Rigo and Maciej Fijalkowski. "cffi documentation". 2018. URL `https://cffi.readthedocs.io/en/latest/`.

L. Sandoval. Structure of a global network of financial companies based on transfer entropy. *Entropy*, 16:4443–4482, 2014.

Samer Muthana Sarsam, Hosam Al-Samarraie, Ahmed Ibrahim Alzahrani, and Bianca Wright. Sarcasm detection using machine learning algorithms in twitter: A systematic review. *International Journal of Market Research*, 62(5):578–598, 2020. doi: 10.1177/1470785320921779.

T. Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461–464, 2000.

M. Ángeles Serrano, Marián Boguñá, and Alessandro Vespignani. Extracting the multiscale backbone of complex weighted networks. *Proceedings of the National Academy of Sciences*, 106(16):6483–6488, 2009. ISSN 0027-8424. doi: 10.1073/pnas.0808904106. URL `https://www.pnas.org/content/106/16/6483`.

Behrendt Simon, Dimpfl Thomas, Peter Franziska J., and Zimmermann David J. Rtransferentropy — quantifying information flow between different time series using effective transfer entropy. *SoftwareX*, 10(100265):1–9, 2019. URL `https://doi.org/10.1016/j.softx.2019.100265`.

J. David Spiceland, James F. Sepe, and Mark Nelson. ntermediate accounting. chapter 1. McGraw-Hill Education, 2018.

James Stone. *Information Theory: A Tutorial Introduction*. 02 2015. ISBN 978-0956372857. doi: 10.13140/2.1.1633.8240.

Jacob Thomas and Frank Zhang. Overreaction to intra-industry information transfers? *Journal of Accounting Research*, 46:909–940, 09 2008. doi: 10.2139/ssrn.1028285.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

CLARE WANG. Accounting standards harmonization and financial statement comparability: Evidence from transnational information transfer. *Journal of Accounting Research*, 52(4):955–992, 2014. doi: https://doi.org/10.1111/1475-679X.12055. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/1475-679X.12055.

Wharton School. Wharton research data services, 1993. data retrieved from Wharton research data services, https://wrds-web.wharton.upenn.edu/wrds/.

Paul L. Williams and Randall D. Beer. Generalized measures of information transfer. *ArXiv*, abs/1102.1507, 2011.

Patricia Wollstadt, Joseph T. Lizier, Raul Vicente, Conor Finn, Mario Martinez-Zarzuela, Pedro Mediano, Leonardo Novelli, and Michael Wibral. Idtxl: The information dynamics toolkit xl: a python package for the efficient analysis of multivariate information dynamics in networks. *Journal of Open Source Software*, 4(34):1081, 2019. doi: 10.21105/joss.01081. URL https://doi.org/10.21105/joss.01081.

Ying Yang. Research on the optimization of the supplier intelligent management system for cross-border e-commerce platforms based on machine learning. *Information Systems & e-Business Management*, 18(4):851 – 870, 2020. ISSN 16179846.