

FROM THEORY TO PRACTICE: A STOCHASTIC
FRAMEWORK FOR UTILITY-DRIVEN CAPITAL
ALLOCATION

BY

ALIK SOKOLOV

A thesis submitted in conformity with
the requirements for the degree of

Doctor of Philosophy

Graduate Department of Mathematics
University of Toronto

© 2025 Alik Sokolov

To everyone who has supported me on this journey - to my siblings who took care of me when I needed it, and to my wife who took care of me when I didn't. To my son - perhaps this work can contribute to building a better world for him.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Professor Luis Seco, my thesis advisor. Luis is a brilliant mind and a unique presence in the academic community, and I would not be completing my thesis without his support, creativity, and our shared vision.

I am also immensely grateful to my committee members, Dr. Murty and Dr. Nachman, for their feedback, suggestions, and thought they put into my thesis. Their contributions have significantly enhanced the quality of this work.

I also want to thank my family for the support they have provided me, and to anyone who ever patiently listened to me talk about probability, finance, representations or economic utility.

This thesis would not have been possible without the collective support of these individuals, and I am deeply indebted to them for their contributions.

CONTENTS

1	INTRODUCTION	1
1.1	Toward a Utility-Maximizing, Bottom-Up Model	10
1.2	The Full Optimization Problem Set up	11
1.3	Contributions and Structure of the Thesis	11
2	PROBLEM FORMULATION	13
2.1	Literature Review on Multi-Period Optimization and Economic Modelling	14
2.2	High-Level Motivation of the Multi-Period Optimization Problem	16
2.3	Classical Approaches and Their Limitations	20
2.4	A Bottom up Approach - Agentic Formulation	21
2.5	Extended Product Space and Agent Response	23
2.6	Mathematical Results	25
3	CAUSAL DISCOVERY	31
3.1	Introduction	31
3.2	Causal Discovery Literature Review	32
3.3	Methodology	36
3.4	DAG Generation - LLM for Causal Discovery	38
3.5	Applications	42
3.6	Limitations and Future Work	46
3.7	Credit Spread Figures and Tables	48
4	EXTERNALITY-AWARE COMPANY RISK RATINGS	51
4.1	Introduction	51
4.2	NLP and ESG Scoring Literature Review	52
4.3	Definitions and Notation	53
4.4	Approach	53
4.5	Using NLP Model Results for ESG Scoring	60
4.6	Additional Research Avenues	62
5	EXTERNALITY-AWARE PORTFOLIO THEORY	64
5.1	Externalities and Portfolio Theory Literature Review	64
5.2	Framework	65
5.3	Results	68
5.4	Conclusion and Additional Research Avenues	71
6	REPRESENTATION LEARNING FOR FINANCIAL TIME-SERIES DATA	73
6.1	Motivation	73
6.2	Technical Background	74
6.3	Definitions and Notation	77

6.4	Approach and Empirical Implementation	79
6.5	Evaluating the Representations	82
6.6	Significance	83
6.7	Additional Research Avenues	84
7	LEARNING REPRESENTATIONS OF INTERRELATED FINANCIAL TIME-SERIES	85
7.1	Introduction	85
7.2	Representations of Financial Time-Series Data Literature Review	86
7.3	Pre-Training of Financial Time-series Networks	88
7.4	Network Architecture	90
7.5	Applications	92
7.6	Additional Research Avenues	96
8	THE TOPOLOGY OF LEARNED RIFT REPRESENTATIONS	97
8.1	Transformer Models for Financial Time Series Data	97
8.2	Topology Background	98
8.3	Experimental Approach and Results	100
8.4	Conclusion and Additional Research Avenues	101

INTRODUCTION

Mathematical finance has advanced rapidly over the last 50 years, and its techniques are now widely adopted in academia, industry and government. Yet, at the time of writing, financial institutions and governments still lag behind the latest theories and computational capabilities, especially in incorporating developments from other fields such as primary research in machine learning and AI. In recognition of this gap, there is increasing demand for modern mathematical finance to address complexities that extend beyond the traditional domain of financial returns and risk. In an era marked by rapid technological innovation, shifting consumer preferences, urgent global challenges such as climate change and measures of social “cost” (*economic externalities*) have become central to understanding both firm-level performance and broader social welfare. While classical portfolio theory approaches have provided a powerful foundation for risk-return analysis, these approaches have limitations in dealing with dynamic systems such as those created by shifting regulatory regimes, the qualitative nuances of stakeholder benefit and welfare (*utility*), and integrating information over long time horizons. This thesis proposes a unified modeling approach that incorporates modern advances in incorporating textual (and other unstructured) data for making predictions, causal discovery, and non-financial considerations into a bottom-up, *utility-maximizing* framework. The following sections outline the mathematical motivations for this thesis through exploring the low-dimensional tradition of quantitative finance.

The Classical Low-Dimensional Tradition in Quantitative Finance

From the vantage-point of today’s applied mathematics and machine learning, the history of mathematical finance can be viewed as a gradual relaxation of simplifying assumptions, with each new modeling paradigm allowing for richer models and deeper empirical integration of realistic economic behaviour. As we will see below, approaches in use even as of the writing of this thesis, both in industry and academia, are still utilizing a surprisingly small number of parameters despite algorithmic advances in other fields and rising computer power. In addition, these approaches are highly dependent on historical correlation, and fail to incorporate rich information regarding the distribution of future expectations that modern machine learning approaches can provide.

Bachelier and the Birth of Brownian Finance

In 1900 Louis Bachelier’s thesis *Théorie de la Spéculation* [1] introduced the first stochastic model for asset prices:

$$S_t = S_0 + \sigma W_t, \quad t \geq 0, \quad W_t \sim \mathcal{N}(0, t),$$

with constant volatility $\sigma > 0$.

Despite some remaining limitations (e.g. the fact that arithmetic Brownian motion can produce negative prices), this idea was revolutionary at the time. This model considers the movement in asset prices as a random walk in continuous time, i.e. accepting that there is an unpredictable component to price movements. This model considers the process completely random, and we will see how other approaches incorporate the expectations that are available about the future below. A future pay-off $f(S_T)$ could be valued by

$$V_0 = \mathbb{E}\left[f(S_T)\right].$$

All randomness is squeezed into a single real-valued parameter σ ; the state space is one-dimensional and the model is fully characterized by the pair (S_0, σ) .

Black–Scholes: A Two-Parameter Diffusion and a PDE

The next leap (Black–Scholes–Merton, 1973) corrected negativity by moving to geometric Brownian motion

$$\frac{dS_t}{S_t} = \mu dt + \sigma dW_t.$$

Under the risk-neutral measure an option price $v(t, s)$ solves the parabolic PDE

$$\partial_t v + \frac{1}{2}\sigma^2 s^2 \partial_{ss} v + rs \partial_s v - rv = 0, \quad v(T, s) = f(s).$$

Everything now hinges on two real numbers: volatility σ and risk-free rate r . The PDE lives in the two-dimensional space (t, s) ; once boundary conditions and pay-off are fixed, the market is complete and every price estimate has a unique closed-form value. Constant coefficients, however, rule out stochastic volatility, jumps, and liquidity frictions. Once again, the ability to incorporate expectations and information one may have about the probabilistic scenario of how the future may evolve is limited.

Markowitz, Mean–Variance, and Quadratic Programming

In parallel, the work of Harry Markowitz (1952) shifted the focus from pricing to allocation. Let the random return vector of n assets be $R \in \mathbb{R}^n$, with mean $\boldsymbol{\mu} \in \mathbb{R}^n$ and positive-definite covariance matrix $\boldsymbol{\Sigma} \in \mathbb{S}^n$.

The simplex of fully invested, long-only portfolios is

$$\mathcal{W} = \left\{ \boldsymbol{w} \in \mathbb{R}^n : \mathbf{1}^\top \boldsymbol{w} = 1, w_i \geq 0 \right\}.$$

For a prescribed target expected portfolio return μ_p :

$$\mu_p \in \mathbb{R} \quad \left(\mu_p := \mathbb{E}[\boldsymbol{w}^\top R] \right),$$

The classical mean–variance problem becomes:

$$\min_{w \in \mathcal{W}} w^\top \Sigma w \quad \text{s. t.} \quad w^\top \mu = \mu_p.$$

This is a convex quadratic programme whose Karush–Kuhn–Tucker system admits a closed-form solution. In practice the efficient frontier is parameterised by a single Lagrange multiplier λ , while the high-dimensional covariance Σ is often regularised to a low-rank factor form so the number of free parameters remains manageable. The information that is incorporated into this portfolio theory is the historical (linear) correlation matrix of returns. Once again, the pattern of few free parameters used in practice, as well as limited integration of future expectations, remains.

Black–Litterman: Bayesian Shrinkage of Mean–Variance Inputs

Black and Litterman (1992) kept Markowitz’s quadratic programme but introduced a Bayesian combination of two return inputs:

$$\tilde{\mu} = \underbrace{\mu^{\text{mkt}}}_{\text{CAPM equilibrium}} + \tau \Sigma P^\top (P \tau \Sigma P^\top + \Omega)^{-1} (Q - P \mu^{\text{mkt}}),$$

where

$P \in \mathbb{R}^{k \times n}$ encodes k investor views, $Q \in \mathbb{R}^k$, $\Omega = \text{Cov}[Q]$, $\tau > 0$ is a scalar confidence factor.

The posterior mean $\tilde{\mu}$ feeds back into the familiar Markowitz frontier while keeping the optimization “geometry” constant. Despite accommodating subjective beliefs, the model’s free parameter set is still small: one scalar τ plus a handful of view coordinates (Q, Ω) . Once again, high-dimensional data (returns) are projected onto a low-parameter manifold. The beliefs, as well, are captured as a single point estimate of future beliefs, with risk-returns and more realistic “geometries” of future outcomes remaining absent.

The Persistent Pattern

Across Bachelier, Black–Scholes, Markowitz and Black–Litterman, the same template is observed:

Model	State space	Free parameters
Bachelier	$S_t \in \mathbb{R}$	volatility σ
Black–Scholes	$(t, S_t) \in \mathbb{R}_+ \times \mathbb{R}_+$	σ, r
Markowitz	$w \in \Delta^{n-1}$	target return \bar{r} or risk aversion λ
Black–Litterman	$w \in \Delta^{n-1}$	confidence τ , views (P, Q, Ω)

The practitioner analyst still manipulates highly simplified, few-dimensional objects, and the proofs exploit smoothness, convexity, or martingale representation. This tractability was critical in the age of slow and expensive computation and a nascent machine learning field, and has

fuelled seventy-five years of progress. These approaches have four persistent limitations: minimal feature sets, exogenous structure, linear (or affine) transforms, and an inability to automatically exploit the rich future expectations induced by the causal structure of the environment they are modeling (i.e. the economy and the actors within it). These limitations motivate the next section: replacing hand-crafted, low-dimensional parameterizations with data-driven, causally meaningful representations, while continuing to build on the theories and mathematical rigour that enabled the advances outlined above.

From Feature Engineering to Causal Representations

Classical statistical learning in finance, from the capital-asset-pricing regression of the 1970s to today’s boosted trees, is built on an explicitly designed matrix of features (a *feature representation*).

Let $X \in \mathbb{R}^{T \times d}$ be T observations of d hand-crafted features and $y \in \mathbb{R}^T$ the response (return, default flag, ...). A generic supervised model with parameters θ is learned by

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{T} \sum_{t=1}^T \ell(f_{\theta}(X_{t,\bullet}), y_t),$$

where ℓ is a loss and $f_{\theta} : \mathbb{R}^d \rightarrow \mathbb{R}$ the predictor.

Limitation I: Human-Curated Coordinates

FEATURE ENGINEERING. Practitioners define

$$X_t = \left(\underbrace{\text{P/E}}_{x_t^{(1)}}, \underbrace{\text{Momentum}}_{x_t^{(2)}}, \dots, \underbrace{\text{CO}_2 \text{ Intensity}}_{x_t^{(d)}} \right),$$

then hope that linear or non-linear combinations of these d numbers encode all relevant information. This pipeline entails two mathematical frictions:

- (a) **Representation risk:** omitting a latent driver z_t induces omitted-variable bias.
- (b) **Static semantics:** once chosen, coordinates stay fixed; adding new metrics (e.g. Scope-3 emissions) requires manual (human-driven) changes on X .
- (c) **Data-hungry approaches:** X encodes no prior knowledge on the features and their dynamics, relying completely on historical data to learn relationships. This is easy to see since all commonly used algorithms, such as regression, random forests, boosted tree variants, and even artificial neural networks used for tabular tasks are all invariant to permutations of X , as described in the next section.

Limitation II: Order-Indifferent Algorithms

Most off-the-shelf learners are coordinate-permutation invariant. Let $P \in \mathbb{R}^{d \times d}$ be any permutation matrix. For linear regression,

$$f_{\hat{\beta}}(PX_{t,\bullet}) = (PX_{t,\bullet})^\top \hat{\beta} = X_{t,\bullet}^\top (P^\top \hat{\beta}),$$

so re-ordering the columns merely shuffles the coefficients; the optimization objective is unchanged. Analogous symmetry holds for decision trees, random forests and even fully-connected neural networks: weights re-label to compensate.

The key consequence here is that the learned mapping can ignore the *economic meaning* of each feature. A variable labelled “CO₂ intensity” could swap places with “momentum” without affecting the optimizer, even though the two carry very different causal interpretations, and have very different interactions with the target variables in expectation (beyond what can usually be gleaned from recent historical data) given the real-world nature of these relationships.

Automated Causal Discovery as a Solution

The fixed design matrix is replaced by a data-driven, semantics-preserving feature map

$$\mathcal{G}_c : \mathcal{D} \rightarrow \mathbb{R}^m,$$

where \mathcal{D} is the raw data space (financial statements, satellite imagery, text filings). The map \mathcal{G}_c is what the human analyst or data scientist can “hand-craft” via the feature engineering process, but can also be defined algorithmically under two structural assumptions:

1. **Causal graphs:** a directed acyclic graph $G = (V, E)$ with $V = \{1, \dots, m\}$ encodes structural equations $\xi_t^{(i)} = g_i(\xi_t^{(\text{pa}(i))}, \varepsilon_t^{(i)})$, ensuring each latent coordinate $\xi^{(i)}$ has clear parent and child nodes.
2. **Variable semantics:** every node is anchored to a document or signal (“coal capacity”, “carbon price”, ...), so order matters.

Bridging to the Optimization Problems

The causally grounded feature map \mathcal{G}_c feeds into the multi-period optimization of Section 2.2. By preserving semantics, it is possible to:

- propagate policy interventions (taxes, subsidies) along structural edges to forecast their impact on \mathbf{X}_t and \mathbf{E}_t ;
- derive error bounds on value-function approximation that depend on identifiable causal pathways, not on opaque latent vectors;
- explain allocation outcomes to domain experts, which is a critical advantage over black-box factor models.

Automated causal discovery answers both limitations: it eliminates hand-crafted coordinates and the laborious process of creating them, and integrates economic meaning into the learning algorithm.

From Numbers to Meaning

In the history of mathematics and of the practical adoption of its techniques, there have been several moments when the shift in the conceptualization of structures gives rise to deeper analysis. For example, shifting mathematical study from numbers to that of data—shapes, topological spaces, or manifolds has given rise to advances in physics, chemistry and material sciences, and even the deep understanding of tensor algebra that fuels machine learning today. A similar shift is now happening in finance: while one starts with time-series of returns, prices, or volumes, the real utility of these data emerges only when they are connected to the *meaning* they carry, such as the information encoded in \mathcal{G}_c .

For instance, a share price alone reveals little if one ignores that the relevant company’s profit depends on whether it manufactures smartphones, invests in patents, or benefits from certain consumer trends. Real-economy drivers such as R&D, regulatory changes, and shifting consumer preferences, form the causal backbone of financial outcomes, and in a multi-decade horizon, these drivers cannot be reduced to a single numeric series. They are best represented as a *causal graph*, where nodes encode fundamental elements such as company fundamental (e.g., revenue, cost, EBITDA), operational performance (number of phones sold, oil refining efficiency) and their drivers (R&D innovations, quality of products, macroeconomic growth, environment factors), and edges encode relationships or causal links.

This thesis leverages the power of such graphs to tackle modern challenges in portfolio theory, economic modeling, and resource allocation. Traditional frameworks, starting with classical Mean-Variance Optimization [2], have laid out important mathematical principles for single-period asset allocation. Yet they often abstract away real-world complexities: multi-period decision-making, dynamic regulatory environments, and long-horizon externalities such as climate impact or consumer health.

Recent research in machine learning, especially Natural Language Processing (NLP), has shown how textual data (such as data taken from news, social media, company or regulatory reports, etc.) can reveal hidden relationships among companies, their supply chains, and the environment. This includes the author’s research illustrating the capabilities of these techniques in addressing the problems outlined above; see Chapters 4, 5. Beyond text, causal discovery algorithms help one distinguish genuine cause-effect relationships from correlations, guiding robust forecasts of firm fundamentals (e.g., revenues, costs), as in Chapter 3. Representation learning (Chapters 6, 7, 8) serves as an additional means of contextual dimensionality reduction, helping make the optimization problem more tractable. Taken together, these approaches help connect portfolio theory to the meaningful drivers of future expectations, reducing the dependence on human analysts to create feature maps, and allowing the modeling of higher dimensional (and more realistic) parameter spaces.

On Utility Modeling in Finance

Traditional approaches to portfolio construction are usually based on optimizing expected returns for a given level of risk. These methods, however, fall short when the objectives extend beyond financial metrics alone, such as environmental, social, and governance (ESG) issues due to well known failures of free markets to incorporate utility [3] into their “invisible hand” optimization introduced by Adam Smith [4]. Even though shifting societal expectations and knowledge of market participants that such risks can materially affect a company’s long-term financial outcomes can lead to more “pricing in” of economic externalities, modern crises such as climate change or health concerns have led to both concerned consumers and policymakers alike in search of new economic models that account for those concerns.

Initiatives such as the Upright Project [5] illustrate an emerging consensus on the importance of quantifying product-level impacts (positive and negative externalities) to better capture the true utility that companies generate in the real economy. Inspired in part by such initiatives, this thesis treats consumers as the primary drivers of utility, with companies, governments, and other market participants acting as facilitators (or obstacles) in meeting consumer demand.

A Graph-Centric Perspective

When these insights get embedded into a *graph* structure, a twofold benefit is obtained. First, a graph is a universal data format for computational tasks: it is well-supported by algorithms for representation learning, approximate inference, and large-scale optimization. Second, it is a framework comfortable to mathematicians, with well-understood properties, theorems, and tools (spanning a wide range of theoretically sound and widely used approaches from network flows to Markov Decision Processes).

Hence, moving from raw time-series to a graph of economic “drivers” answers two key needs:

1. *Representing real-world relationships*: a causal graph represent cause-and-effect linkages, so incorporating textual data, policy instruments, or dynamic demand is natural in this data structure.
2. *Rigorous structure*: graphs serve as the foundation for formal optimization and control methods, such as Bellman equations, contraction mappings, or multi-objective Pareto fronts.

An illustrative example of the data structure can be visualized in two images. First, consider the “build-up” of revenues for a company such as Apple, seen in figure 1.1. Company forecasting in industry is typically performed by breaking down complex effects such as top-line revenues for a Fortune 100 company such as Apple Inc. into more forecastable building blocks or “drivers”, such as *iPhone Sales* and its “operational” components (such as *Number of iPhones Sold* \times *the Price of an iPhone*). These operational drivers can be further broken down into potentially more varied causal drivers that influence them, as seen in 1.2. Note this is just a logical outline of the node types that empirically materialize within the causal map \mathcal{G}_c .

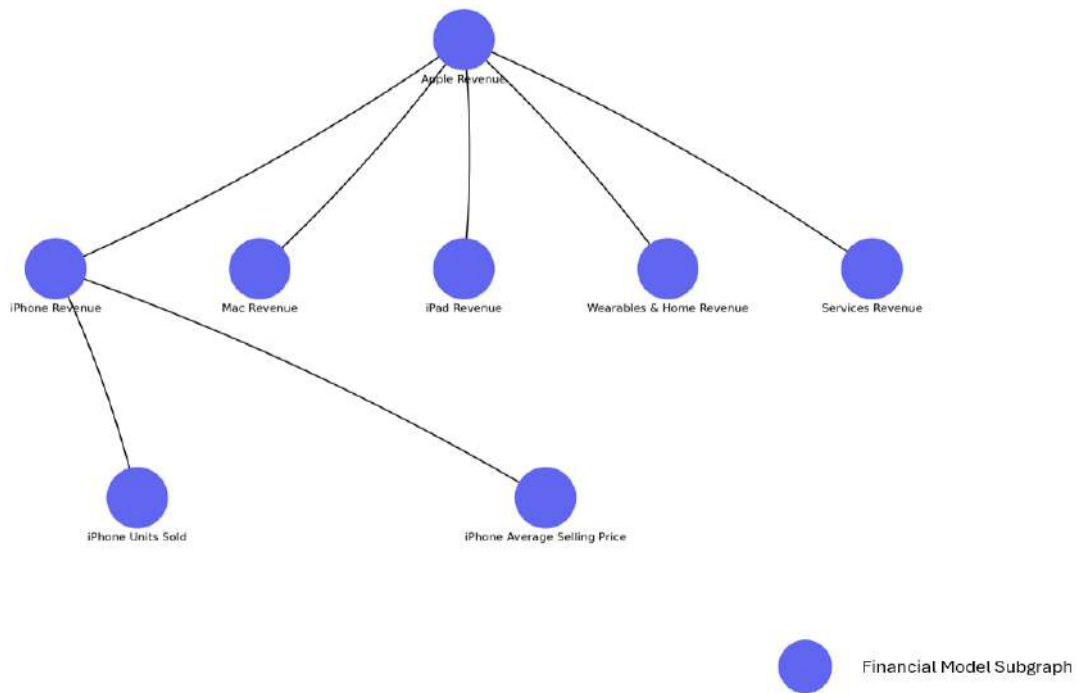


Figure 1.1: Illustrative Company Build-up Graph as Seen in a Typical Financial Model

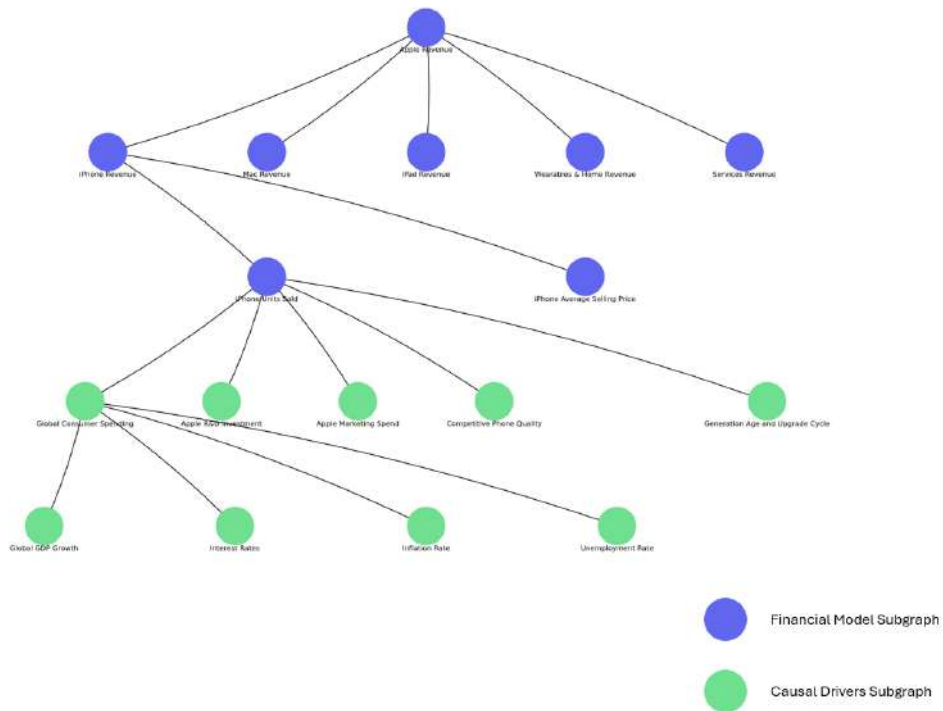


Figure 1.2: Illustrative Expanded Causal Graph for Apple Inc.

This thesis proposed a *multi-period, stochastic control problem* in which each node in the feature map \mathcal{G}_c represents data that updates over time and can itself be forecasted (production

costs, consumer preferences, policy variables) over time and under uncertainty. An optimal policy (or capital allocation strategy) that maximizes a carefully defined utility function is sought, which accounts for both traditional returns and broader externalities.

The optimization formulation takes inspiration from:

1. Dynamic Programming (DP) and Approximate DP from [6, 7, 8] for problem formulation approaches to working with high-dimensional state spaces.
2. Controlled Markov Process Theory from [9, 10] to ensure existence, uniqueness, and stability of solutions over long horizons, as well as handle uncertainty inherent in fundamental and economic forecasting.
3. Multi-Objective Optimization from [11, 12] to capture the trade-offs between financial returns, non-financial considerations, and other externalities.

In short, the aim is to unite the very practical tasks of *causal discovery*, *textual signal extraction*, and *long-horizon forecasting* under a rigorous mathematical umbrella. By the end, it will be shown how capital allocation can be steered not just by historical correlations in financial time-series, but by a structured, graph-based representation of real-world drivers—bridging the gap between computational feasibility and deep economic meaning.

Structure of the Thesis

The remainder of this introduction will preview how each chapter expands on these ideas—from formalizing the full optimization problem (including multi-period constraints and dynamic entry of new firms) to detailing the neural embedding strategies for high-dimensional fundamentals, to providing error bounds and computational techniques that keep the entire approach tractable for large-scale deployment. The goal is to create an outline for a program that applied mathematicians, finance practitioners and policy makers can leverage to inform to make more profitable and socially beneficial capital allocation decisions.

Building Blocks from the Literature

This work draws on a series of prior papers, with each one contributing a foundational piece to the unified framework:

1. **Natural Language Processing and portfolio construction:** papers such as “*Weak Supervision and Black-Litterman for Automated ESG Portfolio Construction*” [13] and “*Building Machine Learning Systems for Automated ESG Scoring*” [13] introduce methods to extract textual signals from unstructured text data and incorporate these signals into portfolio decisions. These frameworks demonstrate how large-scale news or social media data can inform more socially responsible investment choices, aligning capital allocation with broader societal values without necessarily sacrificing performance.
2. **Financial time-series representations and forecasting:** in “*Neural Embeddings of Financial Time Series Data*” [14] and “*RIFT: Pre-Training and Applications for Representations of Interrelated Financial Time-series*” [15], researchers expand the classical toolkit

by learning nuanced representations of financial time-series. Deep learning can capture complex, interrelated dynamics that elude simplistic linear models. Here, these methods will be adapted to fundamental forecasts (e.g., revenues, costs) rather than just asset returns, bridging the gap between raw financial data and real-economy drivers.

3. **Automated causal discovery:** “*Towards Automating Causal Discovery in Financial Markets and Beyond*”[16] highlights how recent advances in machine learning, and namely Large Language Models (LLMs), can uncover causal structures embedded in existing data and domain knowledge. Rather than relying on historical correlations, these frameworks aim to identify causal relationships that inform fundamental variables. For instance, understanding how climate policies (e.g. carbon taxes) affect production costs can significantly enhance long-term projections of company performance.

1.1 TOWARD A UTILITY-MAXIMIZING, BOTTOM-UP MODEL

Modeling Fundamentals

A key departure from purely finance-based approaches is the focus on real-economy fundamentals rather than asset returns alone. As is typical of financial modelling in industry, each company’s forecast includes:

- **Revenues:** Derived from the quantity of goods or services sold and the associated price, which itself may respond to consumer preferences and regulatory changes.
- **Costs:** Incorporating raw material inputs, labor, capital expenditures, and external factors such as climate-induced disruptions or carbon-pricing schemes.
- **Profitability:** Derived from the gap between revenues and costs, which feeds into the company’s longer-term viability and potential expansion decisions.

Crucially, new assets (e.g., future companies, infrastructure projects) may enter the economy to meet unfilled consumer demands, especially over a 10-50 year horizon. Such dynamic entry requires both a forecasting framework that can anticipate emerging technologies and a utility function that guides resource allocation to these new opportunities if they promise to improve overall consumer welfare.

Integrating Externalities Through Fundamentals

While externalities like climate change are not forecasted directly (as a separate time series), they enter the model through changes in costs and demand:

- **Climate impacts:** Higher temperature extremes may elevate operating costs for energy-intensive industries.
- **Policy instruments:** Carbon taxes and regulations shift cost structures in favor of cleaner technologies.

- **Health outcomes:** If poor environmental quality lowers consumer health (or QALYs), consumer utility is reduced, influencing demand for certain products (e.g., green energy, healthcare).

By treating these externalities as part of the core fundamental drivers, the approach remains internally consistent: if climate risks reduce consumer utility, the optimization naturally allocates resources toward alternative goods or services that yield higher overall welfare.

The Utility Function

The *consumer-based utility function* forms the bedrock of the optimization problem. In addition to standard consumption utility—defined as the product of quantity and a utility-per-unit measure—the function incorporates two critical augmentations:

- **Health and quality-of-life adjustments:** Extending beyond market transactions, the utility function accounts for the impact of consumption on health, knowledge, and well-being, building on foundational work that incorporates non-market time use and welfare dimensions into utility analysis [17, 18].
- **Time-preference and discounting:** The model recognizes that utility in future periods may be discounted, following canonical intertemporal optimization frameworks [19], while also acknowledging the policy sensitivity of long-term outcomes to the choice of discount rate, especially in climate and sustainability contexts [20].

All regulatory regimes, taxes, or subsidies feed into this utility objective by reshaping production costs, shifting supply–demand equilibria, and influencing the revealed preferences of consumers.

1.2 THE FULL OPTIMIZATION PROBLEM SET UP

Combining these components, the overarching goal is to maximize expected consumer utility over a 10 to 50 year horizon subject to constraints such as resource availability, production capacity, and policy instruments. The problem is inherently **stochastic**, incorporating uncertainty in factors like technological breakthroughs, economic shocks, climate scenarios and of course the economic agents themselves, including company-level forecasts. Model calibration and validation can be accomplished via:

1. **Historical data and short-term predictive accuracy:** Ensuring the model can replicate near-term fundamentals, i.e. traditional backtests.
2. **Scenario analyses:** Stress-testing different policy regimes and climate trajectories to gauge how resilient the model is to extreme or unexpected conditions, as well as to model distributions of outcomes rather than point-outcomes.

1.3 CONTRIBUTIONS AND STRUCTURE OF THE THESIS

This thesis has the goal of unifying streams of research – textual analysis, neural time-series embeddings, and causal discovery – into a single, operational framework that captures real-

economy outcomes in a more interconnected way. Instead of modeling purely financial risk-return the goal is to move towards modelling societal utility, and offer the reader a blueprint for allocating resources in ways that can mitigate negative externalities and promote long-term welfare.

- **Part 1: Detailed Formulation of the Optimization Problem.** Chapters 1 and 2 out the foundational multi-period, utility-maximizing framework. We begin by reviewing the relevant literature on multi-period optimization and utility theory, then presents the full problem statement (including resource constraints, policy instruments, and stochastic elements). The part also explains why a purely financial approach is insufficient in capturing real-economic impacts and introduces the core objective of incorporating externalities into the utility function.
- **Part 2: Causal Discovery.** Chapter 3 shows how modern causal discovery techniques, including those leveraging Large Language Models, can identify and validate the structural relationships among economic variables. This part references prior research in automated causal discovery [16], illustrating how unveiling causal graphs helps refine forecasts and pinpoints which firm-level fundamentals matter most for the optimization.
- **Part 3: Forecasting Company-Level Fundamentals and Estimating Utility.** Chapters 4 and 5 focus on capturing and modeling externalities using Natural Language Processing (NLP), and discuss approaches for how externalities can be quantified and potentially mapped to consumer-centric utility function. Relevant works, in modelling externalities [21] and [13] are integrated to illustrate how textual data can factor into these long-horizon forecasts and utility assessments.
- **Part 4: Representation Learning for Optimization.** Chapters 6-8 address the high dimensionality and complexity of multi-decade, firm-level modeling by applying advanced representation learning. Works on financial time-series representation learning [14] and [15] are incorporated, showing how low-dimensional embeddings can be learned for large-scale time-series data. These representations can be used to make the optimization problem outlined in Part 1 more tractable, through dimensionality reduction built specifically for financial data.

Ultimately, the goal of this thesis is setting up a model of the economy that is driven solely by historical financial returns but proactively steers capital allocation based on estimates of views of both future economic performance of economic agents, as well as overall utility. The work incorporates classical techniques and modern advances in quantitative finance, machine learning, and causal reasoning while acknowledging the complexities of modern markets and the necessity of tackling global challenges from a bottom-up perspective.

PROBLEM FORMULATION

Classical portfolio theory offers elegant, yet ultimately static, tools for capital allocation. Sustainable investing, climate economics, and high-frequency data reveal two critical gaps:

1. the failure to couple *real-economic fundamentals* with portfolio choice
2. the inability to *internalize externalities* – especially climate and social costs – over multiple decision epochs.

Both are remedied by casting capital allocation as a multi-period, utility-maximization problem with externalities, which is what this chapter describes in detail.

Introducing the Optimization Objective

$$\max_{\{\mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t\}_{t=0}^T} \mathbb{E} \left[\sum_{t=0}^T \beta^t U(\mathbf{X}_t, \mathbf{E}_t, W_t; \theta) \right], \quad 0 < \beta \leq 1$$

Here, β gives a discount factor for utility in future time periods, and X_t is induced by the causal mapping \mathcal{G}_c as described in the introductory chapter.

Mathematical Results

- R1 Forecast-error robustness:** Lemma 2.1 establishes a *linear* $O(\varepsilon)$ upper bound on the deviation of the multi-period utility when each agent’s forecast is perturbed by at most ε , under bi-Lipschitz and compactness assumptions.
- R2 Discrete Policy Feasibility is NP-complete:** Lemma 2.2 (*Policy-Utility*) shows that, once the state, action, and product spaces are discretised and the horizon T is fixed, deciding whether there exists a policy that attains a target discounted utility is NP-complete via a reduction from 0–1 KNAPSACK. This justifies the use of approximation and representation-learning techniques later in the thesis.
- R3 Latent-Space Contraction:** Theorem 2.3 proves that, provided the encoder Φ is *bi-Lipschitz* and the original dynamics G are uniformly Lipschitz in the high-dimensional driver state, the induced map in the low-dimensional latent space is a strict contraction. A unique fixed point therefore exists and iterative algorithms in the latent space converge geometrically.

2.1 LITERATURE REVIEW ON MULTI-PERIOD OPTIMIZATION AND ECONOMIC MODELLING

Multi-Period Optimization Models

Multi-period optimization models extend portfolio choice across multiple stages or time periods, often using dynamic programming and stochastic control techniques. Early foundational work by [6] introduced dynamic programming as a method to solve sequential decision problems, noting the “curse of dimensionality” (the exponential growth of computation with state variables) that has long since constrained multi-period portfolio optimization.

In finance, [2] laid the groundwork for modern portfolio theory with a single-period Mean-Variance Optimization (MVO) framework. In order to generalize their work beyond a one-shot decision, [22], [23], and [24] applied dynamic programming methods to multi-period portfolio selection. These works showed how investors could optimally rebalance portfolios in response to new information, rather than treating portfolio choice as static. An important insight from this early literature was the possibility of *myopic* portfolio policies: under i.i.d. returns and time-additive CRRA utility, the optimal multi-period strategy can reduce to a sequence of single-period optimizations, although the fact that returns are at least partially forecastable (e.g. as seen early in [25]) complicates this. Partial forecastability of returns and increasing ability to model expectations of dynamical systems remains a key motivation for the work in this thesis.

Because exact dynamic programming can be computationally intensive for large state spaces, later research explored approximate or specialized methods. Multi-stage stochastic programming (scenario tree optimization) became popular in asset-liability management contexts [26]. These models choose current decisions by accounting for future scenarios and “recourse actions”, effectively optimizing over a scenario tree (i.e. a graph). Production-based pricing reconnects returns with investment decisions, linking portfolio choice to real investment behaviour. Recent advances emphasize objectives beyond simple return maximization; for instance, goal-based investing frameworks incorporate intermediate or terminal targets (liabilities, income goals), and they include frictional costs (transaction costs, taxes) and illiquidity [27].

Utility Maximization with Externalities

Classical utility maximization typically focuses on an agent’s or investor’s consumption or wealth; however, it does not usually account for *Externalities*, which arise when costs or benefits accrue to parties not directly engaged in the transaction [3], which in turn creates a gap in information for ESG (Environmental, Social, and Governance) investing.

In macroeconomics, [28] introduced the DICE (Dynamic Integrated Climate-Economy) model, integrating a climate externality into a multi-period macro framework. In it, a social planner chooses control variables (e.g., emissions abatement) to maximize utility of consumption while accounting for climate damages. Similarly, other integrated assessment models consider negative externalities from pollution, formalizing how consumption, emission reduction, and technology choice interact over time.

Accounting for externalities has also spurred the rise of sustainable and responsible investing in the world of finance. One approach is to modify investors' utility functions so that they derive utility (or disutility) from positive (or negative) externalities. [29] model green assets that confer non-pecuniary utility benefits on socially conscious investors. Consequently, green firms trade at a premium (lower cost of capital), while "brown" firms trade at a discount. Empirical findings by [30] support this: so-called "sin" stocks with negative social externalities (e.g., tobacco, alcohol) exhibit higher expected returns because some investors avoid them.

Another strand of research imposes constraints or objectives related to ESG outcomes. [31] study how green investment or divestment can alter corporate behavior by raising polluting firms' cost of capital. [32] introduce the concept of an "ESG-efficient frontier," analogous to Markowitz's risk-return frontier, but augmented by a portfolio-level ESG score. Within this framework, each investor trades off expected return and ESG performance, effectively internalizing an externality at the portfolio construction stage.

Real-Economy vs. Financial Models

Traditional quantitative finance methods usually focus on asset prices, returns, and portfolio allocation, whereas the real economy and its study emphasize production, consumption, and policy-driven resource allocation. Traditional finance models, such as CAPM, multi-factor models, or MVO-based portfolio selection, abstract away much of the real-economy detail, focusing solely on maximizing (risk-adjusted) returns. Although factor models (tradition inspired by [33]) partially link returns to fundamental characteristics (size, value, etc.), they do not directly account for production volumes, costs, and specific *company-level* drivers of financial and non-financial performance.

In macroeconomics, the real-economy perspective is exemplified by growth models like [19] or integrated assessment models such as [28], where a social planner optimizes consumption and capital accumulation while subject to resource and technological constraints. [34] introduced *production-based asset pricing*, connecting firms' investment decisions (a real-economy variable) to expected returns (a financial outcome). Similarly, [35] showed how cross-sectional return anomalies like the value premium may stem from production-side frictions, showing the importance of causal linkages back to real economic fundamentals.

The distinction matters because financial-return-driven models can overlook important real-economic factors; a purely financial MVO might, for instance, overweight profitable polluters even though they generate negative externalities, or ignore how resource constraints and climate shocks disrupt production capacity. Even from a pure returns perspective, it may ignore important future expectations (such as an R&D breakthrough) not captured in historical returns data. To bridge these gaps, modern research increasingly integrates real-economy mechanisms (production functions, resource usage, consumer demand) with financial optimization, which helps to align asset allocation with broader social objectives.

Contrast with Classical Approaches

Classical methods like [2] MVO and factor-based models (CAPM, APT, Fama-French) offer a foundation but typically assume:

- A *single-period* or purely static perspective.
- Investors only care about *mean return* and *variance* (or factor loadings).
- No direct accounting for *externalities* or real-economic constraints.
- A lack of connectedness of economic agents (i.e. companies) to drivers of performance, i.e. returns not being linked to real fundamental or economic drivers

Extensions or augmentations are possible. [32] embed ESG preferences into a risk-return optimization, deriving an “ESG-efficient frontier.” [36] propose combining equilibrium returns with subjective investor views within MVO, potentially allowing real-economic considerations to inform the expected return vector. Meanwhile, multi-stage stochastic programming [37] addresses the multi-period limitation but often retains purely financial objectives unless further modified.

Hence, while classical frameworks are computationally tractable and well understood, they are insufficient for dealing with the real-economy context full of externalities that this thesis aims to address. Modern developments layer in several additional considerations:

- **Time-consistency and dynamic programming:** ensuring decisions remain optimal as information updates.
- **Non-financial utility components:** incorporating ESG, health, or climate damages into the utility function.
- **Real-economy modeling:** reflecting production volumes, resource constraints, and emerging companies that might meet future demand.

In this thesis, these insights from the literature above and the author’s own research will be used to formulate a multi-period optimization problem that integrates externalities, aligns with real-economic fundamentals, and extends from classical portfolio optimization to bottom-up economic modelling and optimization.

2.2 HIGH-LEVEL MOTIVATION OF THE MULTI-PERIOD OPTIMIZATION PROBLEM

Drawing on seminal contributions in dynamic programming [6] and multi-period portfolio choice [23, 24], a *discrete-time, stochastic* optimization problem over a horizon of T periods is considered. Unlike the classical one-period Mean-Variance Optimization (MVO) framework by [2], the following formulation incorporates *real-economic fundamentals* and *externalities* directly into both the state dynamics and the objective function, similarly to the evolution of climate-economy integration [28] and ESG-focused modeling [29, 32], where external factors such as environmental damage, societal impact, or governance structures that materially affect long-term welfare began to be seriously taken into account.

States, Decisions, and Externalities

STATE VARIABLES. Let us define:

- \mathbf{X}_t : A vector of *real-economic fundamentals* (e.g. revenues, costs, production capacity, technology levels) at time t . Inspired by production-based asset pricing [34, 35], these fundamentals are treated as driving both the real economy and, indirectly, financial returns.
- \mathbf{E}_t : A vector capturing key externalities such as pollution levels, carbon emissions, or health metrics, generalizing integrated climate-economic state variables [28].
- \mathbf{W}_t : Aggregate wealth or financial resources available for allocation at time t . Although Markowitz’s original MVO [2] only tracked portfolio holdings, this is expanded to include real-economic aspects (similar in spirit to [31] and [30], who emphasize how investor choices influence firm-level actions).

All three vectors above can be induced by \mathcal{G}_c as feature map, reducing the representation risk and labour of human feature curation, while also allowing for linkages from \mathbf{X}_t , \mathbf{E}_t and \mathbf{W}_t to their stochastic as well as recursive drivers over multiple steps of the optimization.

DECISION VARIABLES. At each period, the model must allocate resources across existing companies or new projects, and potentially endogenously create future assets:

- \mathbf{w}_t : The portion of financial resources invested in each *existing* firm’s expansion, new technology, or R&D.
- \mathbf{n}_t : Binary or continuous choices reflecting the creation or scale of *new* firms or infrastructure, building on real-economy expansions akin to those in [19] (for capital accumulation) and multi-stage scenario models [37].
- \mathbf{u}_t : Possible *policy decisions* (tax rates, subsidies) if the model includes a regulatory or social-planner perspective, linking to insights from dynamic climate models [28] and green investment constraints and their interplay with corporate behaviour, i.e. multi-period effects [31].

Objective: Consumer-Centric Utility Maximization

$$\max_{\{\mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t\}} \mathbb{E} \left[\sum_{t=0}^T \beta^t U(\mathbf{X}_t, \mathbf{E}_t; \theta) \right], \quad (2.1)$$

where $0 < \beta \leq 1$ is a discount factor, and $U(\cdot)$ denotes a *consumer-based* utility function. Unlike the *wealth-focused* approach of classical finance [24], the utility function $U(\cdot)$ can encapsulate:

- **Consumption of Goods/Services:** The real value created by the companies (akin to production-based asset pricing [34, 35]).
- **Health and Social Factors:** Negative contributions of externalities (e.g., pollution, public health risks) and positive externalities (cleaner air, advanced infrastructure) as in [32].
- **ESG and Sustainability Preferences:** If investors or policymakers gain non-pecuniary utility from holding “greener” assets or penalize “brown” assets [29, 30].

Dynamics and Constraints

STATE TRANSITIONS. Each period, the economy evolves according to:

$$\mathbf{X}_{t+1} = f(\mathbf{X}_t, \mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t, \omega_t), \quad (2.2)$$

$$\mathbf{E}_{t+1} = g(\mathbf{E}_t, \mathbf{X}_t, \mathbf{w}_t, \mathbf{n}_t, \omega'_t), \quad (2.3)$$

$$\mathbf{W}_{t+1} = \mathbf{W}_t + h(\mathbf{X}_t, \mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t, \omega''_t). \quad (2.4)$$

Here, $\omega_t, \omega'_t, \omega''_t$ represent stochastic shocks (e.g. technological breakthroughs, climate events, demand fluctuations). By specifying f, g, h to include resource constraints, production functions, and externalities, the model generalizes the purely financial process in [2] to a richer macro-financial perspective [28, 35].

Now we detail how \mathcal{G}_c can help model richer state transitions, accounting for the interdependencies in \mathbf{X} , \mathbf{E} and \mathbf{W} . Let

$$\mathcal{G}_c = (V, E), \quad V = V_X \cup V_E \cup V_W,$$

be a directed acyclic graph whose nodes V are partitioned into economic fundamentals V_X , externalities V_E , and aggregate wealth components V_W . For every node $i \in V$ denote by $\text{pa}(i) \subset V$ its parents. The structural equations are

$$\xi_{i,t+1} = F_i(\xi_{\text{pa}(i),t}, \mathbf{a}_t, \omega_{i,t}), \quad i \in V,$$

where $\mathbf{a}_t = (\mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t)$ are the controls and $\omega_{i,t}$ exogenous shocks.

Projection maps. Define linear selectors

$$\pi_X : \mathbb{R}^{|V|} \rightarrow \mathbb{R}^{d_X}, \quad \pi_E : \mathbb{R}^{|V|} \rightarrow \mathbb{R}^{d_E}, \quad \pi_W : \mathbb{R}^{|V|} \rightarrow \mathbb{R},$$

so that the full latent vector $\boldsymbol{\xi}_t = (\xi_{i,t})_{i \in V}$ yields the state variables

$$\mathbf{X}_t = \pi_X \boldsymbol{\xi}_t, \quad \mathbf{E}_t = \pi_E \boldsymbol{\xi}_t, \quad W_t = \pi_W \boldsymbol{\xi}_t.$$

In this formulation \mathcal{G}_c simultaneously (i) *extracts* $\mathbf{X}_t, \mathbf{E}_t, W_t$ via the projections π , and (ii) *governs* their joint time evolution through the structural maps F_i . Thus the graph encodes both the feature representation and the causal dynamics driving the multi-period optimisation problem.

RESOURCE, POLICY, AND CAPACITY CONSTRAINTS. Following the integrated approach of [31] and [32], we incorporate resource and policy constraints. At each decision epoch t the control triple $(\mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t)$ must lie in the admissible set $\mathcal{A}_t(\mathbf{X}_t, \mathbf{E}_t, \mathbf{W}_t)$ defined by

$$(i) \text{ Wealth budget: } \mathbf{1}^\top \mathbf{w}_t \leq W_t \quad (2.5)$$

$$(ii) \text{ Fiscal schedule: } \mathbf{u}_t = \tau(\mathbf{E}_t) - \sigma(\mathbf{X}_t), \quad (2.6)$$

$$(iii) \text{ Capacity update: } \mathbf{X}_{t+1} = f(\mathbf{X}_t, \mathbf{w}_t, \mathbf{n}_t, \mathbf{u}_t, \omega_t) + \mathcal{C}(\mathbf{n}_t), \quad \text{,where} \quad (2.7)$$

- $W_t = \pi_W \boldsymbol{\xi}_t$ is the scalar wealth component of the causal state.
- $\tau(\cdot)$ encodes taxes (positive entries) and $\sigma(\cdot)$ encodes subsidies (negative entries), and
- $\mathcal{C}(\mathbf{n}_t)$ converts the new-investment vector \mathbf{n}_t into additional productive capacity for the next period.

Relating to Existing Literature

CLASSICAL VS. EXTENDED APPROACHES.

- [2] (MVO) laid the ground for portfolio optimization but is *single-period* and does not incorporate externalities.
- [24] extended to continuous-time dynamic portfolio choice, focusing on *maximizing investor wealth* rather than real-economic welfare.
- [28] introduced a climate-economy integrated model, highlighting negative externalities of carbon emissions over multiple periods from a social planner’s viewpoint.
- [29] and [32] emphasize ESG in portfolio choice, revealing how investor preferences for sustainable assets shape market equilibria and capital costs.

The Need for a Realistic Economic Lens

Traditional finance models often treat company fundamentals as exogenous drivers of returns. By contrast, the thesis follows the *production-based asset pricing* perspective [34, 35], acknowledging that investment, resource usage, and technology adoption endogenously affect both firm performance and societal welfare. This resonates with recent ESG-focused research [31, 30] indicating that portfolio allocation decisions can shape firm behavior, and with integrated climate models [28] illustrating how economic activity feeds back into environmental change. This approach has several advantages:

- **Utility:** By allowing for negative/positive externalities in the utility function, the model “prices in” climate or social costs that are typically ignored in purely financial frameworks [3].
- **Scenario Analysis:** Policymakers can evaluate how taxes, subsidies, or regulatory constraints shift production and consumption over time, akin to integrated assessment models [28].
- **Managing Complexity:** Multi-period, high-dimensional state spaces demand approximation (e.g., scenario trees, reinforcement learning). This is an open challenge in large-scale

dynamic models [23, 38], which both automated causal discovery and financial representation learning help address.

The technical approaches involved are elaborated on in subsequent chapters, where Chapter 3 outlines approaches to automated causal discovery to identify the fundamental drivers of \mathbf{X}_t and \mathbf{E}_t , Chapters 4, 5 explore externality-aware forecasting methods for firm-level and macroeconomic variables, and Chapters 6, 7, 8 explore representation learning to handle large-scale, high-dimensional time-series data in a tractable manner.

2.3 CLASSICAL APPROACHES AND THEIR LIMITATIONS

We have already covered constraints and their historical context in the single-period focus and focus on historical returns for modern portfolio theory, as well as limitations in integrating real economic drivers and economic externalities. We cover some additional motivating constraints below to introduce additional components of the proposed framework.

Historical Constraints in Computation and Data

Even if a researcher wanted to build large-scale, bottom-up models in past decades, the lack of computational resources was a practical barrier. Multi-period, high-dimensional dynamic programming has been deemed “intractable” by many [6]. Moreover, data on firm-level fundamentals and externalities was sparse; even where such data existed, synthesizing it into coherent drivers demanded advanced modeling techniques that were not yet available.

Limited Dimensionality Reduction and Causal Understanding

Without robust representation learning, classical models were forced to rely on a handful of factors or heavily aggregated economic indicators [33] curated by human researchers. Even as the number of curated factors grew over time [39], their first-order and more complex causal inter-relatedness have been difficult to elucidate. Historically, models typically relied on linear regressions or correlations, lacking modern AI techniques that can unearth complex, nonlinear causal relationships through representation learning [40]. This thesis advocates for modern advances in AI and machine learning being used to help automate aspects of *causal discovery*, and as incorporating information for how drivers (e.g., macro drivers such interest rates, commodity prices, demand for goods, or technological improvements) and policy changes influence different sectors or companies into economic models.

Modern Advances in AI and Computation

The rise of affordable high-performance computing, coupled with breakthroughs in deep learning and natural language processing, allows one to handle the scale and complexity that once stymied multi-period models [41]. Representation learning (e.g., autoencoders, neural embeddings) reduces dimensionality while preserving relevant structure [40], and large language models (LLMs) now

enable automated or semi-automated *causal discovery* from vast textual corpora [16]. These developments pave the way for a bottom-up, agent-level approach that integrates real-economy fundamentals, externalities, and dynamic policy variables into a unified utility maximization framework.

Bridging Classical and Modern Methods

It is worth noting that classical insights (e.g., diversification principles, factor exposures) still hold tremendous value and are critical to incorporate. Factor-based approaches can guide the initialization of forecasting modules or represent baseline risk premia for different sectors. However, the context has evolved: it is possible to now incorporate exogenous and endogenous drivers, track agent-level production, and “price in” the expected cost or benefit of externalities over time. Consequently, the next sections introduce a more flexible and computationally feasible framework that extends beyond one-period returns, aiming to capture the deeper causal and structural relationships underpinning the real economy.

Having established the limitations of classical approaches and the enabling role of modern AI-driven methods, a modern *Agentic Formulation* that forms the backbone of a bottom-up, multi-period economic model is now presented.

2.4 A BOTTOM UP APPROACH - AGENTIC FORMULATION

A key departure from classical, static models of economic activity is in the adoption of a dynamic, bottom-up framework for representing companies, infrastructure projects, and other economic agents into the proposed framework. Rather than fix the set of agents from the outset (as many static or partial-equilibrium models do), the framework accommodates for an evolving agent set that grows to accommodate new firms or infrastructure whenever certain conditions, such as unmet demand, changes in technology, or policy incentives, make such expansion viable. This is critical to enable modelling across very long time horizons, where the agent universe will naturally evolve.

A Dynamic Set of Agents

Let the economy begin at $t = 0$ with a finite set of agents

$$\mathcal{A}_0 = \{a_1, \dots, a_{N_0}\}.$$

Each a_i is an economic entity whose micro-state $\alpha_{i,t} \in \mathbb{R}^{d_\alpha}$ is a sub-vector of the global causal state $\xi_t \in \mathbb{R}^{|V|}$ generated by the graph \mathcal{G}_c (Section 2.2). Formally we use selector matrices as binary row vectors that extract the relevant entries of ξ_t :

$$\alpha_{i,t} = S_i \xi_t, \quad S_i \in \{0, 1\}^{d_\alpha \times |V|} \text{ a selector matrix.}$$

At each subsequent period $t = 1, \dots, T$ a data- or forecast / expectation-driven entry rule $\mathcal{E}_t(\boldsymbol{\xi}_t) \subset \mathbb{R}^{d_\alpha}$ may add new agents, yielding the evolving set

$$\mathcal{A}_t = \mathcal{A}_{t-1} \cup \left\{ \boldsymbol{\alpha} \in \mathcal{E}_t(\boldsymbol{\xi}_t) \right\}.$$

Hence the appearance of innovative firms or infrastructure projects is endogenous to the causal state encoded in \mathcal{G}_c .

Agent-Level Forecast Functions

For every active agent $a \in \mathcal{A}_t$ we define a differentiable *forecast map*

$$F_a : \mathbb{R}^{d_z} \times \mathbb{R}^{d_\alpha} \longrightarrow \mathbb{R}^{d_f}, \quad (\mathbf{z}_t, \boldsymbol{\alpha}_{a,t}) \mapsto \left(\text{Prod}_{a,t}, \text{Cost}_{a,t}, \dots \right),$$

where

- $\mathbf{z}_t = \Phi(\boldsymbol{\xi}_t) \in \mathbb{R}^{d_z}$ is the *common driver vector*, obtained by a fixed projection $\Phi : \mathbb{R}^{|V|} \rightarrow \mathbb{R}^{d_z}$ of the causal state generated by \mathcal{G}_c ;
- $\boldsymbol{\alpha}_{a,t}$ is the agent's micro-state selector introduced above;
- F_a carries agent-specific parameters $\boldsymbol{\theta}_a$ (technology level, capital efficiency, fundamental and operational performance of companies, ...) that are updated each period

Aggregating Outputs to the Macro State

Because every micro-variable is a node of the causal graph $\mathcal{G}_c = (V, E)$, the global state vector $\boldsymbol{\xi}_t \in \mathbb{R}^{|V|}$ already contains each agent's production and cost entries. Macro aggregates are therefore obtained by *linear projections*, not by an external summation step. Define selector matrices $S^{(p)} \in \{0, 1\}^{1 \times |V|}$ that pick out production nodes associated with product p . Then

$$\text{ProdTot}_t^{(p)} = S^{(p)} \boldsymbol{\xi}_t, \quad p = 1, \dots, N_P,$$

and the entire vector $(\text{ProdTot}_t^{(p)})_p$ is a block of the macro state $\mathbf{X}_t = \pi_X \boldsymbol{\xi}_t$ introduced earlier. Utility aggregation is handled in the objective (2.1): $U(\mathbf{X}_t, \mathbf{E}_t)$ acts as the sole scalar summary of production, costs, and externalities, obviating a separate market-clearing layer.

Implications for Representation Learning and Causal Discovery

Because it is possible to end up with potentially thousands (or more) of individual forecast functions $F_a(\cdot)$, each tied to common drivers and dynamic interactions, the dimensionality of the system can become prohibitive. Construction of \mathcal{G}_c such that drivers are given by $\mathbf{z}_t = \Phi(\boldsymbol{\xi}_t)$ are common across agents (i.e. a single causal graph) is a key unlock for dimensionality reduction. Historically, this posed a major barrier [6], but modern AI techniques mitigate these challenges:

- **Representation Learning:** Neural embeddings and other dimensionality-reduction methods [40] help collapse large feature spaces into low-dimensional latent representations, enabling more tractable optimization.
- **Automated Causal Discovery:** By discerning causal relationships among drivers, it is attainable to reduce the complexity of \mathcal{G}_c and identify which factors most strongly influence specific agents [42, 16]. Beyond dimensionality reduction, these enhanced representations will enhance the interpretability of bottom-up models, and enable the modeling of more complex and realistic “scenario trees”.

Thus, what was once infeasible due to computational constraints or limited modeling tools becomes achievable through the synergy of increased computing power, AI-driven representation learning, and advanced causal inference. These techniques are detailed in Chapters . . .

In the following sections, the fundamental building blocks of the model are expanded on: specifying the multi-period utility framework, integrating externalities, and formulating the objective that maximizes a broad-based notion of social welfare.

2.5 EXTENDED PRODUCT SPACE AND AGENT RESPONSE

The causal graph \mathcal{G}_c delivers the macro state $\mathbf{X}_t, \mathbf{E}_t, W_t$ and the common driver vector $\mathbf{z}_t = \Phi(\boldsymbol{\xi}_t)$. What remains is to clarify what is produced or consumed and how those quantities feed consumer welfare.

Unified Product Vector

Let

$$P = \{p_1, \dots, p_{|P|}\}$$

index a finite list of products. We treat tangible goods (e.g. **Wheat**, **Electricity**) and intangible factors (e.g. **CleanAirIndex**, **MentalHealthIndex**) symmetrically.

Each agent-level forecast $F_a(\mathbf{z}_t, \boldsymbol{\alpha}_{a,t})$ already returns a vector in $\mathbb{R}^{|P|}$ as seen in Section 2.4.

Aggregation by Selector Matrices

Because every micro variable is a node of \mathcal{G}_c , total supply of product p_i is obtained via a selector row $S^{(p_i)} \in \{0, 1\}^{1 \times |V|}$:

$$\text{ProdTot}_t^{(p_i)} = S^{(p_i)} \boldsymbol{\xi}_t, \quad i = 1, \dots, |P|.$$

The stacked vector $\left(\text{ProdTot}_t^{(p_i)}\right)_{i=1}^{|P|}$ is part of $\mathbf{X}_t = \pi_X \boldsymbol{\xi}_t$; no separate summation step is required.

The future expectation-driven entry rule $\mathcal{E}_t(\boldsymbol{\xi}_t)$ defined in Section 2.4 adds new agents whenever a profitability or policy threshold is met.

Consumer Utility

Utility depends on the aggregated global state only:

$$U_t = U(\mathbf{X}_t, \mathbf{E}_t, W_t; \theta) = U(\text{ProdTot}_t, \mathbf{E}_t, W_t; \theta),$$

where ProdTot_t is the product vector selected above. Thus the welfare criterion in Eq. (2.1) already prices externalities through \mathbf{E}_t and intangible product coordinates, so a separate market-clearing layer is unnecessary.

In conclusion, the problem set up with products embedded as coordinates of \mathbf{X}_t and drivers embedded in \mathbf{z}_t , the model maintains fixed dimension, transparent causal structure, and is compatible with the global optimization problem. Assuming the existence of a coherent global causal graph \mathcal{G}_c , this set up when combined with modern machine learning and optimization techniques, and computational resources, provides a potentially tractable set up for the bottom-up utility optimization problem.

Methodological Considerations

REPRESENTATION LEARNING High-dimensional drivers (\mathbf{z}_t) and numerous product categories ($|p|$) can pose computational challenges. Representation learning techniques, such as autoencoders or neural embeddings, offer pathways to reduce dimensionality while retaining essential structure in the data.

CAUSAL DISCOVERY To improve forecast accuracy, it is crucial to isolate the *causal* relationships and commonalities among drivers, especially when intangible goods are involved. Machine learning methods for causal discovery can direct attention to the most influential drivers, mitigating spurious correlations.

REAL-WORLD CONSIDERATIONS FOR THE OPTIMIZATION PROBLEM

- **Unified Treatment of Externalities:** By including intangible “externalities” in the product space, we avoid separate modeling of pollution or other side effects, but put additional pressure on the causal map \mathcal{G}_c to unify a more complex interplay of real-world drivers
- **Heterogeneous Agents:** θ_a allows each agent to respond uniquely, capturing firm-level idiosyncrasies.
- **Complex Calibration:** Realistic simulation demands careful definition of forecasts F_a , calibration data, and the \mathbf{z} -transition. Resource constraints will need to be rigorously modeled to avoid unrealistic expansions in intangible outputs.
- **Modelling in Expectation:** due to uncertainty in all of the variables considered, distributions of outcomes will need to be modeled rather than point outcomes, putting additional pressure on the computational resources required

RECAPPING THE FRAMEWORK This proposed multi-period model unifies tangible goods and intangible externalities within an *expanded product space*, driven by a constant-dimensional state vectors \mathbf{X} , \mathbf{E} , and \mathbf{W} whose values evolve according to economic dynamics and random shocks and whose definitions are induced by the causal graph \mathcal{G}_c . Heterogeneous agents, defined by functions $F_a(\mathbf{z}_t; \theta_a)$, generate or consume these goods, while a single consumer-based utility function ranks outcomes over time. The approach can serve as a foundation for analyzing how factors such as technological shifts, policy changes, and environmental considerations influence long-term welfare in an evolving economy.

2.6 MATHEMATICAL RESULTS

The following section includes several mathematical results relevant to the formulation described above:

- Lemma 2.1 proves the boundedness of error propagation under imperfect forecasts, proving the potential effectiveness of this bottom-up approach
- Proposition 2.2 shows that this problem setup is an NP-Complete problem, motivating the need for effectiveness dimensionality reduction and representation learning
- Lastly, Theorem 2.3 shows that neural embeddings (such as those created by the RIFT model) can be used as a stable latent space to optimization the problem set up

Lemma 2.1 (Bounded Error Propagation Under Imperfect Forecasts). *Consider the discrete-time horizon $t = 0, \dots, T$ ($T < \infty$) of the multi-period model, with a state space z projected from \mathcal{G}_c as a union across inputs across all forecasts F .*

$$\mathbf{z}_{t+1} = G(\mathbf{z}_t, \mathbf{y}_t, \omega_t), \quad \mathbf{y}_t = \left\{ \text{Prod}_t^a \right\}_{a=1}^N.$$

Let the baseline production forecasts

$$F_a : \mathbb{R}^d \longrightarrow \mathbb{R}^{|p|}, \quad a = 1, \dots, N,$$

satisfy $\|F_a(\mathbf{z}) - F_a(\mathbf{z}')\|_1 \leq L_F \|\mathbf{z} - \mathbf{z}'\|$ (Lipschitz in \mathbf{z}).

Assume the perturbed forecasts

$$\tilde{F}_a(\mathbf{z}) = F_a(\mathbf{z}) + \Delta F_a(\mathbf{z}), \quad \|\Delta F_a(\mathbf{z})\|_1 \leq \varepsilon, \quad \forall \mathbf{z},$$

are used by every agent. Denote total production vectors

$$\text{Prod}_t = \sum_{a=1}^N F_a(\mathbf{z}_t), \quad \widetilde{\text{Prod}}_t = \sum_{a=1}^N \tilde{F}_a(\tilde{\mathbf{z}}_t).$$

Assumptions for Lemma 2.1

(A1) ***U Lipschitz.*** $U : \mathbb{R}^{|p|} \times \mathbb{R}^d \rightarrow \mathbb{R}$ satisfies $|U(\mathbf{q}, \mathbf{z}) - U(\mathbf{q}', \mathbf{z}')| \leq L_p \|\mathbf{q} - \mathbf{q}'\|_1 + L_z \|\mathbf{z} - \mathbf{z}'\|$.

(A2) ***G Lipschitz.*** G is (λ_z, λ_y) -Lipschitz, i.e. $\|G(\mathbf{z}, \mathbf{y}, \omega) - G(\mathbf{z}', \mathbf{y}', \omega)\| \leq \lambda_z \|\mathbf{z} - \mathbf{z}'\| + \lambda_y \|\mathbf{y} - \mathbf{y}'\|_1$ for all ω . Put $\lambda := \lambda_z + \lambda_y N L_F$ and assume $\lambda < 1$.

(A3) ***Compactness.*** The trajectories $\{\mathbf{z}_t\}$ and $\{\tilde{\mathbf{z}}_t\}$ remain in a common compact set (guaranteeing bounded U and G and allowing global Lipschitz constants).

Let $V_{\text{orig}} = \sum_{t=0}^T \beta^t U(\text{Prod}_t, \mathbf{z}_t)$ and $\tilde{V} = \sum_{t=0}^T \beta^t U(\widetilde{\text{Prod}}_t, \tilde{\mathbf{z}}_t)$, with discount $0 < \beta < 1$. Then

$$\left| \tilde{V} - V_{\text{orig}} \right| \leq C \varepsilon, \quad C = \frac{(L_p N + L_z K)}{(1 - \beta)}, \quad K := \frac{\lambda_y N}{1 - \lambda}.$$

Proof. First, we consider a one-step production deviation. For any t ,

$$\|\widetilde{\text{Prod}}_t - \text{Prod}_t\|_1 \leq \underbrace{N\varepsilon}_{\text{direct perturbation}} + \underbrace{N L_F \|\tilde{\mathbf{z}}_t - \mathbf{z}_t\|}_{\text{baseline response}} \quad (1)$$

Now, we establish a state-deviation recursion. Apply (A2) with $\mathbf{y}_t = \text{Prod}_t$ and $\tilde{\mathbf{y}}_t = \widetilde{\text{Prod}}_t$:

$$\|\tilde{\mathbf{z}}_{t+1} - \mathbf{z}_{t+1}\| \leq \lambda_z \|\tilde{\mathbf{z}}_t - \mathbf{z}_t\| + \lambda_y \|\widetilde{\text{Prod}}_t - \text{Prod}_t\|_1.$$

Insert (1) and set $A := \lambda_z$, $B := \lambda_y N(L_F + \varepsilon/L_F)$. Because $N\varepsilon \leq \|\widetilde{\text{Prod}}_t - \text{Prod}_t\|_1$, a simpler bound suffices:

$$\|\tilde{\mathbf{z}}_{t+1} - \mathbf{z}_{t+1}\| \leq \lambda \|\tilde{\mathbf{z}}_t - \mathbf{z}_t\| + \lambda_y N \varepsilon, \quad \lambda := \lambda_z + \lambda_y N L_F < 1.$$

With initial equality $\|\tilde{\mathbf{z}}_0 - \mathbf{z}_0\| = 0$, solving this linear recursion gives

$$\|\tilde{\mathbf{z}}_t - \mathbf{z}_t\| \leq \frac{\lambda_y N}{1 - \lambda} \varepsilon = K \varepsilon \quad \forall t \leq T. \quad (2)$$

Consider the utility deviation per period. Using Lipschitz constants (L_p, L_z) and bounds (1)–(2),

$$\left| U(\widetilde{\text{Prod}}_t, \tilde{\mathbf{z}}_t) - U(\text{Prod}_t, \mathbf{z}_t) \right| \leq L_p \|\widetilde{\text{Prod}}_t - \text{Prod}_t\|_1 + L_z \|\tilde{\mathbf{z}}_t - \mathbf{z}_t\|.$$

Apply (1) and (2):

$$\leq L_p (N\varepsilon + N L_F K \varepsilon) + L_z K \varepsilon \leq (L_p N + L_z K) \varepsilon,$$

because $\lambda = \lambda_z + \lambda_y N L_F < 1 \implies L_F K < 1/(1 - \lambda)$ so the coefficients fold into the constant.

Finally, we sum over the entire horizon.

Weighted by β^t and summing,

$$\left| \tilde{V} - V_{\text{orig}} \right| \leq \sum_{t=0}^T \beta^t (L_p N + L_z K) \varepsilon \leq (L_p N + L_z K) \varepsilon \frac{1}{1 - \beta} = C \varepsilon.$$

The constant C is independent of ε , completing the proof. \square

We now work with a discretized version of the economic system of Section 2.5. Concretely, we assume

1. Each macro-driver state \mathbf{z}_t takes values in a finite set \mathcal{Z} ; likewise every component of the expanded product space p and every agent action/production level is drawn from a finite alphabet. This assumption is realistic as they all draw from a finite \mathcal{G}_c , by construction.
2. The planning horizon is an integer $T \geq 1$. Utility is accrued period-by-period and discounted geometrically, giving total utility $\sum_{t=0}^T \beta^t U_t$.

Because all state and action spaces are finite, any admissible policy (a complete sequence of activation or production choices for $t = 0, \dots, T$) can be encoded by a finite bit-string whose length is polynomial in the description size of the instance (numbers of agents, states, products, horizon, and so on, including binned choices for each continuous variable).

We focus on the decision variant

Given a target level U^ , does there exist a policy whose total discounted utility is at least U^* ?*

Below we show that this *Policy-Utility* problem is **NP**-complete. The hardness proof proceeds by a polynomial-time reduction from the classical 0–1 *Knapsack* problem. Note discretization has recently shown to be an important optimization in difficult multi-horizon optimization setups under uncertainty, prominently in poker with the Libratus poker AI [43] using discretization as one of its key optimizations).

Lemma 2.2 (NP-completeness of the finite-horizon policy problem). *Fix an integer horizon $T \geq 1$. After discretizing:*

- the driver state $\mathbf{z}_t \in \mathcal{Z}$ (finite),
- the product space $p = \{p_1, \dots, p_m\}$ (finite),
- every agent action/production choice (finite),

suppose each discrete value is describable with $O(\text{poly}(n))$ bits, where n denotes the overall input length. For a discount factor $0 < \beta \leq 1$ and a target utility $U^* \in \mathbf{Q}$, consider the decision problem

Policy-Utility: $\left\{ \begin{array}{l} \text{Input : economic instance } (\mathcal{Z}, p, \text{agents}, T, \beta, U^*); \\ \text{Question : Does there exist a policy } \pi \text{ (sequence of activation/action choices} \\ \text{for periods } t = 0, \dots, T) \text{ such that } \sum_{t=0}^T \beta^t U_t(\pi) \geq U^* ? \end{array} \right.$

Then *Policy-Utility* is **NP**-complete.

Proof. The proof has two parts: (i) membership in NP and (ii) NP-hardness via reduction from the 0–1 Knapsack problem.

(I) **MEMBERSHIP IN NP.** A policy π consists of finitely many agent-activation or action bits: at each period there are at most $|\mathcal{A}|$ binary activation choices, hence $|\pi| \leq |\mathcal{A}|T = O(\text{poly}(n))$. Given such a policy, we can simulate the discrete dynamics for the fixed horizon T : compute productions, update the driver state via the finite map G , evaluate U_t at each step, and accumulate $\sum_{t=0}^T \beta^t U_t$. Every arithmetic operation is on numbers of size $O(\text{poly}(n))$, where $\text{poly}(n)$ is a polynomial function from \mathbb{R}^n , so verification runs in $O(T \cdot (|\mathcal{Z}| + |p| + |\mathcal{A}|)) = O(\text{poly}(n))$ time. Hence *Policy-Utility* $\in \mathbf{NP}$.

(II) **NP-HARDNESS (REDUCTION FROM 0-1 KNAPSACK).** *Knapsack instance.* We are given n items with weights $w_i \in \mathbb{N}$ and values $v_i \in \mathbb{N}$, a capacity $W \in \mathbb{N}$, and a target value $V^* \in \mathbb{N}$.

Question: does there exist a subset $S \subseteq \{1, \dots, n\}$ with $\sum_{i \in S} w_i \leq W$ and $\sum_{i \in S} v_i \geq V^*$?

We begin by encoding a simple one-period ($T = 1$) economy.

1. **Agents.** For each item i create one candidate agent a_i .
2. **Products.** Define two discrete products **Capacity** and **Value**.
3. **Driver state.** Only one driver value \mathbf{z}_0 is used; no dynamics for $t > 0$ are needed.
4. **Agent production choices.** Each agent has exactly two actions:

$$F_{a_i}^{\text{on}}(\mathbf{z}_0) = (-w_i, +v_i), \quad F_{a_i}^{\text{off}}(\mathbf{z}_0) = (0, 0).$$

Activating a_i consumes w_i units of **Capacity** (hence the negative sign) and produces v_i units of **Value**.

5. **Utility.** Let $M := (n + 1) \max_i v_i$ (polynomial in input length). Define the period-0 utility

$$U(\text{Cap}, \text{Val}) = \begin{cases} \text{Val}, & \text{if } \text{Cap} \geq -W, \\ -M, & \text{otherwise.} \end{cases}$$

The penalty $-M$ ensures any capacity violation drives total utility below every feasible target $U^* \leq n \max_i v_i$.

6. **Discount factor and horizon.** Take any $0 < \beta \leq 1$ and $T = 1$. Total discounted utility is simply U_0 .

Correctness of the reduction.

\Rightarrow If the knapsack instance has a feasible subset S , activate precisely the corresponding agents. Capacity usage satisfies $\sum_{i \in S} w_i \leq W$, so no penalty is incurred, and

$$U_0 = \sum_{i \in S} v_i \geq V^* = U^*.$$

Thus the policy achieves the required utility.

\Leftarrow Conversely, if a policy attains utility at least $U^* = V^*$, the penalty must be avoided (otherwise utility would be $-M < V^*$), so the chosen agents satisfy $\sum_{i \in S} w_i \leq W$. The achieved value $U_0 = \sum_{i \in S} v_i \geq V^*$, giving a feasible knapsack subset.

The transformation from an n -item knapsack instance to the *Policy-Utility* instance uses $O(n)$ numbers of size $O(\log \max\{v_i, w_i, W, V^*\})$ bits—hence polynomial time. Therefore *Policy-Utility* is NP-hard.

CONCLUSION. Parts (i) and (ii) establish *Policy-Utility* \in **NP** and NP-hard, so it is NP-complete. □

Latent-space dynamics and a contraction result

We work with the Euclidean norm $\|\cdot\|$ on every \mathbb{R}^d and \mathbb{R}^k ; all Lipschitz constants are understood with respect to that norm (constants for equivalent norms differ only by a fixed factor).

Theorem 2.3 (Latent-space contraction mapping). *Let the full-state dynamics be*

$$\mathbf{z}_{t+1} = G(\mathbf{z}_t, \mathbf{y}_t, \omega_t), \quad \mathbf{z}_t \in \mathbb{R}^d,$$

where \mathbf{y}_t lies in a compact control set \mathcal{Y} and ω_t in a disturbance set Ω .

Assume

(A1) Bi-Lipschitz representation. *There exists an injective map $\Phi : \mathbb{R}^d \rightarrow \mathbb{R}^k$ ($k \ll d$) such that for all $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^d$*

$$\|\Phi(\mathbf{z}) - \Phi(\mathbf{z}')\| \leq \lambda_\Phi \|\mathbf{z} - \mathbf{z}'\|, \quad \|\Phi^{-1}(\mathbf{h}) - \Phi^{-1}(\mathbf{h}')\| \leq \mu_\Phi \|\mathbf{h} - \mathbf{h}'\|,$$

with finite constants $\lambda_\Phi, \mu_\Phi > 0$. The image $H := \Phi(\mathbb{R}^d)$ is closed, hence complete.

(A2) Uniform Lipschitz dynamics. *There exists $\lambda_G > 0$ such that for every $(\mathbf{y}, \omega) \in \mathcal{Y} \times \Omega$ and all \mathbf{z}, \mathbf{z}'*

$$\|G(\mathbf{z}, \mathbf{y}, \omega) - G(\mathbf{z}', \mathbf{y}, \omega)\| \leq \lambda_G \|\mathbf{z} - \mathbf{z}'\|.$$

(A3) Contraction constant. $\lambda_T := \lambda_\Phi \mu_\Phi \lambda_G < 1$.

Fix any admissible pair $(\mathbf{y}, \omega) \in \mathcal{Y} \times \Omega$ and define the latent update

$$T_{\mathbf{y}, \omega} : H \rightarrow H, \quad T_{\mathbf{y}, \omega}(\mathbf{h}) := \Phi\left(G\left(\Phi^{-1}(\mathbf{h}), \mathbf{y}, \omega\right)\right).$$

Then $T_{\mathbf{y}, \omega}$ is a contraction on the complete metric space $(H, \|\cdot\|)$ with constant λ_T ; i.e.

$$\|T_{\mathbf{y}, \omega}(\mathbf{h}_1) - T_{\mathbf{y}, \omega}(\mathbf{h}_2)\| \leq \lambda_T \|\mathbf{h}_1 - \mathbf{h}_2\|, \quad \forall \mathbf{h}_1, \mathbf{h}_2 \in H.$$

Consequently, by the Banach fixed-point theorem, there exists a unique latent state $\mathbf{h}^* \in H$ satisfying $T_{\mathbf{y}, \omega}(\mathbf{h}^*) = \mathbf{h}^*$; equivalently $\mathbf{z}^* := \Phi^{-1}(\mathbf{h}^*)$ solves the fixed-point equation $\mathbf{z}^* = G(\mathbf{z}^*, \mathbf{y}, \omega)$.

Proof. Let $\mathbf{h}_1, \mathbf{h}_2 \in H$ and $\mathbf{z}_1 = \Phi^{-1}(\mathbf{h}_1)$, $\mathbf{z}_2 = \Phi^{-1}(\mathbf{h}_2)$. Using bi-Lipschitzness of Φ and the Lipschitz property of G we have

$$\begin{aligned}
\|T_{\mathbf{y},\omega}(\mathbf{h}_1) - T_{\mathbf{y},\omega}(\mathbf{h}_2)\| &= \left\| \Phi\left(G(\mathbf{z}_1, \mathbf{y}, \omega)\right) - \Phi\left(G(\mathbf{z}_2, \mathbf{y}, \omega)\right) \right\| \\
&\leq \lambda_\Phi \left\| G(\mathbf{z}_1, \mathbf{y}, \omega) - G(\mathbf{z}_2, \mathbf{y}, \omega) \right\| \\
&\leq \lambda_\Phi \lambda_G \|\mathbf{z}_1 - \mathbf{z}_2\| \\
&\leq \lambda_\Phi \lambda_G \mu_\Phi \|\mathbf{h}_1 - \mathbf{h}_2\| \\
&= \lambda_T \|\mathbf{h}_1 - \mathbf{h}_2\|.
\end{aligned}$$

The constant $\lambda_T < 1$ by (A3), so $T_{\mathbf{y},\omega}$ is a strict contraction on the complete space H . Banach's contraction-mapping principle therefore guarantees a unique fixed point $\mathbf{h}^* \in H$, and applying Φ^{-1} gives the corresponding fixed point \mathbf{z}^* in the full state space. \square

REMARKS.

- If the control \mathbf{y}_t or disturbance ω_t varies over time but always belongs to $\mathcal{Y} \times \Omega$, each step map $T_{\mathbf{y}_t, \omega_t}$ is a contraction with the *same* constant λ_T ; the composition over T periods remains a contraction with rate λ_T^T .
- Assumption (A1) is satisfied, for instance, by an injective linear projection followed by an affine transformation on a compact subset, or by a trained neural encoder that is locally invertible and has bounded Jacobian.

CAUSAL DISCOVERY

This chapter introduces a novel machine learning (ML) framework for causal discovery based on recent advances in Large Language Models (LLMs) and discusses the applications of these causal discovery techniques to investment management. Unlike typical data-driven methods for data discovery, the framework uses the implicit “world knowledge” in state-of-the-art LLMs to automate the expert judgement approach to causal discovery. A key application that is explored in detail is end-to-end causal factor analysis, where the utility of this method is demonstrated in specifying and analyzing detailed causal models for financial markets. This chapter also conducts a comparative analysis, juxtaposing the new approach with conventional methods, to underscore the enhanced capability of the framework in revealing intricate causal dynamics in financial data.

3.1 INTRODUCTION

In recent years, the intersection of machine learning (ML) and causal inference has emerged as a vital research area, promising to unveil deeper insights into the causal structures within complex datasets. Traditional approaches to causal discovery often struggle with the intricacies and non-linear relationships present in real-world data. In addition, machine learning techniques have historically lacked a “world model” which meant an over-reliance on statistical modeling and observed effects and difficulty in overlaying priors regarding causal relationships in an automated fashion. To address these challenges, a novel machine learning framework for causal discovery is introduced, designed to be versatile and applicable across a spectrum of domains, by leveraging recent advances in Large Language Models (LLMs).

Factor investing, which involves identifying and leveraging specific factors or drivers behind asset returns, serves as a robust testbed to showcase the efficacy of the framework. The finance industry, characterized by its complex and dynamic data, demands robust methods for causal analysis to enhance investment strategies and risk management. Unlike conventional correlation-based analyses, this chapter looks at creating alternate systematic approaches for selecting factors for constructing factor models.

The chapter outlines the process of employing this machine learning-driven framework for causal discovery in factor investing. It begins with the formulation of causal models, and applications of LLMs to their formulation (causal discovery). This is followed by an end-to-end experiment for how an LLM-assisted causal discovery approach can aid in creating causal factor models, as well as a demonstration of how the framework can be easily applied to different feature sets.

A key aspect of the end-to-end application is the analysis of factor betas, their stability, and their significance in the causal framework. In order to conduct simulations that assess the causal impact of various factors on asset returns, Do-Calculus was utilized, and as with traditional

causal investing frameworks, these simulations provide insights into the potential effects of market shifts or strategic interventions.

3.2 CAUSAL DISCOVERY LITERATURE REVIEW

Causal modeling in Finance

The field of causal modeling is becoming of greater importance in finance and investment management, addressing the need for formulating robust economic theories to counter-act the ease with which financial strategies that perform well on a backtest but may not generalize well into the future can be found. The advent of machine learning and big data has accelerated the trend of false investment theories being identified in research as shown in [44], and causal methods in finance have been proposed as a general methodology for making the search of investment strategies more robust as seen in [45].

In classical approaches to Factor Investing such as [33] and [46], the authors theorize that simple models of several *factors* can explain components of equity returns. These methods are predicated on hidden hypothesis of causal relationships between the factors used and returns. On the other hand, the betas that are computed in practice using supervised techniques such as Ordinary Least Squares (OLS) are based on correlation and represent an associational rather than causal relationship, and so most of the time it is an unstable estimated value that can result in “spurious” findings in econometrics and finance. As in [45], causal frameworks for factor investing have been proposed in which an explicit causal relationship between factors and returns of assets, formulated as a *Directed Acyclic Graph (DAG)*, is the started point. These frameworks makes the pre-assumed hypotheses of causality falsifiable by causal discovery algorithms or experts’ opinion.

Classical approaches to this problem include data-driven methodologies as seen in [47], where the authors claim that classical feature selection algorithms in quantitative finance like Stepwise regression analysis (SRA), Principal Component Analysis (PCA), Decision Tree (DT), and information gain, are based on correlation and so they can not represent the causal direct influence of features on assets’ returns. They propose the Causal Feature Selection (CFS) algorithm and they show that their causal feature selection method improves feature selection in both accuracy-based and investment metrics.

Causal Discovery in Finance

Causal discovery is a core aspect of causal modeling approaches, and causal discovery in finance holds a particularly important role due to the lack of consensus around causal relationships in economics, and the widespread use of novel datasets and factors.

In [48], the authors take a philosophical view toward the rising number of applications of causal inference in quantitative finance. They argue that these applications may be limited due to the complexity and reflexivity of the financial markets. First, causal graphs can be verified ex-post and models being used ex-ante for prediction need calibration. Second, as financial markets are not a closed system, the causal mechanism in the market may change over time. Also,

due to the reflexivity of the market and different models that financial activists use, the arrow of causation may change in different periods. This increases the need of more automated and dynamic approaches for causal discovery and combinations of “first-principles” and data-driven causal discovery methods; the framework shows and end-to-end automation for both.

In [49], the authors explain the limitations of data-based Causal Discovery algorithms in high-dimension data sets, which is the case in Finance, due to the reduction of accuracy and increase of computation complexity. They mention two main approaches of Causal Discovery algorithms: Structure learning approaches and Direction approaches learning. They classify the structure learning approaches, into two main groups *constraint-based methods* like PC and FCI and *search & score methods* like Bayesian Information Criterion (BIC). The problem with these methods is that they only identify the DAG up to Markov equivalence class, the Conditional Independence (IC) test that these methods are based on, in high-dimensional data becomes extremely difficult, and searching space of the DAG becomes super-exponential in the number of variables. For the directional approaches learning, that are based on additive noise models (ANM), they mention two *LINGAM* and *Information-Geometric Causal Inference* (IGCI) algorithms that decrease the time complexity of Causal Discovery but work only on low-dimensional data sets. To overcome the curse of dimension in the Causal Discovery method they introduce a three-step algorithm, the *Causal Discovery in High Dimension* (CDHL).

In a more recent attempt for causal discovery in a high-dimensional data set, in [50], the authors remind the necessity of causal structure in Causal Machine Learning for simulating the data generation process. They also mention the difficulty of the Causal Discovery task for high-dimensional data. As a solution, they propose imposing a prior on the structure of the DAG from prior knowledge, expert opinion, or other information sources. To use this prior knowledge for Causal Discovery from data, they introduce a Reinforcement Algorithm that penalizes the Causal Discovery Algorithm for deviating from priors. Thus they offer a systematic way to impose priors on the DAG.

Large Language Models for Causal Discovery

Large Language Models (LLMs) have recently begun seeing widespread use for causal discovery. In [51], the authors deal with the question of whether a passive learner can learn causal intervention strategies that it can use in the test data to uncover its underlying causal structure. They designed a two-stage agent that in the first stage (*experimentation strategy*), intervenes in the data generation process and learns the causal structure, and in the second stage (*exploitation strategy*), it uses its causal knowledge to intervene in the remaining of the episode to achieve a goal g . To learn the experimentation part, they use a large language model (a 70 billion parameter Chinchilla) and train it by the next word prediction task. They deduce that despite confounding variables in the training data set and passive learning, the passive learner agent can come up with causal discovery tasks in the test data, though an active RL learner with human feedback on on-policy data may improve the results.

In [52], the authors mention three types of causal inference that one may expect from a large language model. Type I is to identify the causal relationship between two subjects using domain knowledge. Type II is causal inference from a data set and for example, deciding which variable is the cause of the other. Type III is the quantitative estimation of the effect of a cause variable

on a dependent variable. This paper shows several practical improvements for human to LLM communication strategies to determine Type I causal relationships, as well as combining LLMs for causal discovery with existing methodologies to create an end-to-end system that can combine all three forms of causal inference.

In [53], authors construct a verbalized task of inferring causation from correlations, CORR2CAUS, using conventional causal discovery methods like Peter-Clark (PC), which consists of 400K data points. By giving this task to 17 main LLMs, they deduce that the outputs are almost random for all language models and none of them are able to infer causality from correlation structure. By fine-tuning these models with this task, the results improve slightly but still not significantly. This paper shows that the usage of current state-of-the-art LLMs (GPT-4) combined with robust strategies for interfacing with LLMs in the most optimal ways can generate causal inferences that are of practical use.

In [54], the authors take a positive attitude toward causal discovery and causal reasoning by LLMs. They divide causal discovery from two points of view. The first angle is *Covariance vs. Logic-based Causality*. Covariance causality is the causal discovery using statistical methods and numerical evidence to discover the existence and strength of a causal relationship between two variables. While in a logic-based approach, uses logical reasoning and domain expertise to discover a causal relationship. The other view is *Type vs. Actual Causality*. In type causality, it tries to perform a causality inference such as causal discovery or causal effect estimation between two variables. While Actual Causality probes for causal reasons for a special event. The authors also consider various causal tasks including *Causal Discovery*, *Effect Inference* and *Attribution*. Using GPT-3.5 and GPT-4, the authors claim that they could achieve a higher accuracy in various causality tasks. They've achieved 97 percent accuracy for pairwise causal discovery, 92 percent accuracy in logic-based (or counterfactual) reasoning tasks, and 86 percent accuracy in actual causality tasks which improve the current causal discovery algorithms by more than 10 percent. Despite these improvements, LLMs make some trivial mistakes that open new doors for further research and developments in the field of Causal reasoning by LLMs

In [55], the authors use LLMs to draw causal edges between 18 medical features related to lung cancer. The authors mention the lack of use of domain experts in usual score-based or constraint-based causal discovery methods that can be filled by using LLMs for achieving DAGs. They compare the DAGs generated by LLMs from both edge-by-edge interfacing or interfacing for the whole graph at once, and they achieve higher accuracy with respect to usual methods of causal discovery when compared based on Bdeu score. They also mention that the accuracy might even increase if the LLM was trained over a specialized database. The framework presented within this paper allows for DAG-generation that is more robust and scalable to hundreds of input features, as detailed in the Approach section.

In [56], the authors mention that usual causal discovery methods have the limitation that they only find the DAG up to *Markov Equivalence Class (MEC)*. And then an expert can choose the right DAG from the MEC. For cases when experts can make mistakes, they formalize the question of finding the true DAG among MEC by an *Imperfect Expert* as minimizing the size of the MEC chosen by the usual causal discovery method while the probability of inclusion of the true DAG is controlled and superior to a constant. Then they use an LLM as an imperfect expert that mitigates the performance. They suggest using a Bayesian causal discovery approach and replacing MEC-based prior with a learned posterior distribution over graphs. This paper

uses similar ideas by splitting the DAG generation task into sub-graphs, as detailed in the LLMs for Feature Clustering section.

While the articles just mentioned use LLMs as domain experts to deduce causal relationships between variables, in [57] the authors combine the classical score-based Causal Structural Learning (CSL) methods with LLMs for obtaining the DAG for a data set. They query various LLMs including GPT-4 for causal relationships between pairs of variables giving a short description of the variables. Then they use the LLMs response as a prior for score-based CSL algorithms in two hard and soft settings. In the hard setting, they enforce the causal ancestry relations proposed by the LLM to be completely satisfied by the set of Bayesian Networks they search in. In the soft setting, they consider a penalty for deviating from the prior conditions but let some of the LLMs suggestions be violated. The authors report that they have achieved a higher accuracy for the Causal Discovery task and drawing the true DAG, when using the LLMs as a prior for CSL algorithms compared to the case that they run the CSL algorithms directly on the data without LLMs suggestions as a prior.

In [58], the authors propose using LLMs for *Context-Aware Automated Feature Engineering (CAAFE)*. They mention that the most time-consuming tasks of a data scientist are data cleaning and feature engineering and most data engineering algorithms come up with dis-intuitive features. So they propose CAAFE as a pipeline for automatic feature engineering empowered by vast knowledge of LLMs. They describe the task and the data set by a call to a LLM. The LLM generates a Python code that transforms the initial raw data into processed features that an AutoML system can use as input data. It also describes the features which are extracted in terms of natural language that increases the interpretability of the AutoML pipeline. Although they increase the ROC AUC by this method, LLMs' hallucinations are still pitfalls of this automated feature engineering method. This paper utilizes some similar ideas for constructing different features to represent graph edges in the causal DAGs, as detailed in the LLMs for Feature Engineering section.

Overall, there has been a significant uptick in research for LLMs for causal discovery and modeling, but this chapter provides perhaps the most complete overview to date of an end-to-end causal discovery and modeling framework applicable in the finance and investment management domains.

Large Language Models for Quantitative Finance

The field of finance has been influenced by the introduction of *the Natural Language Processing (NLP)* method by sentiment analysis and nowcasting. With the introduction of Large Language Models, finance has been considered as one of the first fields to benefit from, and the rising number of articles related to the applications of LLMs in quantitative finance in recent years just shows this. However, using LLMs that are trained on general databases and even FinLLMs that are trained on financial data in financial tasks may lead to challenges reviewed in this section including availability, output volatility, and hallucinations.

In [59], the authors propose the problems of using LLMs trained on General text data for applications in finance. The difficulty might be in the different meanings or sentiments of words in general and financial contexts or dilution of financial data in a general database. There have

been some efforts like BloombergGPT to train LLM models on financial data that have overcome general LLMs in financial tasks. But lack of access to their database and API's to the public makes it difficult to use them for research and development purposes and making contributions. In addition, training their model is expensive and for real applications in finance we'd need to fine-tune the model on new and live data. So the authors propose *Financial Generative Pre-trained Transformer* (Fin-GPT) as a solution for democratizing financial LLMs. They provide financial data from 34 different data sources regularly updated, through API. They also introduce Reinforcement Learning with Stock Prices as a method for fine-tuning LLMs with market data. They show empirically the effectiveness of their method in various financial tasks.

In [60], the author studies the effect of uncertainty in the LLMs and its impact on decision-making in finance. The uncertainty in the output of a LLM might be due to the temperature setting or the ambiguity of or change in the query. For example in the task of sentiment classification, the sentence might not have a clear positive or negative sentiment. In this article, the author proposes a way to quantify the uncertainty and volatility of a LLM in an investment strategy without direct access to intermediate layers of the LLM in an LLM-based trading strategy. This quantification can be used to enhance the uncertainty effects in the financial decision-making or trading strategy.

In [61], the authors study the hallucination as a deficiency in financial LLMs (FinLLMs). Hallucination is a main issue in financial tasks due to the intricate nature of financial concepts. They empirically measure the performance of FinLLMs in learning and memorizing financial concepts. They also measure the performance of LLMs for a basic financial task, say querying price data. They also investigate four methods to enhance LLMs' hallucination in Finance, say *Few Shot Prompting*, *Decoding by Contrasting Layers* (DoLa), *the Retrieval Augmentation Generation method* (RAG) and *the prompt-based tool learning method*. However, they remind the necessity for more research on LLMs' hallucination issues in finance.

3.3 METHODOLOGY

The initial phase of the causal discovery framework involves the construction of a Directed Acyclic Graph (DAG), $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, to represent the presumed causal relationships among the variables. In this context, each node in \mathcal{G} symbolizes a distinct variable, such as stock returns, economic indicators, or company fundamentals. The directed edges in \mathcal{E} indicate the hypothesized causal directions. For instance, if variable X (e.g., a company's earnings) is conjectured to causally influence variable Y (e.g., stock returns), this relationship is depicted with a directed edge from X to Y in the DAG.

Upon establishing the DAG, the subsequent step entails utilizing observational data to deduce the strengths or weights of the defined causal relationships. This process typically employs statistical methodologies like regression analysis. For a simple scenario where Y is causally impacted by X , the regression model can be formalized as:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

where β_0 represents the intercept, β_1 is the coefficient indicating the causal effect of X on Y , and ϵ denotes the error term. The primary objective is to accurately estimate β_1 , thereby quantifying

the causal impact of X on Y . In essence, the DAG functions as a pre-modeling feature selection mechanism, guiding the formulation of the regression model.

The concept of interventions, operationalized through the *do operation*, is pivotal in simulating the influence of external manipulations on the variables. For example, to explore the effect on stock returns (Y) following a hypothetical increase in a company’s earnings (X), the *do operation*, denoted as $\text{do}(X = x)$, is applied. This operation effectively severs any incoming causal links to X in the DAG, rendering X independent of its usual precursors. The distribution of Y under this intervention is denoted as $P(Y \mid \text{do}(X = x))$, representing the distribution of Y when X is externally set to a particular value x .

In the context of a simple causal relationship where X directly influences Y without confounders, $P(Y \mid \text{do}(X = x))$ can be directly inferred from the regression model by substituting x for X . In more intricate scenarios involving confounders or indirect causal paths, additional adjustments and analytical steps are necessitated to accurately compute $P(Y \mid \text{do}(X = x))$ using do-calculus rules, assuming a correctly specified causal model.

LLMs for Causal Discovery

The authors propose the following general framework for automated causal DAG construction based on any arbitrary set of features for a supervised learning task with the single constraint being each feature must have a valid name and definition.

Though recent advances in LLMs have greatly increased the length of the input context window, the longest context-length models are in practice not always the most performant for highly complex tasks. Due to the complexity of causal discovery and the non-deterministic behavior of LLMs, the authors have observed the following current limiting factors in causal DAG generation:

1. Current state of the art LLMs have limited bandwidth relating to the quantity of features that can be evaluated simultaneously. Indeed, the authors have observed a drop in causal inference quality when the input context contained upwards of 30 features for experiments with GPT-4.
2. The set of causal relationships generated for each LLM inference is not unique.

In order to tackle these limitations, the authors split the causal discovery method into two distinct phases: Feature Clustering and Causal Discovery. The overall process is described below:

LLMs for Feature Clustering

The initial step in the proposed causal discovery process is splitting the input features F into distinct groups of at most n_{feat} features and generating a causal DAG for each group. This is a solution to the dropoff in causal inference quality seen with inputting large numbers of features into a single forward pass for an LLM call. For the experiments in this chapter, the authors set $n_{feat} = 30$ as that was the threshold after which the quality and consistency of causal relationships generated was observed to drop off significantly.

The authors propose as an alternative to classical correlation clustering algorithms by using LLMs to cluster the input features into mutually exclusive groups. This process proceeds in three steps, implemented as three distinct styles of interfacing. This approach helps greatly scale the number of input features F that a DAG can be constructed for. This approach can be generalized to work in a loop (iterating the same task until the sub-groups are small enough to give a second, different task) to scale up the number of input features even further.

Task 1 - pre-clustering. The input features F are grouped into an initial grouping denoted as G_1 . Each group g in G_1 contains a subset of features in F , such that $g \subset F$ and $G_1 = \{g_1, g_2, \dots, g_n\}$, where n is the number of initial groups.

Task 2 - creates sub-groups: For each group g in G_1 , a set of subgroups is created. Denote the set of subgroups created from group g as S_g , where $S_g = \{s_{g1}, s_{g2}, \dots, s_{gm}\}$ and m is the number of subgroups for group g . Each subgroup s is a subset of the parent group g , such that $s \subset g$.

Task 3 - final clustering: Each subgroup s in each S_g is then used as the element in another clustering process. The result of this clustering process is the final grouping, denoted as G_f . Each group in G_f is formed by clustering the subgroups S_g from the initial groups G_1 .

The final clustering is exhaustive and mutually exclusive. Now each group in G_f has an LLM-generated name and description. Thus, if necessary one can combine groups together very easily to obtain a desired number of clusters. One can always also recluster groups in G_f if he wants more granular clusters. This technique thus provides a lot of liberty to the user.

The authors first construct an instruction that utilizes the names and definitions of each feature in F to separate them into n distinct groups or clusters $\{g_1, \dots, g_n\}$.

Using the names and definitions of each feature, they construct an instruction to return another clustering of the initial groups $G_1 = \{g_1, \dots, g_n\}$ into distinct sub-groups $S_g = \{s_{g1}, s_{g2}, \dots, s_{gm}\} \forall g \in G_1$ of at most n_{feat} features each based on the definitions provided for each feature.

The authors compare the results obtained using a classical clustering algorithm versus the proposed method in the Applications section. Depending on the number of features and the average length of the feature names and definitions, it may be needed initially partition the set of features into random distinct groups in order to fit the context length of the LLM. If the initial separation is needed, the intermediate groups S_g regrouped them all using the final task (Task 3). The authors also include a sample Task 4 which could be used to combine clusters using their names and definitions if G_f is considered too granular.

3.4 DAG GENERATION - LLM FOR CAUSAL DISCOVERY

Having generated the final set of groupings G_f , the authors now focus on the generation of the causal DAG modeling the intra-causal and inter-causal relationships between these groups.

In order to generate a valid final DAG, using the names and descriptions of the set of features, all possible causal relationships in each group of features are iteratively generated and modeled as DAGs. Subsequent instructions are then used to validate and refine each of the sub-graph DAGs obtained in this step, similar to the Chain-of-Thought technique seen in [62].

To model the inter-group causal relationships, the authors start by generating descriptions for each group based on the definitions of the features belonging to each cluster in G_f . Using these

descriptions, they generate the inter-group causal DAGs. Finally, the causal relationships present in this inter-group DAG are used to link together each of the features belonging to each group. A visualization explaining this last step can be seen in Exhibit 3.1.

In this step the inter-group causal relationships are fully connected, i.e. if it is determined that $g_j \rightarrow g_k$ for some $g_j, g_k \in G_f$, it is not initially known which nodes (features) in g_j drive which nodes (features) in g_k . Thus, in order to not miss out on valid relationships, the authors elect to over-connect groups and filter out the irrelevant relationships in a step described in the Causal Validation section. Of course, an alternative approach here is to once again use an LLM to validate these emergent nodes, but the authors elect to go with an empirical causal validation approach for performance reasons.

In all the instructions, the authors include templates of the expected output format and describe the structure of the list of features in the input as these are best practices used to improve the quality of responses of GPT-4 [63]. The authors will omit mentioning this step for the rest of the chapter.

Now, here is a detailed description of the necessary steps to generate a valid causal DAG for each group of features. These steps are also used to generate the final causal DAG used to link each group together; the sample instructions for this process can be seen in Appendix B:

1. Generate the set of all possible valid causal relationships for a group of features using GPT-4 and model this as an undirected graph.
2. Generate a causal DAG from the set of causal relationships obtained in the previous step by determining the directions of all the causal relationships using GPT-4.
3. Verify the validity of all the causal relationships and correct any mistakes made during the initial generation of the DAG using GPT-4.
4. Verify the corrections made were valid and verify if there are any remaining mistakes in the DAG using GPT-4.
5. If there were any final mistakes, correct these mistakes and return the final version of the causal DAG.

The authors describe this process in three distinct phases: Causal Exploration, Causal Inference and Causal Validation.

Causal Exploration

During causal exploration, the causal DAGs are repeatedly generated over the same group of features in order to generate the set of all possible causal relationships for the given group. The DAGs were generated 10 times per group for the experiments described in this chapter. This step is necessary to reduce the impact of the non-deterministic behavior of LLMs.

For each iteration of causal DAG generation, the authors provide the following definition of a causal relationship: *A causal relationship between two features means that a change in one feature causes a change in the other feature.* The authors also use general chain-of-thought instructions to instruct GPT-4 [62] to include the feature’s definitions and the definition of causal relationship in its reasoning. The authors use in-context Learning and justifications by example

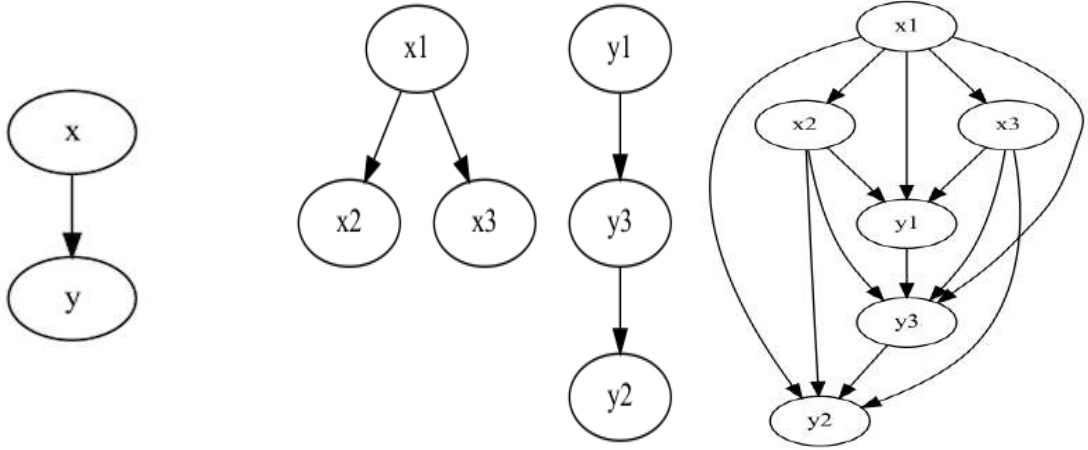


Figure 3.1: A visualization of the last step combining all the groups into one unified graph. In this example, there are categories of features x and y . Each group is composed of 3 features. The DAG on the left models the inter-group causal relationships, the DAG in the middle models the intra-group causal relationships and the DAG on the right is the final graph combining both results.

and thought processes as part of the process of communicating with LLMs in the best way to generate high-quality outputs.

Sample instructions for these steps can be seen in Appendix Section C.1.

Causal Inference

During causal inference, the authors include the set of edges in the instruction used to form the initial undirected graph generated during the causal exploration phase. These are outputted as a string that can be parsed into a list of tuples. Finally, the authors use the same techniques as in the previous phase and also include the definitions of a DAG and a causal relationship in the instructions.

Now for the set of edges, the authors instruct the LLM to generate a causal DAG. GPT-4 performs perhaps surprisingly well for this task and the instruction above, which in combination return a DAG based on definitions of the input features F forming the nodes in the graph. The LLM is also tasked with returning a justification for each decision it makes when constructing the graph. Thus, the LLM returns a justification for the direction of each edge; the authors observe that including the justification also generates higher quality outputs, similar to the approach seen in [62].

Sample instructions for these steps can be seen in Appendix Section C.2.

Causal Validation

During the causal validation phase, the goal is to correct any mistakes the LLM makes have made during its initial DAG generation. The authors therefore use all the same techniques mentioned in the previous two phases in order to improve performance.

The authors also include the outputs of previous instructions in order to provide context and generated justification connected to the previous outputs. This history starts at the causal inference phase. Thus, before performing validation, the instruction includes the initial undirected graph representation, the features names and definitions, the initial DAG generation and the justifications for each edges direction in the DAG. An example of this final validation instruction can be seen in the Appendix Section C.3.

Now there are three main types of mistakes an LLM can make during graph generation:

1. False negative / missing causal relationship
2. False positive causal relationship (i.e. produce a non-existent / illogical relationship)
3. Produce a correct causal relationship with an incorrect causal direction

The authors include this in the verification instruction and direct the LLM to analyze the whole causal DAG generation process and attempt to remedy any mistakes made in this process. If so, the LLM returns a list of the invalid edges and a list of the corrections, both accompanied with justifications for their inclusions. The DAG is then corrected using the parsed responses before starting the final verification step.

During this final verification step, the authors also add to the concatenation of previous correction instructions (i.e. the “chat history”) if any corrections needed to be made. The LLM is ordered to “verify its thought process” iteratively (in a loop) and apply corrections where necessary until the stopping criteria pre-specified within the instruction itself is hit.

Given the validity of the causal DAG, the authors use the do-calculus from observation data to further validate the causal relationship non-parametrically. Do-Calculus provides a statistical significance value for each edge. Edges that are not statistically significant are removed from the DAG. In addition, a cross-validation is performed on subset of the observation data to confirm the statistical significance of the edge. This final step is called the refutation step. The following definition captures the requirement that a causal query Q within the factors be estimable from the factor exposure:

Definition 3.1. A causal query $Q(M)$ is identifiable from a graph \mathcal{G} , given a set of assumptions A and the set of observed factors V , if for any pair of models M_1 and M_2 that satisfy A , we have

$$P_{M_1}(v) = P_{M_2}(v) \implies Q(M_1) = Q(M_2)$$

When a query Q is given in the form of a *do*-expression, for instance, $Q = P(y|do(x), z)$, its identifiability can be decided systematically using an axiomatic system known as the *do*-calculus [64]. The axiomatic system replaces probability formulas containing the *do*-operator with ordinary conditional probabilities in three axiom schemes.

Let X, Y, Z , and W be arbitrary disjoint sets of nodes in a casual DAG \mathcal{G} . Denoting $\mathcal{G}_{\overline{X}}$ the graph obtained from \mathcal{G} by deleting all arrows pointing to nodes in X , $\mathcal{G}_{\underline{X}}$ the graph obtained from \mathcal{G} by removing all arrows emerging from nodes in X . The following three rules are valid for every interventional distribution compatible with \mathcal{G} .

Rule 1 (Insertion/deletion of observations):

$$P(y|do(x), z, w) = P(y|do(x), w) \text{ if } (Y \perp\!\!\!\perp Z|X, W)_{\mathcal{G}_{\overline{X}}}$$

Rule 2 (Action/observation exchange):

$$P(y|do(x), do(z), w) = P(y|do(x), z, w) \text{ if } (Y \perp\!\!\!\perp Z|X, W)_{\mathcal{G}_{\overline{XZ}}}$$

Rule 3 (Insertion/deletion of actions):

$$P(y|do(x), do(z), w) = P(y|do(x), w) \text{ if } (Y \perp\!\!\!\perp Z|X, W)_{\overline{XZ(W)}}$$

where $Z(W)$ is the set of Z -nodes that are not ancestors of any W -node in $\mathcal{G}_{\overline{X}}$.

Note that in cases where the set X is empty, Rule 1 corresponds to the application of d-separation principles to interventional distributions, and Rule 2 becomes analogous to the backdoor adjustment criterion.

To establish identifiability of a query Q of factors, the rules of *do*-calculus are repeatedly applied to Q , until the final expression no longer contains a *do*-operator. The final *do*-free expression can serve as an estimator of Q . The *do* calculus was proven to be complete to the identifiability of causal effect [65]; [66]. As such, the causality relationship generated by LLM is endorsed by observation data.

The authors also utilize L_1 regularization (Lasso Regression) when fitting the final models as a final causal validation / feature selection step, reducing the number of features used in the final model and reducing the multi-collinearity of the final feature set. Since the Lasso model is fit on each training fold in a backtest, the final feature selection / causal validation step is dynamic for each period.

LLMs for Feature Engineering

In order to represent the relationships between the different features (edges) in the DAG as actual features, LLMs can also be used to select the most logical transformation of the two features composing each edge to create a new feature.

A simple approach for this is to provide a set of “primitives” or standard transformations and query an LLM to select one the definitions of the features and the direction of the causal relationship between the two features.

This step could possibly be improved by using deep learning / LLM-based feature engineering frameworks or the newly proposed CAAFE framework cited in the literature review, as seen in [58].

3.5 APPLICATIONS

The authors focus on two key applications of the framework outlined above: causal factor investing, and the application of the DAG generation framework to other feature sets (using credit spread modeling as an example). In the following section, all the Do-Calculus computations on the DAG are done through the use of the following python package: DoWhy [67]. The authors omit mentioning its use through the rest of the results section. The details of each are outlined below.

Applications to Causal Factor Analysis

Factor investing is a financial theory aiming to explain the returns of stock as connected to risk *factors*, that plays a key role in risk management (where factor loads in a portfolio are carefully managed) and in investing strategies that aim to strategically load up on specific factors (e.g. “smart beta”). A researcher usually determines this set of factors by making causal assumptions and calculating factor exposures concerning the stock’s returns using procedures inspired by [33] and [68] as mentioned in [45].

Now even if these causal assumptions drive the subsequent modeling decisions the researcher will make in his analysis, they are rarely explicitly stated. Clearly stating these causal assumptions would require the researchers to provide empirical evidence to back their claims. Nevertheless, this opens up the opportunity to use the rules of Do-Calculus to de-bias their data and improve explainability of the modeling structure, especially when using a large set of factors. Consequently, the proposed framework bridges this gap by automating this potentially incredibly laborious process.

The authors investigate the set of 153 factors presented in [39]. Using the base definitions and names presented for each factor, the authors manually improve the definitions in order to make them more verbose and representative of their construction (another step that could potentially be automated/improved using LLMs). The time period analyzed is from 1971-11-30 to 2022-12-30. Each factor has an associated ETF modeling its daily returns over this period.

Now the authors compare the causal DAGs generated using their proposed LLMs clustering method to a classical correlation clustering algorithm (described in the LLMs for Feature Clustering section). They generate a first clustering using a LLM (GPT-4) and compare it to a baseline correlation clustering approach, as seen in e.g. [39] using hierarchical agglomerative clustering. The authors in [39] define the distance between factors in the second clustering as one minus their pairwise correlation and use Ward as the linkage criterion. The authors also define the correlation based on CAPM-residual returns of U.S. factors signed as in the original paper [69]. Exhibit 3.2 compares inter-cluster correlations of the two methods; although the method proposed in this thesis naturally has lower inter-cluster correlations, the causal DAGs for each cluster are substantially more cohesive from an expert judgement perspective, as can be seen in this [GitHub repository](#). Despite not using correlation or time-series data as an input, the LLM clusters produce average inter-cluster correlations of 0.292, compared to 0.543 for the correlation clustering approach, and intra-cluster correlations of 0.0527 compared to 0.0047. The clusters produced by the LLMs are also in some ways more orthogonal, having average absolute inter-cluster correlations of 0.086, compared to 0.1451 for the correlation clustering approach. The authors then generate a complete DAG for each set of clusters (“causal” LLM clusters or correlation clusters) as described in the Methodology section.

Now in order to confirm the causal DAGs using Do-Calculus, the ETF returns are used to construct proxies for the factors of each individual stock in the S&P500 by computing the rolling beta exposures of each stock to the 153 factor ETFs. This generates a feature set X_t of 500 stocks and 153 features per stock per period, with each feature being a proxy for a factor exposure computed using an individual ETF. The authors then use these proxies for factor exposure to verify all the causal relationships in each causal DAGs generated by the LLM edge-wise. In other words, they use Do-Calculus to confirm the validity of edge in the causal DAGs. However, due

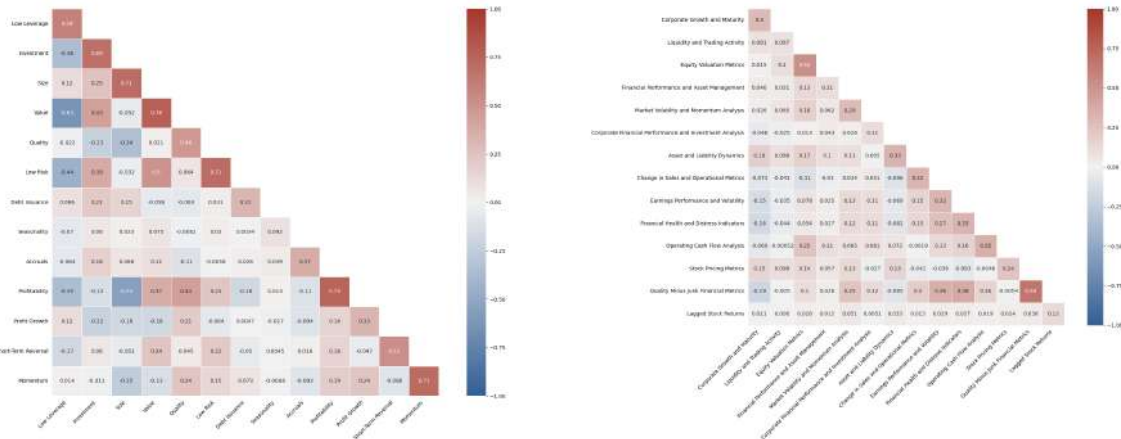


Figure 3.2: This figure shows the average pairwise Pearson correlation between factors from different clusters and factors from same clusters for hierarchical agglomerative clustering (left) and LLM clustering (right) using data on US stocks from 1971 to 2021. The figure on the left is inspired from [39].

to computational constraints, the authors did not run the Do-Calculus on the fully connected graph as in 3.1. Indeed, they ran Do-Calculus separately on each DAG modeling the intra-group causal relationships.

In the case of the causal DAGs modeling the inter-group causal relationships, the authors executed the Do-Calculus using the first n principal components sufficient enough to explain at least 85% of the groups variance. If any of the components is proven invalid by the Do-Calculus computation, all associated edges are removed. Now, having validated all the causal DAGs, the authors add a new node each of them representing the stock returns. An example visualization can be seen in the Exhibit 3.3. For each causal DAG representing the intra-group causal relationships, the authors then use Do-Calculus to analyze the causal relationships of each factor with respect to the S&P500 constituents stock returns. The authors remove from the causal DAGs any causal relationship/edge to the stock returns who are not statistically significant with a p-value of 0.05. These statistically insignificant relationships represent factors who do not directly impact S&P500 constituents stock returns. Nevertheless, their values may still impact the stock returns through their causal relationships with other features in the DAGs.

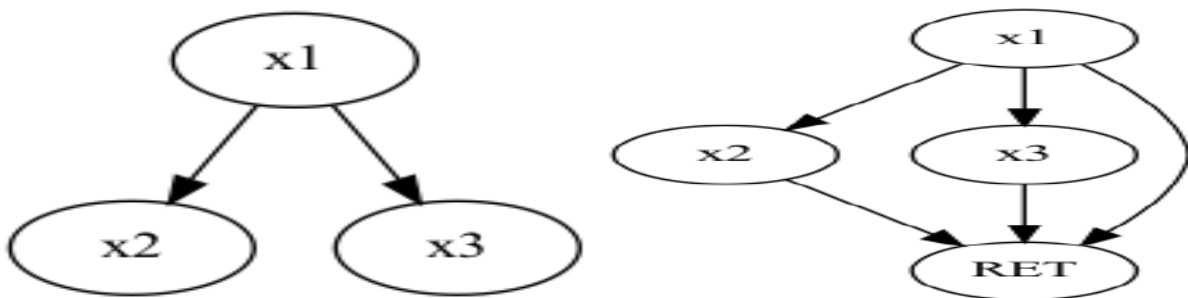


Figure 3.3: This figure shows how the authors connect each feature to the Y node of interest. In the factor investing case, this Y node is represented by the stock returns RET.

Table 3.1: A comparison of the DAGs generated using hierarchical and LLM-based clustering.

CLUSTERING METHODOLOGY	AVERAGE NUMBER OF NODES	AVERAGE NUMBER OF EDGES	% REFUTED NODES	REFUTED EDGES %
CORRELATION CLUSTERS	10.93	13.71	3.64%	31.29%
LLM CLUSTERS	11.77	11.54	3.15%	32.43%

Table 3.2: Summary statistics of in- and out-of-sample RMSE for each of the correlation clusters inspired by [39]

	AVG RMSE (OUT-OF-SAMPLE)	AVG R^2 (IN SAMPLE)	AVG ADJUSTED R^2 (IN SAMPLE)	AVG NUMBER OF NODES USED	TOTAL NUMBER OF NODES ON GRAPH	AVG NUMBER OF EDGES USED	TOTAL NUMBER OF EDGES
LOW LEVERAGE	0.0997	0.0496	0.0465	10.6818	11	13.9766	14
DEBT ISSUANCE	0.0971	0.0290	0.0269	6.6703	7	10.9318	11
PROFITABILITY	0.1001	0.0468	0.0436	10.6993	11	14.9923	15
SHORT-TERM REVERSAL	0.0972	0.0260	0.0247	5.6987	6	4.9538	5
LOW RISK	0.1006	0.0683	0.0641	17.6613	18	15.9772	16
INVESTMENT	0.1027	0.0645	0.0601	21.6206	22	17	17
SIZE	0.0958	0.0251	0.0241	4.7000	5	3.9856	4
ACCRUALS	0.0967	0.0251	0.0235	5.7115	6	6.9643	7
VALUE	0.1012	0.0579	0.0540	17.6442	18	14.9987	15
QUALITY	0.0987	0.0571	0.0540	16.6799	17	10	10
MOMENTUM	0.1044	0.0509	0.0474	7.6683	8	18.9626	19
PROFIT GROWTH	0.0978	0.0419	0.0394	11.6651	12	9.9462	10
SEASONALITY	0.0973	0.0467	0.0446	11.6827	12	6.9011	7
AVERAGE	0.0992	0.0453	0.0425	11.4449	11.7692	11.5069	11.5385

For an average period, there are in total of 103.4 factors left out of the initial 153 factors, combined with 184.6 features representing the edges in the DAG (from a total of 192 across all of the LLM-cluster DAGs).

In addition, a more detailed analysis comparing the average performance per each of the hierarchical clusters as seen in [39] is provided in table Exhibit 3.2, and of the LLM-clusters in Exhibit 3.3. One can see that the LLM clusters provide a comparable level of predictive power for monthly returns, while providing a much less correlated and more interpretable feature set. The LLM clusters also utilize more edges per cluster, leading to a potentially richer representation of causal relationships. In addition, the % of the refuted nodes are much lower than in the combined dataset, given there is less redundancy within the individual clusters.

Applications to Causal modeling of Credit Spread Drivers

The drivers of changes in credit spread is a long-lasting argument. Early studies by Black and Scholes [70] and Merton [71] introduced structural model to explain corporate default risk and inspired the search for company-level and macro variables, later Collin-Dufresn et al [72] stated the impact of market-related factors, and Chen et al [73] identified the high correlation between the equity market performance with the spread. But is there any causality relationship between these discovered factors? If so, the redundant features can be removed using a causality approach.

To identify to causal relationship within the factors that might drive the credit spread, the authors utilize the dataset [74], and generate the DAG following the steps mentioned in the methodology. This numerical dataset consists of different macro-economic factors and market

Table 3.3: Summary statistics of in- and out-of-sample RMSE for each of the LLM clusters

	AVG RMSE (OUT-OF-SAMPLE)	AVG R^2 (IN SAMPLE)	AVG ADJUSTED R^2 (IN SAMPLE)	AVG NUMBER OF NODES USED	TOTAL NUMBER OF NODES ON GRAPH	AVG NUMBER OF EDGES USED	TOTAL NUMBER OF EDGES
CORPORATE GROWTH AND MATURITY	0.0950	0.0122	0.0119	1.6827	2	0.9615	1
QUALITY MINUS JUNK FINANCIAL METRICS	0.0957	0.0205	0.0196	3.7452	4	3	3
STOCK PRICING METRICS	0.0949	0.0121	0.0118	1.6731	2	1	1
ASSET AND LIABILITY DYNAMICS	0.0999	0.0587	0.0546	18.6346	19	17.9647	18
FINANCIAL HEALTH AND DISTRESS INDICATORS	0.0977	0.0347	0.0329	4.7423	5	8.9744	9
LAGGED STOCK RETURNS	0.0993	0.0521	0.0490	9.7135	10	15.8894	16
FINANCIAL PERFORMANCE AND ASSET MANAGEMENT	0.1005	0.0599	0.0558	17.6528	18	16.9717	17
CHANGE IN SALES AND OPERATIONAL METRICS	0.0957	0.0173	0.0161	3.6779	4	5.9135	6
CORPORATE FINANCIAL PERFORMANCE AND INVESTMENT ANALYSIS	0.1010	0.0597	0.0549	15.6611	16	22.9791	23
MARKET VOLATILITY AND MOMENTUM ANALYSIS	0.1181	0.0992	0.0900	26.6546	27	48.9714	49
OPERATING CASH FLOW ANALYSIS	0.0965	0.0279	0.0265	5.6955	6	5.9744	6
LIQUIDITY AND TRADING ACTIVITY	0.0999	0.0534	0.0507	11.6731	12	10.9773	11
EARNINGS PERFORMANCE AND VOLATILITY	0.1002	0.0509	0.0473	11.6907	12	16.9604	17
EQUITY VALUATION METRICS	0.1005	0.0547	0.0509	15.7212	16	14.9526	15
AVERAGE	0.0996	0.0438	0.0409	10.6156	10.9286	13.6779	13.7143

related factors. The data consists of monthly and quarterly data. Each edge in the DAG is validated using the rules of Do-Calculus. The Exhibit 3.4 presents the statistical significance of the rejection of causal relationships within the edges of the DAG.

In Exhibit 3.4, each edge is associated with its effect (i.e. strength of the relationship), its p-value and its refutation p-value. The p-value represents the statistical significance of the edge and the refutation p-value represents the robustness of the p-value associated with the effect. Indeed, the refutation p-value is computed using K-Fold Stratification of the data to test the degree of randomness associated with the edge. The authors reject / remove any edge who’s p-value is greater than 0.05 as shown in 3.5. Any edge where p-value is significant (i.e. smaller than 0.05), but whose refutation p-value is significant is rejected as well. The authors will refer to the steps described as validating the causal DAG *edge-wise*.

The authors reject 39 edges using Do-Calculus out of the 89 edges that GPT-4 returned. This is within the expectation as the LLM DAG generation approach focused mainly on recall, and is expected to generate denser initial graphs. In addition, the dataset is much smaller than the dataset used in the previous section, leading to more frequent rejections.

Using the framework, the authors are able to identify the causality relationship within the drivers of credit spread; to a practitioner, this may be a good starting point for a feature set or a causal model, especially in situations where expert judgement in a given domain is not readily available in-house.

3.6 LIMITATIONS AND FUTURE WORK

The authors leverage GPT-4 for all the analyses presented in this chapter. In its current state, GPT-4 is a powerful language model that is able to produce causal relationships intuitive to human experts and those that align well with empirical observations.

Nevertheless, it has some key limitations researchers and practitioners should be aware of. One is in making unpredictable mistakes which can be very costly in the causal DAG generation and similar use cases. Indeed, the authors of [54] repeatedly mention this in their experiments on pairwise causal discovery and full graph discovery that LLMs produce infrequent but unpredictable errors making the process of automating causal discovery unfeasible for the moment. Thus it is essential to have human supervision to correct the LLMs in the case they produces a hallucination

(i.e. incoherent output). In order to make this process less laborious, the an LLM can also be used to recursively correct its own mistakes (as can be seen in Appendix C). The authors of [54] found that GPT-4 was the only model capable of verifying the consistency of its outputs (self-consistency), though the capabilities of LLMs are evolving rapidly and it is possible other models can display similar capabilities with additional development of interfacing strategies.

The authors also hope that the DAGs generated in the chapter, which can be accessed on GitHub [here](#), will spur further research into optimal representations for risk and factor models. This chapter shows that causal approaches can now be scaled to provide a working alternative towards correlation-based approaches for generating feature representations in cases where the universe of possible features is large and the signal-to-noise ratios are too low to find the optimal feature representations empirically.

These causal DAGs similar to those demonstrated in this chapter can be leveraged to improve model robustness and out-of-sample performance, similar to the process of injecting expert judgement priors into predictive models. The authors are particularly interested in how these alternative feature representations can be used in constructing risk and factor models, and how such models may compare with state-of-the art models used by practitioners today.

It is also necessary to remember that the performance of the LLMs drastically relies on the process of creating useful instructions, and the current state-of-the-art in foundational models. The authors believe that the end-to-end approach can be improved substantially both through injecting additional expert knowledge into the instructions, and through the evolution of foundational models.

The authors believe that combining state-of-the-art LLMs and techniques that can be used to push their performance, as described in this chapter, in combination correlation-based methods as in [67] can be an fruitful area of research and a significant step towards automating end-to-end causal discovery in finance and beyond.

3.7 CREDIT SPREAD FIGURES AND TABLES

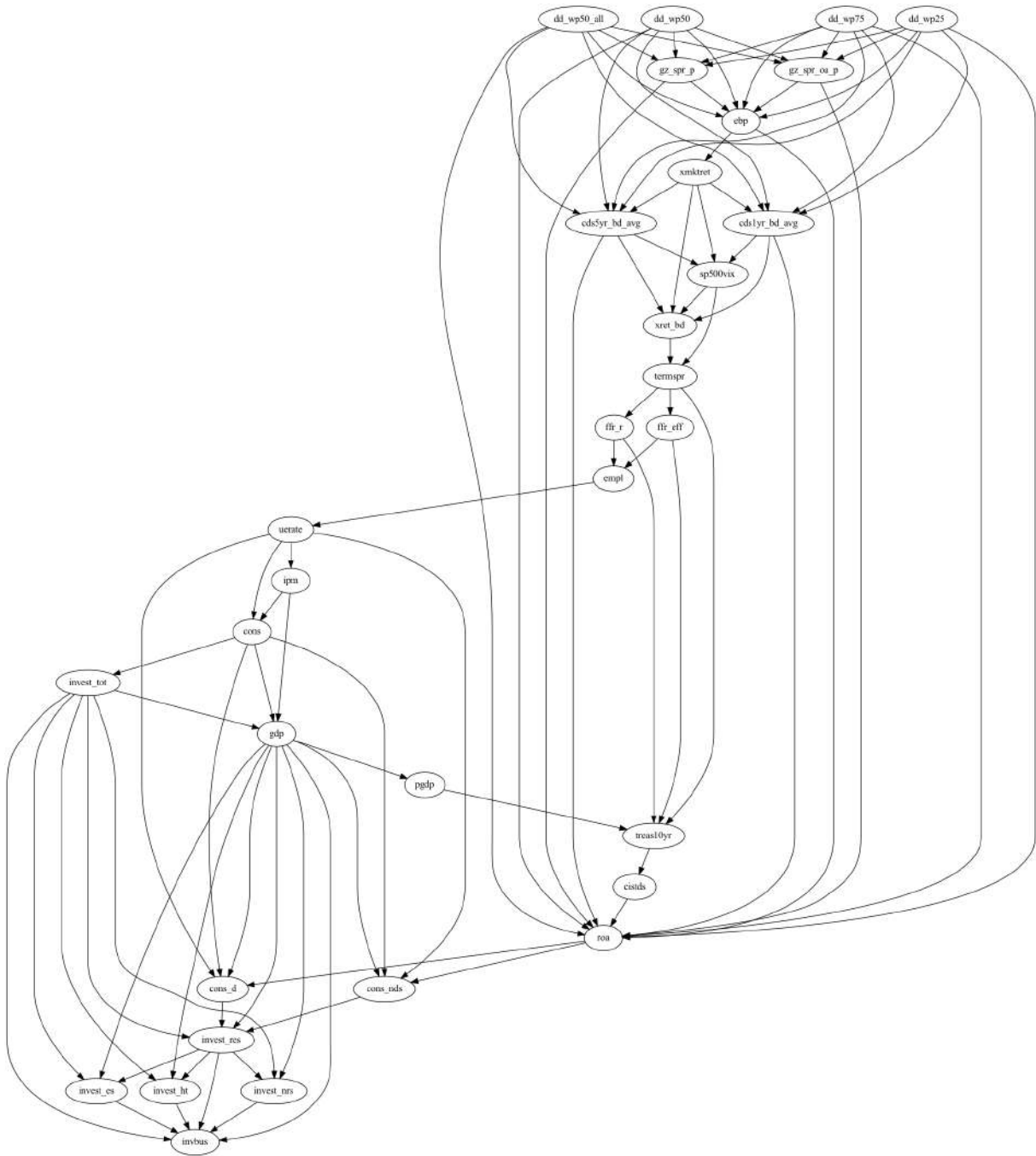


Figure 3.4: This figure shows the credit spread DAG generated by the causal DAG generation framework before Do-Calculus.

start node	end node	effect	p-value	ref. p	start node	end node	effect	p-value	ref. p
ebp	roa	-0.50	0.00	0.94	invest_res	invest_nrs	-0.15	0.00	0.84
ebp	xmktret	-2.20	0.00	1.00	dd_wp25	ebp	-0.29	0.16	0.96
xmktret	sp500vix	-0.36	0.00	1.00	dd_wp25	roa	0.19	0.55	0.70
xmktret	xret_bd	1.46	0.00	0.98	dd_wp25	gz_spr_p	-0.42	0.05	0.54
xmktret	cds1yr_bd_avg	0.02	0.05	0.84	dd_wp25	gz_spr_oa_p	-0.01	0.00	0.78
xmktret	cds5yr_bd_avg	0.01	0.19	0.94	dd_wp25	cds1yr_bd_avg	-0.18	0.64	0.72
sp500vix	xret_bd	-0.08	0.51	0.98	dd_wp25	cds5yr_bd_avg	-0.12	0.91	0.92
sp500vix	termspr	-0.09	0.00	0.96	dd_wp50	ebp	-0.26	0.13	0.98
xret_bd	termspr	-0.04	0.03	0.70	dd_wp50	roa	0.11	0.99	0.98
termspr	treas10yr	0.23	0.00	0.90	dd_wp50	gz_spr_p	-0.45	0.00	0.82
termspr	ffr_r	1.10	0.00	0.96	dd_wp50	gz_spr_oa_p	-0.06	0.62	0.94
termspr	ffr_eff	1.27	0.00	0.94	dd_wp50	cds1yr_bd_avg	-0.18	0.39	0.86
empl	uerate	0.00	0.00	0.96	dd_wp50	cds5yr_bd_avg	-0.09	0.93	0.94
uerate	ipm	-2.63	0.00	0.80	dd_wp75	ebp	-0.07	0.99	0.94
uerate	cons	40.35	0.05	1.00	dd_wp75	roa	0.00	0.51	0.96
uerate	cons_nds	233.01	0.00	0.96	dd_wp75	gz_spr_p	-0.29	0.01	0.78
uerate	cons_d	32.00	0.00	1.00	dd_wp75	gz_spr_oa_p	-0.08	0.09	0.90
ipm	gdp	135.04	0.00	0.92	dd_wp75	cds1yr_bd_avg	-0.06	0.93	0.80
ipm	cons	97.26	0.00	0.94	dd_wp75	cds5yr_bd_avg	-0.04	0.85	0.94
cons	gdp	1.25	0.00	0.86	dd_wp50_all	ebp	-0.47	0.00	0.86
cons	invest_tot	0.28	0.00	0.80	dd_wp50_all	roa	0.46	0.00	0.78
cons	cons_nds	0.87	0.00	0.92	dd_wp50_all	gz_spr_p	-0.54	0.00	0.56
cons	cons_d	0.11	0.00	0.90	dd_wp50_all	gz_spr_oa_p	-0.03	0.48	0.98
invest_tot	invest_res	1.02	0.00	0.86	dd_wp50_all	cds1yr_bd_avg	-0.41	0.00	0.90
invest_tot	gdp	0.63	0.00	0.90	dd_wp50_all	cds5yr_bd_avg	-0.35	0.00	0.98
invest_tot	invbus	0.44	0.00	0.88	gz_spr_p	roa	-0.84	0.00	0.98
invest_tot	invest_es	0.76	0.00	0.80	gz_spr_p	ebp	0.08	0.00	0.74
invest_tot	invest_ht	-0.26	0.00	0.88	gz_spr_oa_p	roa	1.69	0.00	0.88
invest_tot	invest_nrs	0.39	0.00	0.88	gz_spr_oa_p	ebp	-2.46	0.00	0.88
gdp	invbus	-0.20	0.00	1.00	cds1yr_bd_avg	sp500vix	-4.53	0.22	1.00
gdp	invest_res	0.03	0.81	0.80	cds1yr_bd_avg	xret_bd	-1.23	0.51	0.98
gdp	pgdp	0.01	0.00	0.98	cds1yr_bd_avg	roa	-1.27	0.00	0.94
gdp	cons_nds	-0.08	0.03	0.98	cds5yr_bd_avg	sp500vix	16.35	0.00	0.94
gdp	cons_d	0.09	0.02	0.80	cds5yr_bd_avg	xret_bd	-1.77	0.37	0.94
gdp	invest_es	0.04	0.24	0.94	cds5yr_bd_avg	roa	-1.43	0.00	0.68
gdp	invest_ht	0.22	0.00	0.96	ffr_r	empl	691.27	0.06	0.98
gdp	invest_nrs	-0.21	0.00	0.76	ffr_r	treas10yr	0.44	0.00	0.96
pgdp	treas10yr	-0.14	0.00	1.00	ffr_eff	empl	3683.78	0.00	0.76
treas10yr	cistds	-21.59	0.00	0.94	ffr_eff	treas10yr	0.36	0.00	0.86
cistds	roa	-0.02	0.00	0.76	cons_nds	invest_res	-0.82	0.00	0.86
roa	cons_nds	36.22	0.00	0.78	cons_d	invest_res	2.11	0.00	0.96
roa	cons_d	35.81	0.00	0.86	invest_es	invbus	-0.76	0.00	0.90
invest_res	invbus	0.02	0.27	0.98	invest_ht	invbus	-2.60	0.47	0.98
invest_res	invest_es	0.31	0.00	0.98	invest_nrs	invbus	-0.96	0.00	0.96

Table 3.4: Nonparametric causal estimation results for the credit spread DAG. Each edge in the graph is designated by a start node and an end node. The effect consists of a measure of the strength of the causal relationship, the p-value signifies its level of significance, and the refutation p-value is a measure of how robust the p-value is by testing its significance through K-Fold stratification.

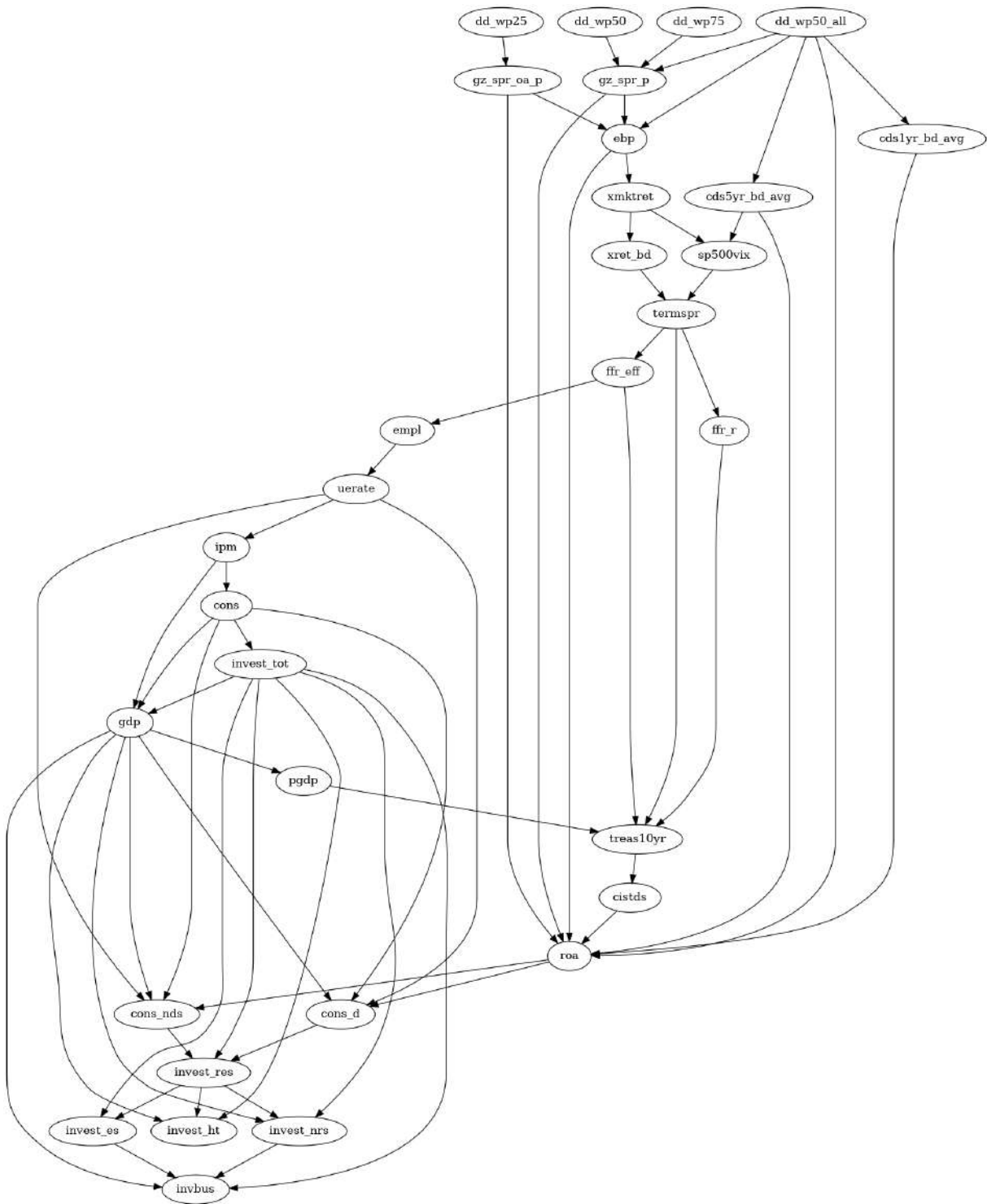


Figure 3.5: The figure shows the process of how do-calculus removed the redundant edges within the LLM generated causal DAG for credit spread. The original LLM generated DAG can be found in figure 3.4.

EXTERNALITY-AWARE COMPANY RISK RATINGS

Although investing in Environment-Social-Governance (ESG) driven portfolios is already a large and growing portion of global assets under management, applications of quantitative techniques to improve and standardize ESG scoring and the construction of ESG portfolios are underutilized. In this chapter, we propose an approach to automatically convert unstructured text data into ESG scores by using the latest advances in deep learning for Natural Language Processing (NLP). We also show how a state-of-the-art NLP technique, BERT, can be incorporated to improve the accuracy of assessing relevance and content of documents in an ESG context using social media data as an example, and discuss the relevance of this approach to automating ESG scoring and constructing ESG portfolios.

4.1 INTRODUCTION

Environmental, social and governance factors, commonly abbreviated as ESG, have seen an explosion in interest from asset managers in recent years and are rapidly becoming key considerations for asset managers globally. Both retail and institutional investors are becoming increasingly focused on these factors in making investment decisions. It has been shown that corporations' financial results have a positive correlation with the levels of sustainability in their business models, and that ESG investment methodology can help reduce portfolio risk, while generating returns often competitive with those of traditional portfolios [75]. Increased demand combined with competitive performance has led to momentum effects and tremendous growth in ESG assets under management, which has in turn led to an increased demand for ESG due diligence and reliable data. However, one barrier for ESG scoring is the lack of relatively complete and centralized information source. Currently, ESG analysts generally leverage financial reports, investor calls, and regulatory disclosures to collect the the necessary data for proper evaluation, relying on companies to disclose pertinent information. However, the majority of corporate directors do not believe disclosures on sustainability matters are important in helping investors make informed decisions [76], illustrating the importance of incorporating external data sources for additional validation.

At the same time, investors are increasingly demanding more stringent due diligence of the social responsibility levels of their portfolios. In Canada, the total amount of assets under management in Responsible Investing funds has grown 20% year over year 2015-2017 based on the latest growth numbers reported by the Canadian Responsible Investment Association [77]. And according to research from Optimas LLC [78], ESG assets under management now total \$30 trillion globally, up from \$23 trillion in 2016, and projected to grow to \$35 trillion by the end of

2020. According to research from BCG [79], global assets under management in 2018 were at \$74 trillion, meaning close to 50% of Global asset managers are incorporating ESG techniques into their investment methodology [80].

As demand for ESG grows, investors are set to start demanding more accurate and timely responses to ESG issues, and the incorporation of data sources beyond companies' own reports, surveys, and regulatory filings. In the dynamic investment environment, incorporating relevant data from news and social media is primed to become a key component of ESG investment strategies. Machine learning approaches are in turn extremely promising, as they can limit the human cost of continuously monitoring vast volumes of diverse information related to ESG, while providing access to this information in a consistent, real-time reporting stream. We create a prototype for such a model, detail how it was created and how it can be improved; to the best of our knowledge, our study is the first detailed overview of the benefits of modern machine learning, including representation learning and transfer learning, in its potential applications to ESG. We also discuss the uses of such a model to automatically score the level of a company's exposure to ESG, and to build ESG-focused portfolios. To the best of our knowledge,

4.2 NLP AND ESG SCORING LITERATURE REVIEW

ESG investing and ESG portfolio construction are studied primarily from the point of view of financial returns and portfolio construction, as well as applying technology to the creation of ESG portfolios. Both of these areas of research mainly focus on the performance of ESG portfolios defined using human judgement, and have so far lagged behind in using the latest advances in machine learning, and specifically Natural Language Processing (NLP). In this chapter, we show how start of the NLP can be applied to ESG scoring and the construction of ESG portfolios, as well as elucidate how modern NLP techniques can be applied to make such approaches practical.

The ESG industry has been enjoying significant growth, as seen through sharp increases in AUM detailed in the previous section. In addition, numerous other positive trends have emerged. For example, in recent years, portfolios created using ESG criteria have often demonstrated performance competitive with that of broad market portfolios. Climent and Soriano [81] show that between 2001-2009, green portfolios (ESG portfolios focused on the Environmental category) have had adjusted returns in line with conventional mutual funds. Some studies even demonstrate superior performance across specific time periods over the last several years, perhaps due to momentum effects linked to AUM growth: a study by Amundi Asset Management [82] shows that, in the years 2014-2017, incorporating ESG criteria into portfolio construction was a source of excess returns, across all 3 ESG pillars, and in both North America and the Eurozone. And although there is not a clear consensus on the relationship between ESG performance and returns, a large panel study of over 2,200 studies [75] found that 90% of research papers studied found at least a non-negative relationship between ESG ratings on corporate financial performance.

These positive trends have led to significant momentum within the ESG industry, and as a result a rapid spike in interest for ESG data and reliable ESG ratings. Such ESG criteria have been incorporated as part of portfolio construction methodologies; Henriksson et al. [80] provide a survey of ESG usage for portfolio construction, and show that incorporating industry-specific data for ESG scoring has the potential of creating better ESG portfolios. They also show that ESG

criteria have a significant impact on company valuations (but not excess returns), emphasizing the importance of deep and reliable ESG data.

Henriksson et al. [80] also discuss limitations on current ESG rating methodologies, where input data is typically voluntary, is very sparse, and scoring methodologies are variable across ESG data vendors. This is a significant bottleneck to assessing the ESG performance of companies, especially when coupled with the manual effort and required to continuously source and evaluate this data. The manual effort bottleneck is even more significant when dealing with large volumes of text data, such as social media or news data. The approach of ignoring media altogether is unsatisfactory, as many pertinent issues may not be reflected in company financial and regulatory filings, and those filings may significantly lag emerging issues. This creates a strong incentive to use NLP techniques to incorporate this data at scale: several approaches to incorporating such sources via NLP techniques have been studied, summarized below.

Chen et al. [83] incorporate news data into a system that predicts stock market movement returns, fusing representations of news data regarding corporate events (including ESG events) with stock time series representations. Nematzadeh et al. [82] address the need to capture discrete controversies by proposing a clustering approach based on manually engineered features. However, research linking recent advances in NLP to the creation of ESG ratings has been very limited. We propose taking a more focused approach to learning text representation [40] relevant to ESG classification, by creating a human-curated ESG labels at the document level, and utilizing a state-of-the-art NLP representation [84] as a starting point using transfer learning.

4.3 DEFINITIONS AND NOTATION

Definition 4.1. We define the *ESG Category* of each document d_i as $\{c_j(d_i) : i, j \in \mathbb{N}, 1 \leq i \leq N_{documents}, 1 \leq j \leq N_{esg\ categories}, \text{ and } c_j(d_i) \in \{0, 1\}\}$, as an indicator function that specifies whether a document belongs to a particular ESG category. We use these as the labels for training the ESG classifier, and evaluate model performance based on comparing $p_j(d_i)$, the predicted probabilities for each document d_i belonging to category j .

Definition 4.2. We also define a set of documents \mathcal{D}_t as the set of documents generated at time t . We consider daily scores for the purposes of our scoring discussion, so \mathcal{D}_t will refer to all documents generated on a given day.

Definition 4.3. We define an *ESG score* as an aggregation function $F(t, \mathcal{D}_t)$, which combines the predictions p_i on each day in order to evaluate the aggregate performance of a company in terms of each *ESG Category*, based on a set of input documents for that day. These scores are similar in nature to ESG ratings already prevalent within the industry, but we focus on automating their evaluation and detail several approaches to creating company-level scores using a document-level NLP model. We describe some alternatives for defining $F(t, \mathcal{D}_t)$ in Section 5.4.

4.4 APPROACH

An essential component to our approach is a methodology for processing unstructured text data associated with many documents (Tweets in the context of our study). Historically, a common

approach for such a task was creating text ontologies, or lists of words, which could then be used to separate documents into topical categories [85]. Recent advancements in the areas of deep learning have pushed the state-of-the-art in similar NLP tasks towards Transformer-based deep learning model architectures such as BERT [84]. As is typical with modern NLP benchmarks such as GLUE [86], we take the approach of curating a "labeled" data set classification a sample of ESG tweets into topic areas, detailed in Section 4.4.

A key advantage of Transformer-based [87] models is the ability to jointly perform tasks at the document level (i.e. classifying documents into topic areas), and token- or word-level tasks (which may involve more complex rules to be parsed out from the document, such a mention of a company in a specific context). In our study, we focus on the classification tasks, given the very short average length of tweets, as well as our desire create initial ESG representations that can then be applied to more complex subsequent tasks, as explored in section 4.4.

We take a standard approach for fine-tuning a BERT language model for document-level classification to build our ESG NLP model, broadly as labelled data generation (4.4), model training (4.4) and evaluation (4.4), and operationalization (4.5) . Our aim is to show how modern state-of-the art NLP approaches can be applied in determining the ESG content of news and social media (using Twitter as a prototype data source), as well as show how such a model can be operationalized to create a near real-time ESG scores. We also aim to illustrate the advantages of such an approach over what was previously discussed on ESG scoring and portfolio construction in literature.

An important benefit of a deep-learning approach is that each subsequent task requires less training data, as the learned representations can then be used in a feedback loop in place of a manual feature engineering for many related tasks. As Transformer-based architectures have the ability to tackle both document- and token-level tasks jointly, the models we use can be applied to even more complex tasks, creating even more robust representations and helping solve some of the key challenges we detail in Sections 4.5 and 5.4.

Data & Labelling

We opt for a document-level classification approach in designing the learning task for our initial NLP model. The primary application for our NLP model we considered was to measure the average number of documents linked to ESG over a prolonged period of time. Although the NLP model we chose supports both document and token (word) level tasks, we opt to keep our initial NLP modeling at the document level. This is primarily due the low average token lengths of our documents (tweets), as well as our initial goal being an aggregation of documents to determine relative levels of ESG risk.

- We align our ESG categories to MSCI ratings [81], and select a number of topic based on likelihood to be observed in the social media sphere.
- Our data was obtained using the Twitter API; in order to make the amount of manual labeling tractable, we pre-filter each ESG category with a set of keywords, which were then further refined using a level of human review.
- Our final training data comprises of labeled pre-filtered tweets (both relevant and irrelevant to ESG), as well supplementary unfiltered tweets labeled negative (assumed to be ground-

truth negatives due to the sub-1% incidence of ESG topics occurring in randomly sampled tweets, which we expect will have minimal impact on the model). Our final training dataset contains 6K unique tweets, with 1,468 tweets labeled as belonging to one of the ESG topics, and the remainder deemed unrelated to ESG issues after a level of human review.

Our data set covers 10 ESG topics:

- Governance: Business Ethics, Anti-Competitive Practices, Corruption & Instability.
- Social: Discrimination, Health & Demographic Risk, Supply Chain Labour Standards or Labour Management, Privacy & Data Security, Product Liability.
 - The Discrimination category was added by the RiskLab team due to the prevalence of such issues being expressed on social media.
- Environmental: Climate Change and Carbon Emissions, Toxic Emissions & Waste.
- We also model whether a particular document represents an ESG risk, to account for cases where the document captured represents a positive action towards a particular ESG category (e.g. a company announcing a significant donation to fight climate change).

Model Training

We utilize a BERT classifier for this problem [84]. BERT is a pre-trained language model, which carries the benefit of being initialized with the ability to map documents to good general representations. These representations can be used to process documents in a variety of ways, including for classification, which is the primary capability we leverage. In addition, we are also able to specialize or "fine-tune" the model to our specific task of ESG analysis using the labeled dataset we created, which generally performs better than training a standalone model [88].

We utilize a standard pre-training approach with several slight modifications. BERT has recently demonstrated state-of-the-art result on a variety of NLP tasks, and has shown to be robust across many different domains, and ultimately yielded as the best performance across other Transformer-based [87] architectures tested, such as ALBERT [89]. We utilize the pre-trained classifier version as provided by HuggingFace [90] due to the well known benefits of transfer learning [88], which allows us to train a better specialized model with less training data by utilizing a general language model as a starting point. We also add an additional fully-connected layer between the BERT embedding and output layers, in order to simplify fine-tuning. This is achieved through having the initial classifier (trained with the BERT encoder layers frozen) reach a better minimum prior to fine-tuning the BERT layers, as opposed to a single linear output layer. This therefore limits the changes required to the encoder layers of our pre-trained BERT when fine tuning.

Below is the full list of hyperparameters we used to train our classifier:

- We use an output size of 768 and a max sequence length of 128 for the BERT encoder layers, a batch size of 32, and a dropout rate of 0.1.
- We add one hidden layer, of dimension 256, right after the initial (CLS token) BERT embedding, with dropout of 0.5.

- We use the binary cross entropy loss for each category, as the ESG issues may not be mutually exclusive.
- We utilize a two-stage pre-training approach, with a patience of 5 epochs. In the first phase, we train the hidden and output layer; in the second, the full network (with all 10 BERT encoder layers) is fine-tuned.
- We utilize the RADAM [91] optimizer with learning rates of $1e^{-4}$ during the first stage of training, and $2e^{-5}$ during the second stage of training (where BERT encoder layers are fine-tuned); the RADAM optimizer simplifies the learning rate hyperparameter search, and removes the need running "warm-up" epochs. We used no warm-up for training of our model.

Model Evaluation

We optimize our model based on cross-entropy loss, with a test set fixed as a randomly 30% sample of our full dataset.

We evaluate our final models based on the area under the Precision-Recall curve (PRAUC), as well as ROC AUCs for each class to assess model performance. We find that this model, although a prototype, showcases the potential of applying such models at scale. The ROC and PR AUCs for each class, as computed on our holdout set, are presented in Table 4.1.

ESG Category	ROC AUC	PR AUC
Anti-Competitive Practices	0.97	0.57
Business Ethics	0.88	0.36
Climate Change	0.94	0.31
Corruption & Instability	0.98	0.55
Discrimination	0.99	0.71
Health & Demographic Risk	0.91	0.19
Privacy & Data Security	0.97	0.66
Product Liability	0.96	0.66
Supply Chain Labour Policies	0.97	0.76
Toxic Emissions & Waste	0.96	0.41
ESG Risk	0.90	0.22
Not an ESG Risk	0.91	0.51

Table 4.1: NLP Model Evaluation

These results, although they can undoubtedly further improved through collecting additional ground-truth observations, already showcase the potential for applying these models in production. Consider the Precision-Recall curve in Figure 4.1. Our model for this class has a precision of 60% at an 60% recall, meaning the majority of documents pertaining to ESG issues, with 60% of the retrieved documents being true examples of Privacy & Data Security issues. These performance

measures are promising, but given the massive volumes of social media and news data generated daily, would need to be improved further to create a production-ready ESG scoring solution.

It is likely that significant improvements can be made with the addition of more labeled data, as the performance of deep learning models has historically been vastly improved by scaling up the amount of data available for training. The ability to squeeze out additional performance by scaling deep learning models up as additional data becomes available has been consistent across domains [92, 84].

As such, the main challenge in creating robust classifiers across different document types and ESG issues lies with curating a trusted dataset with a sufficient number of diverse examples. We also emphasize that these results are based on training and evaluation the model using Twitter data, and the model would likely need to be fine-tuned in order to apply it to other data sources, such as news or SEC filings. A key benefit to our work, however, is to create initial general ESG representations, that can then be fine-tuned for more complex tasks by collecting additional data.

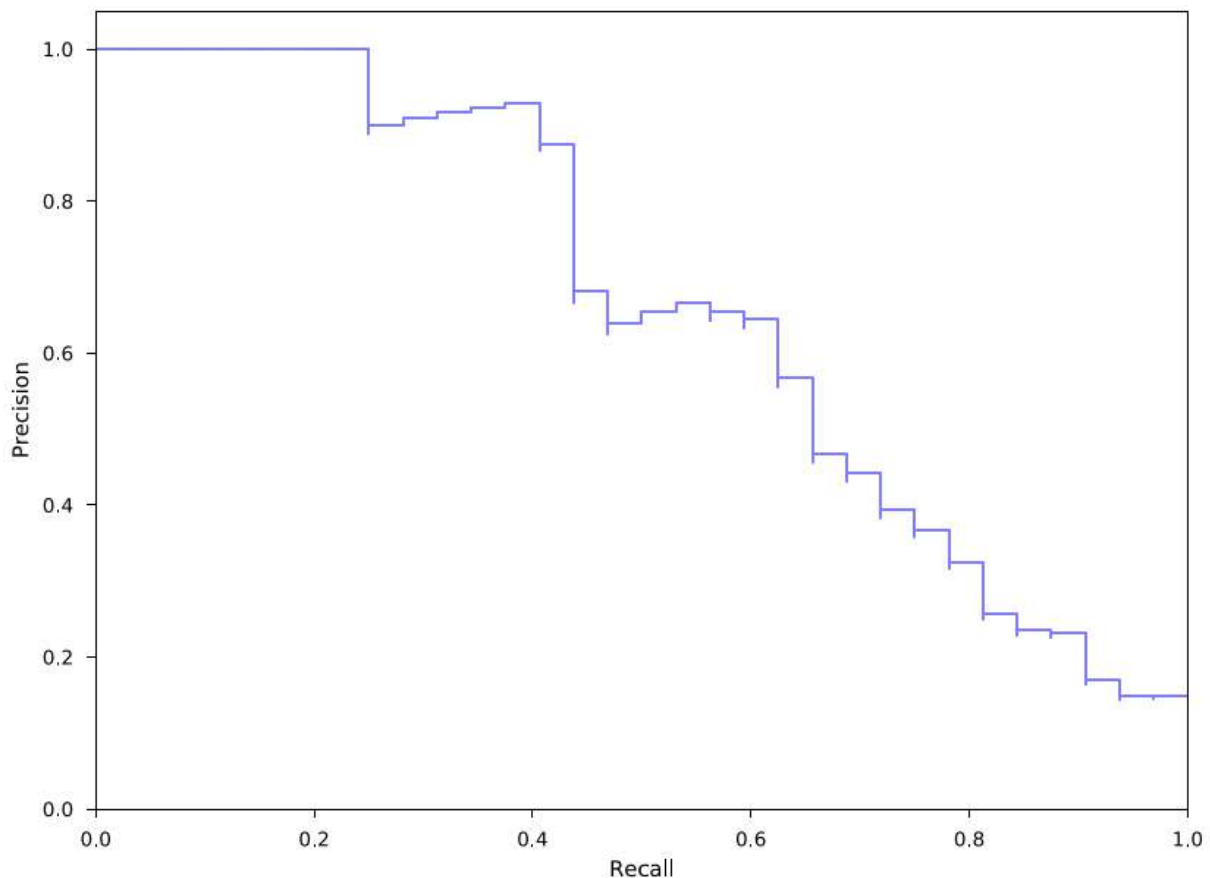


Figure 4.1: Precision-Recall: Privacy & Data Security

Collecting such data would allow us to create even more robust representations of text data suitable towards evaluation ESG issues. These representations can then be used for creating automated, stable and standardized ESG scores that reflect the overarching objectives of ESG investing. We discuss how a document-level model can be used to create company-level scores, as well as the additional tasks our proposed system would need to incorporate in production, below.

Exploring ESG Representations

Representation learning enables much greater transferability of models across tasks, which is perhaps the primary benefit of modern deep-learning driven NLP. Some of these benefits can be illustrated visually by examining the inner workings of our network, which we showcase below. As our model learns to distinguish between various ESG issues (and non-issues), intermediate "representations" of text data are created inside the network. We can extract these representations to help us better understand the totality of ESG issues we encounter, and later on use these same representations to help us tackle more complex tasks from a better starting point.

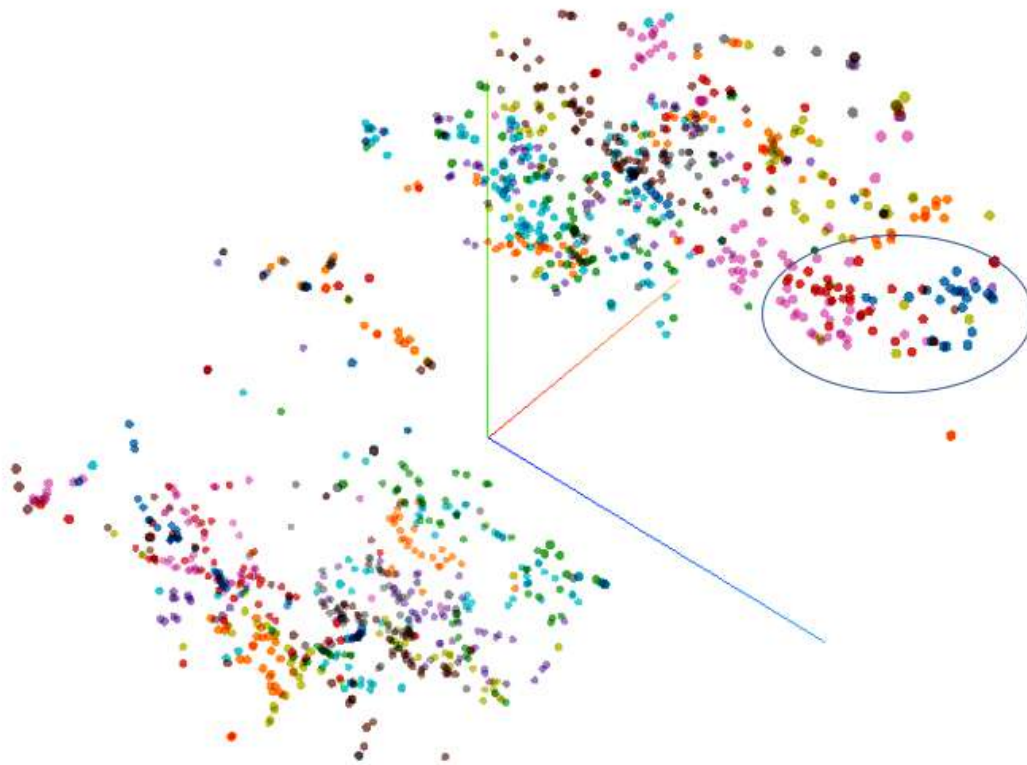


Figure 4.2: Clusters corresponding to ESG issues

Figure 4.2 shows a simplified view of ESG issues in our evaluation data set, plotted using the extracted representations from our model. If two tweets are close to one another in this chart, their numerical representations are similar. The 3 larger "clusters" loosely correspond to Environmental, Social, and Governance issues. The colors represent various issues or themes, determined based on how similar the learned representations are across documents.

Even with a moderate amount of labeled data, our model is already able to distinguish various sub-issues and emerging themes. For example, a common issue with privacy and data security is data breaches, and indeed the blue cluster in Figure 4.3 contains several examples of data breaches being discussed in the public domain.

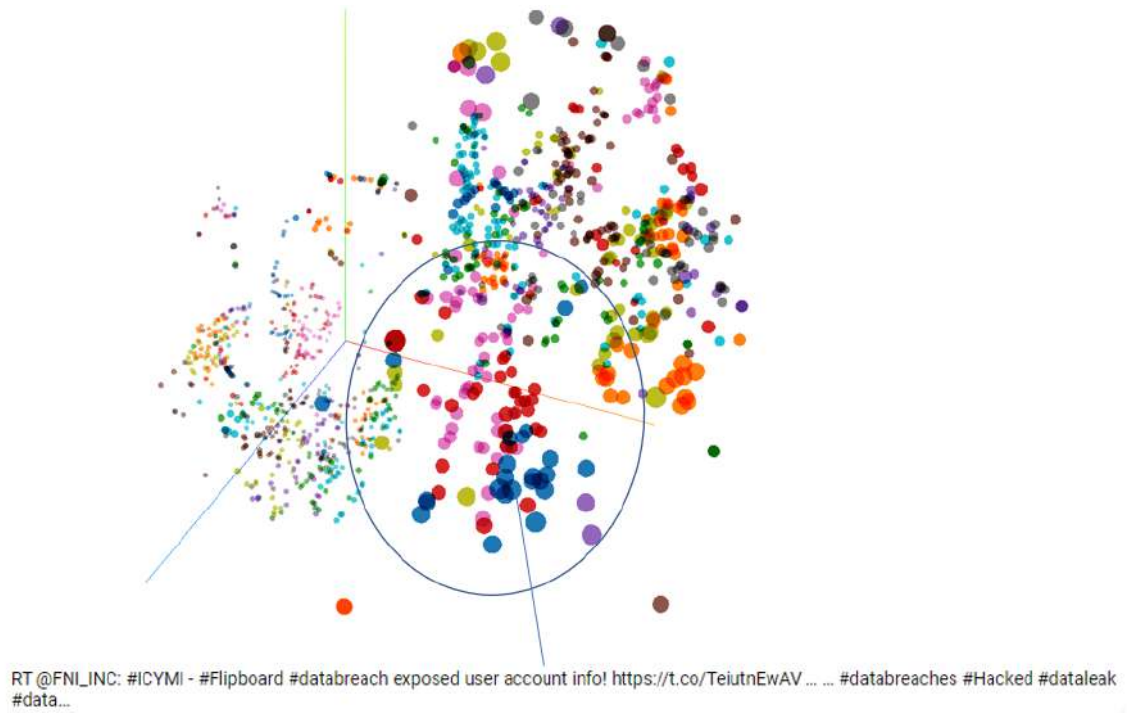


Figure 4.3: The data breach cluster

On the other hand, similar discussions that appear very similar can represent little ESG risk, as seen in Figure 4.4. The nearby pink cluster contains tweets related to cybersecurity education, and discussions on how to limit cyber risk. Both of these issues belong to the Privacy & Data Security category of ESG risk, but are being treated differently by our model because of their relation to other classes, with the pink cluster naturally separating and on average being viewed as less likely to constitute an ESG risk by our model.

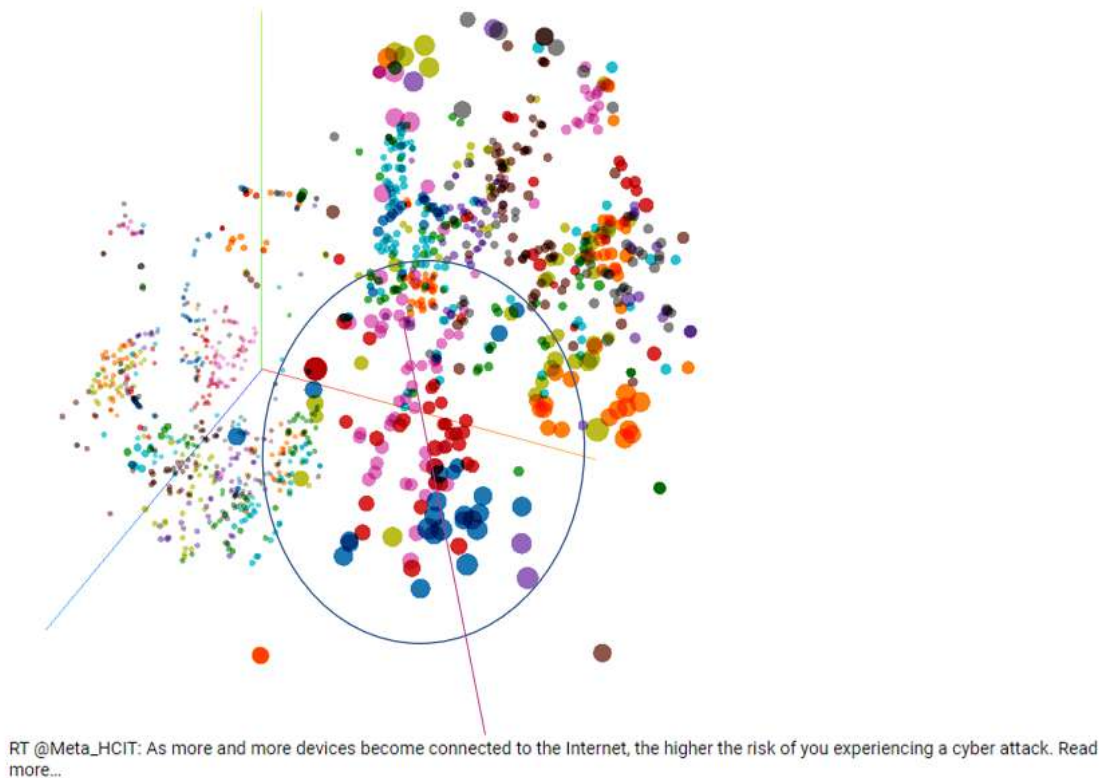


Figure 4.4: Risk-free cluster

It is encouraging that our model is able to distinguish these subtle differences, and naturally separates risky and non-risky issues. This is done in part by priming our model with labels in addition to ESG categories, such as whether a tweet corresponds to an ESG risk or a positive action related to an ESG issue. By adding these kinds of additional information during the labeling processing and representing these as additional training tasks, we expect the learned representations to become more useful towards understanding ESG risk. Examining representation is a valuable technique, but also carries a very practical and tangible benefit for ESG risk detection. Measuring "drift" in the representation stage can be indicative of emerging risks previously unseen in the data, and these types of analyses can also inform the labeling strategy and in turn lead to even better representations, creating a powerful feedback loop.

4.5 USING NLP MODEL RESULTS FOR ESG SCORING

Our chapter demonstrates how state-of-the-art machine learning techniques can be applied to improve the parsing of text data, and demonstrates how such models can be created by detailing the creation of our own prototype model. In order to make the outputs of the model we describe actionable, one has to deal with the additional complexity of converting document-level model outputs, which roughly correspond to the probability of a document belonging to a particular ESG category, into company-level scores. These scores, which update continuously, can then be utilized to construct portfolios that continuously pivot away from ESG exposure, which may be in closer alignment to investor preferences. There are many different ways to accomplish this, for

example by incorporating the scores into the views vector in a Black-Litterman model [93]. The scores can also be used as an alternative, or in addition, to existing ESG ratings.

We propose several alternative approaches through which an ESG score can be constructed from unstructured text data sources. All of our proposed approaches involve aggregating the output of the classifier after it is applied to a large number of documents over a period of time. There are several practical considerations that need to be considered to adopt an automated document classifier into an ESG scoring pipeline. One approach is to simply construct a score based on the average predicted probabilities for each ESG Category, by dividing the sum predicted probabilities $p_j(d_i)$ for each category j by the total number of documents for the day N_t .

$$R_j(t) = \sum_{d_i \in \mathcal{D}_t} \frac{1}{N_t} p_j(d_i)$$

Although intuitive, this approach relies on the learned probability estimates for each class corresponding to their real-life probability distributions; when dealing with text documents, this approach is often unreliable as the distribution of the training data may vary in proportion to the real-life data, and the distribution may shift significantly over time. This leads to the estimated probabilities oftentimes not being meaningful as direct estimates of real-life probabilities. A more robust approach may be to define the score by using our predicted class labels:

$$R_i(t) = \sum_{d_j \in \mathcal{D}_t} \mathbb{1}_i(p_i(d_j))$$

Where $\mathbb{1}_i$ is the indicator function corresponding to class specific cut-offs ε_i , such that:

$$\mathbb{1}_i(p_i, \varepsilon_i) = \begin{cases} 1, & \text{if } p_i \geq \varepsilon_i \\ 0, & \text{if } p_i < \varepsilon_i \end{cases}$$

Although this approach can be an improvement, it still presents a handful of complications: probability outputs may be proportionally off depending on ratios of training data supplied to the model, and may require manual tuning, and so thresholds ε_i can be difficult to set for each ESG category. In addition, such aggregation approaches are still susceptible to unrelated spikes in chatter (e.g. viral marketing campaign may improve overall ESG scores).

Another improvement may be to first group documents into adverse events. This may be desired as it more closely reflects the overarching business purpose behind ESG monitoring, helping identify emerging issues and controversies by grouping related documents together. Leveraging representations as shown in Figures 4.3 and 4.4 can be extremely useful not only for detecting new emerging events, but also for grouping similar documents into events to link documents pertaining to the same issue. This is also an area where token-level capabilities of Transformer models can be leveraged extensively, by building labeled data for extracting mentions of specific types of events or relationships.

With all of these approaches, there is a number of additional considerations that have to be accounted for in order to aggregate document-level model outputs into useful company-level

scores. These considerations may have different optimal ranges depending on the applications for the ESG score, and include:

1. Period of time across which ESG scores are computed; as mentions around ESG topics as well as ESG disclosures may not occur very frequently for most companies, a longer time period may be required to ensure that statistically relevant samples of documents are collected, and meaningful changes are being measured; at the same time, scores should be assessed frequently enough to leverage near real-time capabilities as a core advantage of an automated approach. A reasonable approach to combining longer time periods with more frequent updates is to leverage a time-discounting approach such as exponential smoothing.
2. ESG scores need to be carefully normalized, to account for naturally-occurring differences such as some companies having higher average levels of discourse around them than others, which companies should not be penalized for; in addition, spikes in volume of mentions may not always correspond to spikes in the number of adverse ESG events (Retweets are a basic example); therefore, some level of event de-duplication as described above should be employed, although the volume and sentiment of discussion around a specific event is also an important consideration
3. Not all ESG categories should be equally weighted, but it is difficult to assign relative importance to different ESG categories objectively. As such, our proposed approach to leveraging multiple ESG categories is to provide relative weights as a user input where appropriate for the use case. Alternatively, weights can also be learned by leveraging the created ESG scores in a relevant supervised learning task as a proxy, such as whether a security passed ESG screening in the past, whenever such data is available.

4.6 ADDITIONAL RESEARCH AVENUES

One of the primary areas of future work will be to improve the training data to create a ESG dataset benchmark for NLP, initially focusing on social media data. Given the varied data sources that the industry relies on, it is likely that several different types of NLP models will be need to incorporate diverse text data such as news, SEC filings, and ESG disclosures. We believe that having benchmark datasets for all of these will be critical to the evolution of the ESG data industry, and will look to contribute to the creation of such benchmarks. Some of the improvements we will aim to incorporate to create the first NLP benchmark for ESG will be:

1. Implement redundancy, such as majority voting, around each proposed document label to ensure data quality
2. Obtain additional training data to ensure broader coverage across time periods, companies, and events
3. Expand the dataset beyond Twitter and into other data sources (other social media such as Reddit, blogs and forums, etc., news, SEC filings, ESG disclosures)
4. Expand the dataset to additional token-level tasks described below

There are also additional NLP tasks that would have to be incorporated into an NLP system that captures ESG attitudes across a sufficiently wide number of sources. For example, token level entity recognition models can be used to jointly detect mentions of ESG issues together

with references to a company, directly linking a company to an event. In addition, the action of a particular company in relation to an ESG topic may be positive or negative, as captured in the "ESG Risk" classification component of our model. An NLP ESG scoring system would need to appropriately reflect the correct attitudes and relationships in order to limit false positives, especially in datasets with extremely low signal-to-noise ratios such as social media.

We will also aim to expand the number of ESG issues being considered for this problem, and capture additional issues prevalent in public discourse but not captured in traditional ESG frameworks.

We will also expand this work into creating ESG portfolios, and evaluate various ESG scoring and ESG portfolio construction methodologies as discussed above, and their impacts on portfolio construction, including stability, re-weighting and associated transaction costs. Lastly, we will evaluate ESG ratings for their efficacy as an additional criteria for portfolio construction, and their impact on portfolio risk-return profiles.

EXTERNALITY-AWARE PORTFOLIO THEORY

This chapter proposes an approach that combines modern machine learning techniques in Natural Language Processing with portfolio optimization to incorporate views of companies' ESG performance. This is automatically done through curating and subsequently converting large scale news-data into portfolio management decisions. In the chapter, a machine learning news data classifier is trained to automatically identify several key ESG issues in news data over time. Then, these issues are aggregated over time to generate a “views vector” under the Black-Litterman portfolio framework, after which the performance of an ESG-tilted portfolio is finally compared against a standard Black-Litterman portfolio. Additionally, it is shown how this can be achieved at scale, in a fully automated manner, and with consistency over large periods of times. The methodology in this chapter thus demonstrates a reasonable and agile method for asset managers to incorporate ESG considerations into their portfolios free of any exclusionary frameworks and without sacrificing performance.

5.1 EXTERNALITIES AND PORTFOLIO THEORY LITERATURE REVIEW

The last several years have seen an explosion of interest in ESG, and a continued trend of growth in assets under management for ESG portfolios even throughout the COVID-19 pandemic. Recent years saw some of the world's largest and more prominent investors, including the world's largest private fund manager, BlackRock, and prominent sovereign and pension funds like Japan's GPIF, signal a willingness to commit to ESG investment principles. A recent research report by Morningstar [94] cites continued growth, with the number of sustainable passive funds tripling in the trailing five years leading to June 2019. Sustainable funds have been shown to benefit from significantly higher net inflow of funds in recent years [95], which is consistent with overall changes in assets under management.

The COVID-19 pandemic has served to accelerate these trends. A recent survey by J.P. Morgan demonstrated that the majority of investors see the crisis as a catalyst for ESG investing [96]. This appears to have been consistent with empirical performance: Canada has seen an especially significant uptrend in ESG investing according to TD Securities, with inflows into passive sustainable funds already more than tripling both 2018 and 2019 numbers as of June 2020 [97]. Additionally, recent studies such as S&P Global's paper [98], which focused on exchange-traded ESG funds and ESG mutual funds, as well as a Morningstar's [99], which focused on large-cap US ESG equity indices, show ESG funds outperforming their peers during the crisis. Although these results only represent a short period of time and are not based on robust statistical practices, at the very least, the press generated has the potential to push the demand for ESG funds even further.

In spite of the positive indicators detailed above, ESG data remains a significant challenge for the industry. Self-reported ESG data are extremely challenging to manage, with inconsistency and varying imputation methodologies bringing significant challenges [100]; a recent study has even shown that increased corporate disclosures tends to produce higher variability of ESG ratings [101]. The lack of standardization in the rating industry, combined with views on a specific company impacting a rater’s views of ESG performance across specific categories [102], mean that the availability and quality of ESG data has not kept up with the growth of the industry. As such, we propose a more automated approach be taken to incorporating unstructured ESG information into ESG scores and portfolios, which we believe has the advantages of being more principled and likely to be more trusted by stakeholders.

5.2 FRAMEWORK

The purpose of this chapter is to present a framework to develop an agile ESG approach to portfolio construction, tilted towards increasing allocations to companies with few mentions of ESG issues in news articles over a long time horizon. Our framework consists of three steps: first, we use news data sources (specifically, New York Times data, obtained via the NYT Developer API [103]), which provides historical author-curated dataset tagging that we use as proxies for several key ESG categories. Second, to address the fact that these tags may not be consistently provided, and can vary in their meaning, we introduce a weak supervision approach to create a useful ESG classifier. Specifically, we build a machine learning model using these tags in order to ensure that the ESG scoring is consistent when applied over several decades. This is achieved by isolating the time periods during which tagging was most consistent for tags related to ESG for use as weak labels. Finally, we aggregate the results of our model applied at the article level to generate ESG signals and apply a modified Black-Litterman model [93] to construct ESG portfolios. These portfolios uniquely take into account both ESG and investment risk criteria to create optimized portfolios. We end the chapter with an empirical study of the performance of ESG portfolios constructed through a thorough back-testing setting.

Data Collection & Labelling

Our dataset covers the following ESG topics:

- Governance: Business Ethics, Anti-Competitive Practices, Corruption & Instability
- Social: Health & Demographic Risk, Supply Chain Labour Standards or Labour Management, Privacy & Data Security
- Environmental: Climate Change or Carbon Emissions, Product Quality & Safety, Toxic Emissions & Waste

We utilize 2 key sources for ESG data in this study: New York Times news articles from 1998 to mid 2019 (overlapping with the availability of Compustat dataset described below), as well as equity price data for companies in the S&P 500 index, aligned to the same time period.

In order to generate an appropriate corpus of News Articles relevant to ESG topics, we leverage New York Times Developer API [103] to collect News Articles from the business section of the

Table 5.1: BERT Model Hyperparameter Choices

Parameter	Definition	Value
Stage 1 Learning Rate	Learning rate during stage 1 of training (classification layer)	10^{-3}
Stage 2 Learning Rate	Learning rate during stage 2 of training (classification and encoder layer)	10^{-4}
Early Stopping Eval Limit	Consecutive evals with no improvement before stopping	5
Max Sequence Length	Max sequence length for BERT	256
Batch Size	Batch Size used in training	16
Evaluate Every n Samples	Run evaluation every n samples	400

New York times, published from 1998 to mid 2019. This corpus contains hundreds of thousands of articles related to a variety of news topics, some of which are relevant for ESG. We utilize author-provided tags as weak supervised labels, using relevant tags as proxies for ESG categories listed above. To ensure class imbalance is not a concern during training, we randomly sample the corpus keeping all articles with tags relevant to an ESG category, combined with a sample of articles without relevant tags. In addition, we use company names (also provided as tags via the New York Times API) and, thus, did not use a separate named entity recognition model to identify companies.

Our equity price dataset was obtained from Compustat, for which we use data starting in 1998 and ending with mid-2019 for our portfolio analysis since S&P index data is no longer available past this date. We also apply several data quality improvements to our dataset to ensure consistency over time, such as imputing any missing trading dates and accounting for companies that trade under multiple tickers. Outside of these changes we take the price data as is, and use it solely for portfolio backtesting described in the Portfolio Construction section .

Finally, to align news article data against Compustat data we use a collection of text similarity algorithms to generate a dictionary between Company names available in the NYT and the company names associated with Compustat data, and then evaluate and refine this dictionary, curating a final map between NYT company names and S&P 500 tickers.

Model Training

We train our model as a standard BERT [84] classifier, using a weak supervision approach by leveraging author-provided tags as proxies for ESG categories of interest. We use the pre-trained model from HuggingFace [90], and apply two-stage fine-tuning to train our model, whereby we first freeze the encoder layers and only train the classification layers of our model, and then train the full model briefly after our early-stopping criteria are triggered. We train a single model to classify all of our ESG categories simultaneously in order to combine the labeled dataset from individual categories and create good general ESG text embeddings, and select our hyperparameters based on the weighted average precision scores across our classes. Our final hyperparameter choices are presented in Exhibit 5.1.

We train one network to model for all of the ESG categories described in the ESG News Classifier and Weak Supervision section. We do this in order to utilize all of our labeled data to fine tune a single representation, thus boosting the total number of labels available for fine tuning; another benefit of this approach is to create better general representations of news data for ESG, which can then be applied towards some potential improvements to the model described in the [Additional Research Avenues](#) section.

Portfolio Construction

A key contribution we make is to demonstrate a working example of how the results of a machine-learning NLP model may be used to continuously incorporate views on ESG performance into portfolio construction decisions, even in the presence of noisy labels for ESG categories. We believe our proposed approach addresses a number of key issues inherent in incorporating unstructured data into portfolio construction decisions, demonstrates the technical feasibility of such a combination, and showcases interesting results in the process.

There is no single clear approach through which the results of an NLP model, even when aggregated into an ESG score at a company level, can be used to make investment decisions. We accomplish this by leveraging a widely used Black-Litterman Model [93] optimization approach to incorporate ESG views, which is a very natural approach for this task. The Black-Litterman model allows portfolio managers to incorporate a Bayesian vector of views, in combination with an imputed vector of views expressed by the market, as priors. These priors, used as inputs into the classical Markowitz portfolio optimization algorithm [2], have the effect of pivoting away from investments the portfolio manager views more negatively relative to the market. Converting scored ESG data into a views vector is therefore particularly pivotal to our approach. There are several key decision points to be made in order to convert news data ESG scores into signals. We elaborate on each of these below:

- Initially, our machine learning classifier outputs a number between 0 and 1 for each article, which roughly corresponds to the probability of the article belonging to a particular ESG category. As seen in the precision-recall estimates we present in the ESG News Classifier and Weak Supervision section, we see significant differences across categories in terms of the number of false positives one has to tolerate in order to meet a particular recall cut-off. To add to this complexity, our model performance is often under-estimated by human error and labeling inconsistency, where articles labeled by our model but not by NYT are in fact legitimate members of the category. These considerations lead us to pick a specific probability cut-off for each category for treating an article as indication of ESG risk.
- After these signals are generated, we need to account for natural variance in reporting, both over time as the news cycle moves towards big newsworthy events and away from reporting on corporate issues, as well as companies which simply get more press on average. We accomplish this by first aggregating signals over the last quarter, clipping the highest signals at the 95th percentile (making the overall distribution less skewed), and scaling the signals so that the 95th-percentile signal receives the value of 1.

- Lastly, the normalized weights get turned into a views vector Q . As per the standard Black-Litterman approach [93], we add our views vector to the expected returns vector in order to tilt the portfolio away from companies linked to a significant amount of ESG press.

After producing our ESG views vector as described above, we follow the standard Black-Litterman portfolio optimization approach. As in the seminal work [93]. We compute the posterior returns estimate as over n assets as:

$$E(R) = [(\tau\Sigma)^{-1} + \Omega^{-1}]^{-1}[(\tau\Sigma)^{-1}\Pi + \Sigma^{-1}Q]$$

Where $E(R)$ represents the posterior returns expectation, Σ represents the prior covariance matrix and τ is used as a scalar tuning constant for adding additional uncertainty, Ω as the $n \times n$ uncertainty matrix of views, Π gives the prior vector of expected returns, and Q gives the prior vector of asset manager views. As we discussed above, estimating Q is pivotal to our approach. We further discuss the impact Q has on the make-up of our ESG portfolio in the Portfolio Comparison section.

In addition, we compute the posterior covariance matrix as:

$$\hat{\Sigma} = \Sigma + [(\tau\Sigma)^{-1} + \Omega^{-1}]^{-1},$$

Where $\hat{\Sigma}$ represents the posterior covariance matrix, Σ represents the prior covariance matrix, τ is used as a scaling constant adding additional uncertainty, and Ω represents the portfolio manager's estimate on the uncertainty of their views. We did not compute an ESG-specific estimate Ω , instead using market covariance as the uncertainty matrix estimate. Some ways in which Ω could potentially be estimated when using inputs of a predictive model would be by using model uncertainties to aggregate the total uncertainty of NLP models predictions by using estimates of false positive and false negative rates, as well as the number of predictions made. One could also build on this approach by estimating the statistical distributions and estimating the covariance matrix of model errors.

We use this strategy with the constraint of taking long-only positions to reflect the needs of the long-only ESG-focused investor, and with the optimization objective of maximizing the Sharpe Ratio. Some periods have no feasible solution to the optimization problem with the long-only constraints, during which times we use market weights for both the ESG and the standard Black-Litterman portfolios; the results are summarized in the Portfolio Comparison section.

5.3 RESULTS

ESG News Classifier and Weak Supervision

The choices of hyperparameters detailed in Exhibit 5.1 result in model performance metrics computed by using author-curated tags directly, summarized in Exhibit 5.2. These results are very promising, and based on our analysis of false-positive and false negatives are under-stated (meaning our models are able to generalize well despite the prevalence of missing tags in the historical NYT dataset), as detailed in Exhibit 5.3. Given that the fully labeled hold-out set is

Table 5.2: NLP Model Performance

E/S/G	ESG Sub-Category	ROC AUC	PR AUC
Social	Product Liability	0.92	0.65
Social	Human Capital	0.90	0.62
Social	Privacy	0.91	0.59
Social	Data Security	0.93	0.63
Governance	Anti-Competitive Practices	0.93	0.75
Environmental	Pollution & Waste	0.95	0.70
Environmental	Climate Change	0.96	0.73

Table 5.3: Model Thresholds

E/S/G	ESG Sub-Category	Threshold Precision	Threshold Recall	Sampled Precision at Cut-Off
Social	Product Liability	19%	69%	50%
Social	Human Capital	39%	46%	58%
Social	Privacy	40%	46%	57%
Social	Data Security	49%	31%	56%
Governance	Anti-Competitive Practices	29%	80%	54%
Environmental	Pollution & Waste	17%	62%	58%
Environmental	Climate Change	12%	50%	40%

based on author-provided tasks, we find that in general the true performance of the model for classifying ESG issues is understated by this direct computation. We discuss our estimates of effective performance of the NLP model below, as well as the implications towards using the model within a portfolio optimization framework. We note that good generalization of NLP models despite limitations and inconsistencies in labeled data are expected, and is consistent with recent research on weak supervision in NLP, for example [104].

Based on these results, we are able to choose cut-offs (detailed in Exhibit 5.3) that provide precision-recall trade-offs that are sufficiently high for us to make a determination. We note that the performance of our models at our chosen thresholds is empirically better than what we estimate using NYT labeled data only, due to limitations and a level of inconsistency in historical labeling, as Exhibit 5.3 shows. Our choice of cut-off was made such that the approximate precision of each of our NLP models was $\sim 50\%$, and aimed to maximize recall under this constraint. For example, based on the precision-recall curve we estimate for the Product Liability ESG category as can be seen in Exhibit 5.1, our chosen cut-off allows us to capture $\sim 80\%$ of the total number of mentions of Product and Liability as an ESG issue in relation to S&P500 companies, and we estimate the effective precision as this threshold to be $\sim 50\%$. Other classes similarly yield results that are sufficiently precise to provide accurate estimates of ESG risk exposure when averaging over a large number of news articles. The intuitive results and favorable performance of the Black-Litterman-ESG portfolio, as described in the Portfolio Comparison section, yield credence to satisfactory performance of our NLP models.

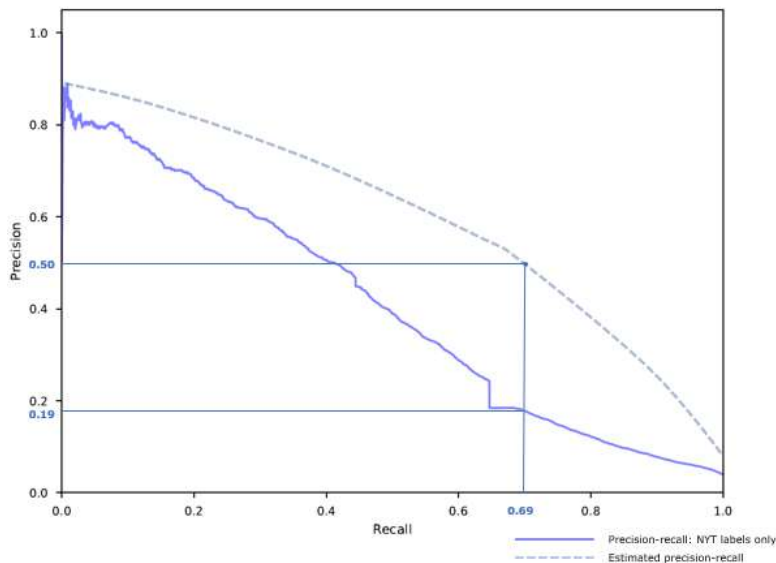


Figure 5.1: Threshold Choice for the Product Liability ESG Category

Portfolio Comparison

Although ESG criteria are generally not performance driven, we compare the historical performance of three ESG portfolios. The first is the market-weighted portfolio provided for reference, the second a Black-Litterman Benchmark portfolio with no additional priors (i.e. a zero-vector used for the portfolio manager’s views), and the third is the Black-Litterman ESG portfolio utilizes the article scores generated by our NLP model to generate a views vector, where a high prevalence of ESG-related news is taken as a negative signal, as described in the Portfolio Construction section above. Exhibit 5.2 shows a comparison of the three portfolios.

Note that the Black-Litterman ESG portfolio performs in-line with the benchmark Black-Litterman portfolio, with both having higher returns than the market portfolio. Exhibit 5.4 includes some additional metrics, which also shows that the performance of the ESG portfolio is in line with the benchmark Black-Litterman portfolio across a range of metrics commonly used in portfolio management. The information ratio is computed using the market portfolio as a benchmark, while the Sharpe Ratio and standard deviation of excess returns are computed using US Treasury spot interest rates. We purposefully avoid running a high number of experiments in order to avoid the pitfalls of P-Hacking in quantitative finance [44], and therefore we do not make the claim that an investment strategy based on ESG investing would produce reliable excess returns in the future. One might hypothesize that this level of performance is a signal of an ESG momentum factor, but a deeper analysis would be required to conclusively determine that. Nevertheless, we believe it is interesting that such a portfolio could have performed in line with an unconstrained portfolio in the past, perhaps indicating possible advantages of implementing ESG portfolio “tilt” strategy, rather than treating ESG considerations as exclusionary constraints.

Exhibit 5.3 are the industries that were under- or over-indexed in the BL-ESG portfolio throughout our testing period. Note much of this is intuitive: the Finance and Insurance industries are under-represented during market downturns, especially during the Great Financial Crisis,



Figure 5.2: Comparison of Market Portfolio, ESG-BL Portfolio and Benchmark BL Portfolio

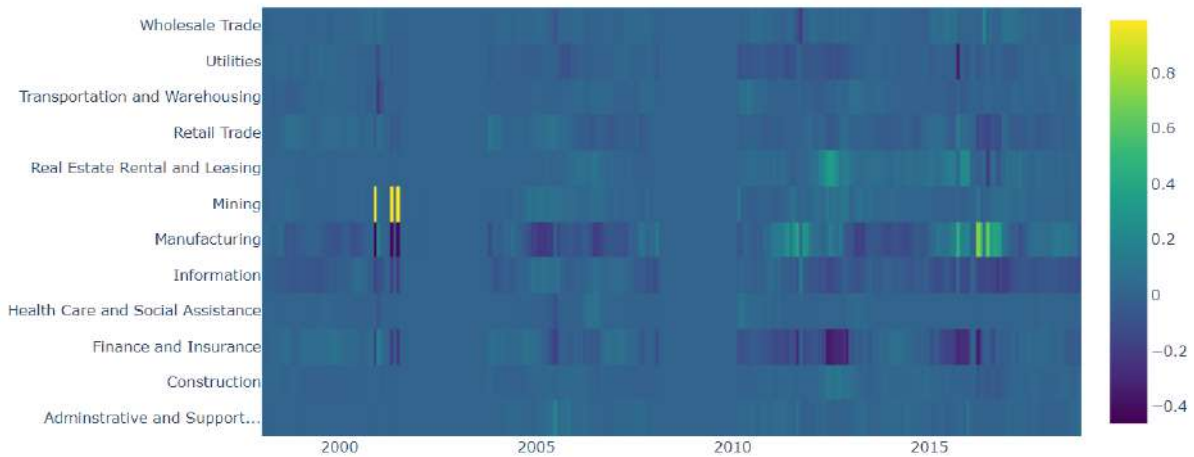


Figure 5.3: Industry under- and over-Indexing for the ESG Portfolio

due to the increases in negative publicity and therefore in mentions of ESG issues. Similarly, the Information industry becomes more under-represented in recent years, as privacy issues have become much more prevalent and drawn additional public scrutiny. Also note that during periods where the market becomes highly correlated, there is no under- or -over indexing as all strategies track the market portfolio during these periods. We also note that the two Black-Litterman portfolios are significantly different, with an average overlap of only 19.85% in their holdings over the periods where Black-Litterman optimization is performed.

5.4 CONCLUSION AND ADDITIONAL RESEARCH AVENUES

In conclusion, we developed and demonstrated an approach that can be used to extract ESG signals from unstructured text data using deep learning models for NLP, then subsequently

Table 5.4: Portfolio Metrics

Portfolio	Information Ratio	Sharpe Ratio	Annualized Returns	Volatility of Excess Returns
BL-ESG Portfolio	0.0147	0.400	9.14%	24.59%
BL-Benchmark Portfolio	0.0142	0.368	8.08%	23.18%
Market Portfolio	0.0	0.237	4.77%	21.63%

incorporate these signals into portfolio construction decisions. Such an approach can generate portfolios that are tilted towards companies with fewer signals of ESG exposure, but is otherwise competitive with a benchmark portfolio across a variety of metrics.

There are a number of opportunities to build on our work described above, and we detail some of the approaches that we deem most promising for improving on our work.

As the state of the art in natural language processing is evolving rapidly, one area of opportunity is incorporating newer NLP models. There are two primary areas of research pushing the state-of-the-art over the large Transformer-based models (BERT [84], ALBERT [89]) we utilized for our NLP work. The first area of research is in pushing the performance on common NLP benchmark tasks and generalizability of NLP models, primarily (but not exclusively [105]) through scaling up transformer architectures to be even larger, e.g. XLNet [106], [107] or GPT-3 [108]. Another key area of research, which is perhaps even more useful in the domain of ESG, is focused on achieving similar results but dramatically scaling down the size of the model, such as DistilBERT [109], TinyBERT [110]. This is very promising due to the large volumes of data related to ESG, where lowering inference costs and reducing / eliminating the need for multiple models (e.g. TF-IDF based pre-filtering) has the potential to save significant engineering and computational costs. One can also incorporate additional detail into estimating the posterior covariance matrix $\hat{\Sigma}$ by incorporating the statistical distribution of NLP model errors to reflect the inherent uncertainty in the views vector.

Another key area of improvement is in collecting additional data, and potentially in combining data from various ESG data sources. Relevant sources of text data for ESG include social media data, news data, regulatory filings, and company ESG disclosures. Building generalized NLP models that are robust to dealing with multiple data sources also has the potential for significantly reducing engineering costs for building ESG scoring systems, and also for creating better robust general representations of ESG text data by pooling labeled data across sources. These better representations can then be used beyond supervised classification tasks and expanded to additional tasks such as detection of ESG events or emerging ESG risks.

REPRESENTATION LEARNING FOR FINANCIAL TIME-SERIES DATA

The dominant approaches for financial portfolio construction are reliant on estimating sample covariance and correlations matrices, which serve as an input into a number of classical portfolio construction techniques. These classical approaches are not forward looking, constrained by the ability to estimate covariance and correlation matrices, and inflexible to incorporating additional information. This chapter proposes a new approach of utilizing learned representations from deep learning networks to augment such classical techniques. This approach is able to incorporate learned estimates of future performance, and can be customized to create tailored representations best suited towards meeting varying financial objectives. This chapter showcases one example of such an embedding, compares and contrasts it with classical approaches to portfolio construction, and looks at additional possibilities for applying representation learning in quantitative finance.

6.1 MOTIVATION

Our work is motivated by viewing covariance and correlation as a special case of a distance metric, or a measure of dissimilarity between securities. Machine learning has been used to learn metrics directly [111], as well as by leveraging deep learning in semi-supervised contexts [112]. Much work has also been put towards developing general-purpose representations [113] for common data sources, classical examples being words [114], text segments [115], [84] and images [116].

In parallel, in the realm of finance, the Efficient Market Hypothesis (EMH) is generally accepted: this hypothesis states that all current information is already factored into a security's price. The EMH is particularly well established in terms of returns data (financial time series and their correlation structure), and although some work exists challenging the EMH [117] in this context, the usefulness of pure time-series based models for statistical arbitrage is still questionable.

In contrast, deep learning offers the flexibility of creating useful representations of complex datasets. Even in cases where direct prediction is not necessarily useful (as in the example presented in the Representations section below), deep representations have proven to be incredibly powerful.

Time series datasets are used extensively for hedging and portfolio construction, despite the difficulty of forecasting individual price movements. Correlation or covariance matrices computed from these time series, being distance metrics themselves, are frequently used for portfolio optimization [2]. We therefore propose an alternative approach for learning distance metrics that can ultimately be used for tasks beyond just portfolio optimization. Furthermore, our approach

has the advantage of incorporating future expectations, as these metrics can be optimized by solving forward-looking tasks. In addition, it offers the flexibility of engineering different metrics by choosing different relevant tasks.

Machine learning has been used extensively in finance, but most applications of machine learning are based around unsupervised techniques. Over the past few years, unsupervised clustering algorithms have found root in portfolio construction [118]. As dealing with time series in general, unsupervised algorithms such as autoencoders, Deep Belief Networks, or PCA have consistently shown to create more workable feature representation of high-dimensional and unstructured time series data [119]. Unfortunately, however, such approaches are limited in their ability to create useful representations as they are trained to solve tasks (such as reconstruction loss) that are unrelated to the ultimate objective. This flaw is principally why our approach utilizes a supervised algorithm instead.

The remainder of the introduction explains the structure of the chapter, while also stating certain general implications of each section. The Technical Background section introduces the conceptual background required for both machine learning and finance. The Definitions and Notation section outlines the preliminary definitions and notation to be used throughout the remainder of the chapter. The Approach and Empirical Implementation section outlines the general process for leveraging learned representations in finance, and outlines a sample empirical implementation for a transformer network to learn the structure of financial time series of NYSE and NASDAQ listed equities. The Evaluation Representations section discusses how these representations can be evaluated. The final sections wrap up the chapter with the significance and potential extensions of our work, specifically the potential to operationalize learned representations for a wider variety of tasks in finance using transfer learning techniques.

6.2 TECHNICAL BACKGROUND

It is our intention to create connections between two vast and well-established fields: finance and machine learning. As such, in hopes of making this chapter more accessible to researchers or professionals in either finance or machine learning, we have provided the following short colloquial backgrounds on each discipline included in this chapter.

Machine Learning

Three interrelated concepts are necessary to discuss for this chapter: transfer learning, representations, and the transformer network. We do, however, assume a very basic knowledge of deep learning and neural networks.

Representations: A learned representation may be thought of as a transformation of data into a more useful space via a machine learning algorithm. The usefulness of re-representing data into another space comes from the task at hand; common benefits are better performance on tasks such as classification, as well as the ability induce a metric on a dataset that did not originate with an intuitive notion of distance. A classical example is that of word representations, or word embeddings. Historically, a big barrier for natural language processing was the high dimensionality of text data, with no obvious way of representing individual words numerically in

a useful manner. It turned out that by training a neural network using a "fill-in-the-blanks" task, the numerical representations created become highly useful [114]. This can be seen in the classical example in Exhibit 6.1, where gender becomes an emergent property of our representation space (as can be seen with the fact that the vector separating "man" and "woman" is the same as the vector separating "king" and "queen"):

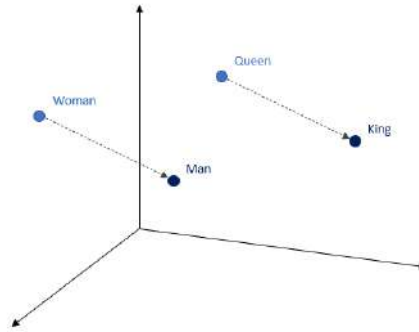


Figure 6.1: Word Embeddings Illustration

Although this result is easy to visualize and understand intuitively, word embeddings have been chiefly responsible for the formidable progress made today in the field of natural language processing. It is worth noting that word embeddings are also incredibly useful, even though the original network used to create these representations has little standalone practical use [114].

The power of deep learning (in comparison to other machine learning algorithms) for the purposes of general representation learning is well known [40]. It has created breakthroughs in fields as diverse as natural language processing, image processing, and biological research. Furthermore, learning state space representations in reinforcement learning has recently propelled AI to dominance in games such as Go and Chess [120]. This potential is well known to extend beyond the specific task at hand, and learned representations have been shown to be useful across a wide variety of adjacent tasks [88].

In this chapter, we will principally be interested in creating a useful representation of financial assets that can generalize across a variety of applications in the domain of finance. Later on in this chapter, we empirically construct a representation for financial assets then proceed to demonstrate some of the applications of said representation.

Transfer Learning: Transfer learning may be defined as the use of the resulting embedding function learned via one machine learning algorithm for the purposes of solving a separate, but similar, task. A common example of this might be using a ResNet representation [116], trained on a substantial amount of images for tackling another, more specific, image classification task.

Transfer learning has become widely used in machine learning, particularly in the fields of natural language and image processing. The significant level of commonality in the datasets and the way one processes such data, regardless of task, has been the primary reason for the wide adoption of transfer learning in these fields. At the same time, transfer learning in finance is relatively under-explored. We hope to show that applying transfer learning techniques to the financial time series domain shows significant promise.

Transformer Networks: Time series datasets have long been tackled with Recurrent Neural Networks, and specifically LSTMs [121]. The "attention mechanism" has been shown to be able to augment LSTM-based architectures and improve results with NLP tasks [122]. More recently, the Transformer architecture has demonstrated that the recurrent network architecture can be abandoned altogether in favor of self attention, ushering in a revolution in the NLP domain [87]. Transformer networks have in the past shown promising results specifically for time series forecasting [123]. In this chapter, we shall utilize the transformer architecture whilst hypothesizing that its ability to capture long-term dependencies and complex interactions will serve for greater accuracy in solving the tasks we assign to our network, and ultimately create better representations.

We refer the reader to a review of the aforementioned papers in this section for a proper introduction to Transformer Networks. For the financially predisposed reader, the main takeaway we'd like to emphasize is simply the adeptness of transformer networks for time series.

Finance

Arguably the most common quantitative metric/statistic asset managers look at when making asset allocations amongst a collection of assets would be sample covariance and correlation.

A prototypical treatment of covariance and or correlation would be as follows (henceforth, "covariance" may be swapped without loss of generality for "correlation"). First, an asset manager would choose a suitable historical time horizon (e.g. 3 months of daily ticks) for a collection of time series' associated to a chosen group of assets. Then, over said time horizon, they would compute the sample covariance. This sample (historical) covariance then might be used for a variety of tasks under the assumption that this past covariance will stay constant for the foreseeable future. Examples of how an asset manager might utilize the covariance could be in constructing a minimum variance portfolio, a risk parity / equal risk contributions portfolio, or trying to understand fundamental relationships between assets via PCA. All techniques mentioned have considerable merit, but all are limited in part due to assumptions of constant covariance.

As discussed previously, the covariance matrix is often used to solve a portfolio optimization task. An approach to solving this oftentimes difficult task is to instead first apply a dimensionality reduction technique. Principal components analysis, PCA, is most commonly chosen in Finance. In contrast to representations learned by neural networks, PCA is a linear technique which learns a transformation of the data with the property that the new representation forms an orthogonal basis in the directions which capture the most amount of sequential variance. We note that PCA has both a probabilistic interpretation (calculated by taking the leading eigenvectors of the covariance matrix as the principal components) and geometric interpretation (linear manifold learning representing a lower-dimension region of the input space in which data density is maximized). [40].

PCA is quintessentially a representation learning technique and, given its level of familiarity to most readers, may serve as a good intuition for representation learning in general. We are, however, more interested in representations learned by deep learning networks for three main reasons:

- PCA is a linear technique, and therefore cannot be applied iteratively (stacked) to create more complex representations, as opposed to layers in a neural network which learn more abstract representations as the complexity of a network is increased, provided a sufficient number of training examples is available.
- PCA takes as an input a simple matrix. We are therefore unable to input tensors of arbitrary dimensionality, and it is left to the user of the technique to create an initial feature representation when working with multi-dimensional data, such as tensors of stacked time series in our case. When working with deep learning representations, we are able to use layers such as convolution and self-attention to work with such high-dimensional data more efficiently, and ultimately create a convenient vector representation directly from a complex tensor.
- PCA is an unsupervised technique, and the representation it learns may not necessarily be optimal for all tasks. We create our deep representations by first solving a relevant task, making it more likely that the abstract representation we learn will be useful for related tasks.

The use of deep learning here is also especially pertinent given the well-known need for capturing the nonlinear nature of relationships in financial markets: Bengio et al. [113] emphasizes that deep learning is able to create “abstract” representations: representations that disentangle the factors of variation present in the input.

In further sections, we will demonstrate the utility of our approach in a common portfolio construction technique of choosing representative securities from a hierarchical clustering of portfolios [118]. We demonstrate that instead of the classical approach of utilizing correlation as an input into the clustering algorithms, one has flexibility in choosing the correct learned representation to define a robust distance metric that fits the user’s criteria, using learned representations.

6.3 DEFINITIONS AND NOTATION

Machine Learning

Definition 6.1. We consider a family of *neural networks* that we train to learn a function $f(X) = \hat{y}$, where X is a tensor representing raw financial time series (such as the log returns of an input asset, its sector, etc.), and \hat{y} is the learned prediction. For our problem, the relevant dimensions are as follows: $X \in \mathbb{R}^T \times d_{\text{feats}}$, $y \in \mathbb{R}^{d_{\text{trgt}}}$, where T is the length of the input time series, d_{feats} is the number of input time series we consider, and d_{trgt} is the number of targets we want to make for each input (e.g. future returns at time t , future volatility at time t , etc.). Lastly, $f(X)$ is found by solving the standard *supervised learning task* of minimizing the *loss function* $\sum_{j=1}^{d_{\text{trgt}}} L_j(\hat{y}_j, y_j)$.

Definition 6.2. We define the *representation* for each asset as the learned outputs of our network for the final network layer (prior to the output layer), $X_{\text{embed}} = F_{\text{embed}}(X) = \{f_j(X) : 1 \leq j \leq$

d_{embed} }, where X is an input tensor, d_{embed} is the dimension of the final neural layer, and $f_j(X)$ is the value neuron j takes for input X .

The motivation for choosing F_{embed} to output the layer preceding our output layer is the fact that the final decision function, $F_{\text{output}}(X_{\text{embed}})$, is linear. Due to the linearity, we therefore establish a sense of well-behavedness for our representation. We extract these embeddings by "stopping short" of producing our final output, extracting values $F_{\text{embed}}(X) = X_{\text{embed}}$, with $X \in \mathbb{R}^T \times d_{\text{feats}}$ as above, and $X_{\text{embed}} \in \mathbb{R}^d$.

Finance

Definition 6.3. If we consider a collection of N assets, we define the *price* and *log returns* (or, for brevity, *returns*) of asset $1 \leq i \leq N$ at time t to be $P_i(t)$ and $r_i(t)$ respectively, where:

$$r_i(t) := \log(P_i(t)) - \log(P_i(t-1))$$

Further, we refer to $\{P_i(t) : t_0 \leq t \leq T\}$ and $\{r_i(t) : t_0 \leq t \leq T\}$ as the *price* and *returns time-series* of asset i . For this chapter, we will always assume time series are finite ($T \in \mathbb{N}, T < \infty$).

One of the most important statistics that asset managers look at when making asset allocations amongst a collection of assets would be sample covariance/correlation, defined in the classical way on $\{P_i(t) : t_0 \leq t \leq t_1\}$. Their interest comes from an appeal to minimizing risk - the more correlation in a portfolio, the riskier said portfolio. That said, once a position/portfolio is established, as covariances amongst assets change through time, asset managers will want to update the way they utilize covariance by rebalancing. Therefore, after some period of time, say $k \in \mathbb{N}$, the asset manager will rebalance based on $\{P_i(t) : t_0 + k \leq t \leq t_1 + k\}$. If we therefore continue looking at covariance in this updated scenario for $k, 2k, 3k, \dots$, we refer to $|t_1 - t_0|$ as the "window", and k as the period of time for which we "roll-over". The choice of k and $|t_1 - t_0|$ being an arbitrary one, however, is yet another disadvantage of this classical approach.

Definition 6.4. We train our network by predicting, jointly, several *target metrics* $\{y_j : 1 \leq j \leq d_{\text{trgt}}\}$, using the time series $r_i(t)$. We therefore learn a function $f : \mathbb{R}^T \times d_{\text{feats}} \rightarrow \mathbb{R}^{d_{\text{trgt}}}$ that maps the feature time our features time series, of dimension $\mathbb{R}^T \times d_{\text{feats}}$, to a prediction for each of our targets, of dimension $\mathbb{R}^{d_{\text{trgt}}}$. The definitions of each target we used to train the network described in this chapter may be found in Section 4.

Definition 6.5. In order to predict the targets $\{y_j\}$, we add some additional information to the returns $r_i(t)$. We concatenate the time series $\{r_i(t)\}$ with the time series $\{t\}$ that captures the relative position of each return (instead of positional encoding as used by Vaswani [87]), as well as several additional time series of market returns as additional features (details outlined in the Empirical Implementation section).

We concatenate all of the time series above into a matrix X , of dimension $T \times d_{\text{feats}}$, with d_{feats} as the number of input time series used.

6.4 APPROACH AND EMPIRICAL IMPLEMENTATION

General Process for Leveraging Learned Representations

In general, our proposal for a new, highly general "learned representation for financial assets" follows the following schema. We describe this proposal from the outlook of a financial analyst.

1. Choose a collection of financial assets for which a financial analyst would be interested in understanding their interrelationships to a higher degree under a particular lens; e.g., portfolio optimization, behaviour in volatile markets, etc.
2. Make a priori and collect *target metrics* or tasks $\{y_j\}$, that should be related to the lens that the analyst would like to look at the interrelationships under; e.g., log returns, covariance, etc.
3. Train a neural network on said target metrics and extract the related representation.
4. Evaluate and use the learned representation in the context that it was created for.

This process therefore allows for the substitution of machine learning for classical statistics found in finance so as to create a highly bespoke/customizable (in terms of the lens an analyst would like through) representation of any collection of financial assets.

We showcase on particular implementation of this process below.

Data and Prediction Task

The dataset used for our empirical analysis was the daily adjusted-close price history of all stocks listed on the New York & NASDAQ Stock Exchanges from 2000 to 2018. The only filters we applied is to only use tickers with at least 128 days of returns history at the time they are sampled, and removing tickers that had data issues with their adjusted closing prices as observed on Yahoo Finance (tickers that had more than 10 days of over 100% or under -50% returns were removed, as these were found not to reflect accurate historical prices).

Upon converting the times series for $N \approx 6,000$ tickers to their log-returns, $R_i(t)$, we use a lag date window of T days in order to incorporate additional longitudinal history for each prediction. In particular, we calculated the following d_{feats} metrics:

- The log returns of the sector corresponding to each stock, calculated using the weighted average stock prices in the sector
- The log returns of the sector corresponding to each stock, calculated using the arithmetic average stock prices in the sector
- The log market log returns, calculated using the weighted average of all the stocks in our dataset
- The log market log returns, calculated using the arithmetic average of all the stocks in our dataset

Our approach involved applying T days of log-return history to predict future stock performance in terms of returns, absolute returns, and volatility at several points in the future. For this

chapter, our choice of T was taken as 128 days. The network was optimized on the following d_{trgt} joint tasks:

- Predict log returns and absolute log returns 1, 7, 14, and 28 in the future for each reference date
- Predict volatility 7, 14 and 28 days in the future for each reference date

We trained our network based on several concurrent tasks of forecasting both returns and volatility measures (absolute returns and standard deviation), across a variety of time horizons. This is done in order to create more general representations, suited for a wider range of downstream tasks. Finally, we calculate the total loss as the sum of L_1 losses for each individual prediction.

Neural Network Architecture

We utilize the *Transformer encoder layer*, per Vaswani [87], as the core component of our neural network. This choice for the architecture is due to the recent successes it has had in model sequential natural language data, as well as superior scalability in GPU environments and a slight advantage in performance on our task over comparable recurrent architectures.

We utilize multi-head attention and position-wise feedforward networks inside each encoder layer per Vaswani, but do not use positional encodings. Instead, we concatenate the ordinal position of each return $r_i(t)$ to our features vector as an additional feature.

As per Vaswani, the inputs and outputs of each layer are of dimension d_{model} , and the output of each sub-layer is $\text{LayerNorm}(x + \text{Sublayer}(x))$. We also utilize a *modified Transformer encoder layer* as the first encoder layer, where residuals connections are not utilized, and the output of each sublayer is simply $\text{LayerNorm}(x)$.

The motivation behind the *modified Transformer encoder layer* is to use an intermediate representation in order to create some internal features within the neural network, aiding our model in learning features representative of the global statistics of the time series (e.g. mean and variance). Although the transformer network excels at finding interactions between log-returns of individual days (as each transformer layer looks at pairwise interaction between the tokens, or in our case, days), it can struggle to find global structure across all tokens / times, which we know to be important in financial markets. In order to help the network capture these long-term dependencies, we connect the output of the first transformer block to the embedding layer of the final encoder block, computing N_{features} additional metrics based on the learned representation corresponding to the final N_{tokens} tokens of the *Modified Transformer Layer* (this adds an additional $N_{\text{tokens}} * N_{\text{features}}$ neurons to embedding layer). In our experiments, we observe that this helped the network converge much faster and to a better local minimum, allowing it to learn global features around the overall time series more effectively.

Our final architecture uses n_{blocks} transformer layers (1 with additional connections to generate internal features and $(n_{\text{blocks}} - 1)$ standard layers per Vaswani) with n_{heads} attention heads each. We connect the time series representation from the first transformer layer to our first hidden layer, and perform aggregations on the intermediate representation to capture the global time-series structure. We use an embedding dimension of n_{embed} and encoder feed-forward network dimension $n_{\text{feedforward}}$, and a dropout of $P_{\text{attention_drop}}$ for the transformer attention layer, and $P_{\text{residual_drop}}$

for the transformer residual layer. This is all consistent with the Transformer network used in the original paper [87], with slightly lower network dimensionality, which we found was the appropriate size given the size of our training dataset, the dynamics of our problem.

We added two hidden layers after the transformer blocks, with dimensionality of d_{hidden} , and dropout of P_{hidden_drop} applied to each layer. We then use the output of the final feedforward layer to extract the embeddings of each particular stock for our downstream analysis tasks, simulating the use of such an embedding as a learned distance metric. These are added primarily to reduce the dimensionality of our final representation, as well as add some additional expressiveness to our network to further linearize our final representation.

We utilized RADAM [91] as our optimizer, with a learning rate of $r_{learning}$, in order to speed up convergence and simplify the hyper-parameter search for the optimal learning rate. We found that convergence was fast and did not benefit from a warm-up epoch when using the RADAM optimizer, and that the hyper-parameter search was significantly simplified as we were able to set a narrower and higher range of learning rates.

Finally, we experimented with our batch size and early stopping criteria. The hyperparameter choices for our final architecture are summarized in Exhibit 6.2:

Hyperparameter Description	Final Choice
Lag window of T days	128
Number of <i>feature time series</i> d_{feats}	7
Number of <i>target metrics</i> d_{trgt}	11
Embedding dimension n_{embed}	256
Number of transformer blocks n_{blocks}	6
Transformer block number of heads n_{heads}	16
Transformer block feed-forward network dimension $n_{feedforward}$	1024
Transformer block attention dropout $P_{attention_drop}$	0.1
Transformer block residual dropout $P_{residual_drop}$	0.1
N_{tokens} for feature computation	3
Internal features computed based on token embeddings	Standard deviation, L1, L2, and L3 norms ($N_{features} = 4$)
Hidden layer dimensions d_{hidden}	256, 128
Hidden layer dropout P_{hidden_drop}	0.25
Optimizer	RADAM
Learning rate $r_{learning}$	5×10^{-5}
Batch size	32
Evaluate every	200 steps
Early stopping	no improvement after 3 consecutive evaluations

Figure 6.2: Hyperparameter Choices

The best model found, based on the final hyperparameter choices described above, had an aggregate L_1 -loss of 1.447×10^{-2} , 19.25% lower compared to a statistically derived benchmark loss of 1.792×10^{-2} . Our approach for calculating the benchmark was estimating future returns and volatility using statistical measures based on historical averages during our lag window of T days, using statistical estimates for each task as follows.

This tells us that the network is able to learn a representation that successfully captures the structure of the time series, and is able to make predictions on future performance reasonably

well. Most of the gain over a statistical estimate comes from predicting volatilities and absolute log returns, with the predictions for 1, 7, and 28 day log returns being only marginally better than utilizing the mean. This is consistent with literature and the known difficulty of predicting returns; but, as we will see, does not prevent us from learning representations having properties that can be generally useful for a range of different tasks.

6.5 EVALUATING THE REPRESENTATIONS

We evaluate our representations by using them to supplement a common approach in quantitative finance of finding clusters of stocks using clustering techniques such as hierarchical clustering [118], which can then be used for downstream tasks such as hedging and portfolio construction.

We compare two approaches towards creating 10 hierarchical clusters on the first trading day of each month from January 2000, to August 2018; for consistency and simplicity, we sample the 1,000 largest stocks in each period by market capitalization for our experiment. The first approach being from quantitative finance:

1. Calculate the correlation matrix of the stocks in each period, again using our standard 128 day window
2. Utilize a modified correlation matrix (in our case, 2 - Corr) to create a pseudo-distance matrix D
3. Use the matrix D to calculate hierarchical cluster assignments for each period using the Ward criterion [124]

For comparison, we apply an alternative approach utilizing our learned representations:

1. Evaluate the embedding of each stock in each period by doing a forward pass through our trained network up to the final fully connected layer
2. Calculate a distance matrix in the embedding space using the standard Euclidean distance between each stock's embedding
3. Use the matrix 2 to calculate hierarchical cluster assignments for each period using the Ward criterion [124]

We note that the representations we create generates clusters that are more similar than correlation clusters in terms of their average risk-return profile in the 28 days following the first day of each period. This is, of course, natural given our training scheme, which optimizes these representations to solve tasks related to this outcome. Still, it is instructive that this structure is captured in the relative representations of each stock in the embedding space. These results are summarized below.

We first evaluate how separated stocks are along some of our key metrics across clusters. To calculate this, we:

1. Calculate the weighted average $\mu_{t,j}$ of each metrics y_j (log returns, volatility, etc.) across all clusters at time t
2. Calculate the intra-cluster volatilities at each time t as $\frac{V_1}{(V_1^2 - V_2)} \times \sum_{i \in t} w_i (y_{t,i} - \mu_{t,j}^*)^2$, weighted by the number of stocks in each cluster, and for each metric

Exhibit 6.3 shows the weighted intra-cluster volatility of the mean stock volatilities, as a measure of how well each clustering approach is able to group stocks that will have similar volatilities in the future. As expected, our approach creates clusters that are consistently better separated in terms of future volatility:

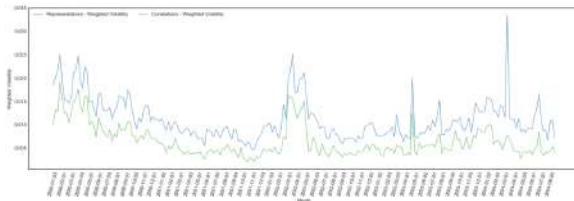


Figure 6.3: Intra-Cluster Volatility: Stock Volatilities

This is not the case for future returns, as can be seen in Exhibit 6.4. This is also to be expected, as predicting future returns based on time series data is known to be ineffective due to rapid repricing. Nevertheless, we believe adding such tasks can induce the learned embeddings to be more useful, even when the performance on a subset of training tasks is not useful when taken in isolation (e.g., as is the case when training word embeddings).



Figure 6.4: Intra-Cluster Volatility: Stock Returns

6.6 SIGNIFICANCE

We show that modern machine learning approaches can be used to learn distance metrics at scale, and offer some examples on how such an approach may differ from using historical correlations. Learning tailored representations based on specific tasks is an approach that has shown tremendous value in domains of natural language processing and image processing, yet has not been utilized in finance. Financial time series data, however, is comparable with those two domains through its abundance of training data to learn rich representations. It is perhaps set to benefit even more from learning such representations due to the prevalence of distribution drift, and the abundance of varying financial objectives that warrant learning more diverse representations of the same data.

The flexibility offered up can help create general-purpose embeddings that serve the financial needs of creating stable, robust portfolios that account for overall market conditions. It is natural to engineer tasks for pre-training that will create desirable properties in representations of time series data, such as stability of representations over time and incorporation of additional market data (including non-time series data such as financial news and fundamental data). All of these advantages make deep neural networks a highly promising approach towards creating new ways

to process market data, creating rich representations suitable for improving existing financial strategies and creating new ones.

6.7 ADDITIONAL RESEARCH AVENUES

Additional research avenues this work motivates include further improvements for stability, capturing additional market information, and learning representations that best capture the behavior of the market jointly.

Creating time series representations that control for stability of representations over time is another promising area of research, which can help create systems of hedging and portfolio construction that are stable over time (and hence more testable), helping reduce some of the risk associated with potentially unknown failure points created by highly nonlinear techniques such as deep learning.

Although such approaches can be criticized for their “black-box” nature, we argue that the downside compared to the traditional correlation-driven approach is low. An additional metric needs to be interpreted, but the portfolio construction and hedging work can be performed as is. The additional flexibility neural networks offer of tailoring the representations to specific pre-training tasks, and our ability to inject domain expertise into how these tasks are specified, will over time lead to wider adoption of such systems.

LEARNING REPRESENTATIONS OF INTERRELATED FINANCIAL TIME-SERIES

This chapter introduces a new financial time-series representation model called **RIFT** (**R**epresentations of **I**nterrelated **F**inancial **T**ime-series). RIFT combines a novel pre-training task and neural network architecture to create generalized representations of multiple financial time-series inputs. The network uses a Siamese architecture to predict pair-wise future correlations of securities; the encoder can then be used to create representations of individual securities for downstream tasks. Similar to successful applications of transfer learning in other domains, the authors test the representations on several downstream tasks common in quantitative finance, including dimensionality reduction, portfolio optimization, and portfolio reconstruction. In particular, the chapter introduces Neural-HRP, an improvement on the Hierarchical Risk Parity algorithm, the current state-of-the-art for portfolio optimization, and shows promising results across a variety of assessment criteria, including a 6.0% relative improvement in annualized returns, and a 5.6% relative improvement in the Sharpe ratio.

7.1 INTRODUCTION

This chapter is motivated by the incredible effectiveness of deep learning models in the domains of Computer Vision (CV) and Natural Language Processing (NLP), largely enabled by the ability of deep learning models to generalize across multiple tasks. This approach, commonly known as *transfer learning*, leverages the learned *representations* or *embeddings* of data in a late-stage layer of a deep neural network [40]. Deep neural networks typically culminate in a linear, task-specific output layer; in contrast, these penultimate representations preserve complex properties which tend to generalize well to adjacent tasks within a data domain [88].

There is a rich history of *fine-tuning* large pre-trained models for a range of tasks in both CV [125, 126] and NLP [84, 127, 108]. The main prerequisites for successful transfer learning are an abundance of relevant data, and a varied set of supervised learning tasks that would benefit from fine-tuning a large pre-trained model. Quantitative finance, in this context, offers many opportunities: there is an abundance of data and a myriad of applications, but transfer learning is yet to be widely applied. This work describes an approach for learning general representations of financial time-series with a correlation pre-training task, and demonstrates the effectiveness of the model by leveraging the representations across several tasks common in quantitative finance.

The Literature Review section outlines a brief history of transfer learning in other domains, justifies the use of correlation as the pre-training task due to its centrality in the field of

quantitative finance, and summarizes recent works applying deep-learning models to financial time series. The Pre-Training of Financial Time-series Networks Section describes the pre-training tasks, and the Network Architecture section provides an overview of the model architecture. Finally, the Applications section demonstrates the effectiveness of the generalized model.

7.2 REPRESENTATIONS OF FINANCIAL TIME-SERIES DATA LITERATURE REVIEW

Deep Learning and Transfer Learning

Transfer learning approaches using large deep learning models have shined in domains with an abundance of data. Computer vision models have leveraged the ImageNet [92] dataset to scale convolutional networks to massive sizes and super-human performance [116], and applying these models to other vision models through transfer learning [125, 126, 88]. More recently, more data domains such as protein sequences [128, 129], are leveraging similar training tasks to enhance the amount of effective labeled data available and improve the ability of models to generalize.

The domain of language data took a longer period of time to develop large pre-trained models, in part due to the lack of a large commonly available labeled dataset like ImageNet. Language translation was the domain in which large deep learning models initially had the most success, due to the abundance of text already translated to multiple language being used as training data, allowing for the creation of large neural network models for machine translation [130, 87]. The BERT model [84] served as the breakthrough for pre-training general language models, with a key innovation coming from leveraging massive amounts of unlabeled text data readily available for pre-training. Many of the subsequent improvements in terms of performance and general applicability of deep learning models for NLP were focused on scaling up the size of the models and the generality of pre-training tasks [131, 108, 132]. In particular, recent advances in the usability of NLP systems such as Chat GPT [133] further showcase the importance of choices of pre-training and fine-tuning tasks.

However, there has been remarkably little research into creating foundational feature extraction models for finance such as BERT [84] or GPT [108] for NLP, or what the right choice of pre-training tasks for such a foundational model should be; related works in the domain of finance are explored in the section below.

Finance

Portfolio optimization is perhaps the most important and widely studied application of quantitative finance techniques. Portfolio optimization theory dates back to 1952 when Harry Markowitz laid the foundation for the modern portfolio theory (MPT) with his seminal paper “Portfolio Selection” [2]. Branching off from Markowitz’s mean-variance optimization (MVO) framework, many mathematical frameworks have been devised in an attempt to enhance portfolio performance based on the theoretical understanding of the correlation structure of markets. For example, the Black-Litterman model [93] uses a Bayesian approach to estimate consensus expectations of returns and the covariance structure of markets, while the frequently used Risk Parity [134]

framework abandons the use of returns altogether due in part to the difficulty of forecasting them.

More recently, a more empirical approach at the intersection of machine learning and portfolio theory began to emerge, with the Hierarchical Risk Parity (HRP) algorithm, first introduced by Lopez de Prado in 2016 [135]. The HRP algorithm is an improvement on the Risk Parity framework, and showed state-of-the-art performance for portfolio optimization and is being widely adopted in the industry since. The algorithm can be divided into three main steps - clustering, quasi-diagonalization, and recursive bisection - is a hierarchical implementation of an inverse-variance allocation with asset weights calculated between clusters of correlated asset returns. In particular the clustering step relies on a notion of distance and uses correlation distance between securities to form hierarchical groups. In this paper, the authors build upon the HRP framework by introducing a new distance metric within the clustering step: the Euclidean distance between asset embedding vectors generated from the financial time-series encoder network, as shown in the Neural Hierarchical Risk Parity section.

Distance metrics and “fixed” representations of financial time series have frequently been studied in financial setting. A classical and common approach to embedding financial time-series is the use of the Fourier transform for analyzing dynamic signals from stock markets [136]. The key downside to approaches like the Fourier Transform is that the embedding is not a learned transformation and does not benefit from the abundance of training data within historical financial time-series data. Another approach common in industry is extracting simple representations of snapshots (in time) of financial time-series data using techniques such as Principal Component Analysis (PCA) for dimensionality reduction of multiple time-series, for example to improve the robustness of portfolio optimization [137], or in highly specialized applications such as extracting yield curve representations [138]. Similarly, PCA and other unsupervised dimensionality reduction approaches do not benefit from the abundance of training data within the financial domain, as well as potentially lacking in their ability to embed complex data compared to more expressive deep neural network models. This paper improves on this prior work by proposing a more general financial time-series model suitable for fine-tuning across a variety of quantitative finance tasks.

Correlation Prediction and Time-Series Architectures

RIFT leverages correlation prediction as its pre-training task, as further described in the Pre-Training of Financial Time-series Networks section. Correlation and covariance are arguably the most prevalent statistical measures used in modern portfolio theory to quantify investment risks, and making reliable forecasts of them is of great interest to many financial practitioners. Prior works in predicting the correlation structure among securities include the use of time-series methods and other techniques, such as RiskMetrics [139] and Ledoit-Wolf shrinkage [140]. However all of these framework are restricted by the theoretical assumptions they make, and also lack the expressiveness compared to deep neural nets to be able to capture complex structures. Classical time-series forecasting relies on autoregressive models such as generalized autoregressive conditional heteroskedasticity (GARCH) [141], but they are mainly applicable for univariate forecasting problems, with limited ability to share information across multiple forecasts.

More recently, machine learning algorithms such as support vector machine (SVM) [142], convolutional long short-term memory (ConvLSTM) [143], Temporal Convolutional Networks

(TCN's) [144, 145] have been popularized for working with financial time-series. Some of their advantages are in being data-driven and self-adaptive, requiring much fewer assumptions about the underlying data, and leveraging historical training data more effectively. They have demonstrated superiority over some conventional statistical approaches by making more accurate future covariance forecasts. The primary focus of past machine learning works in finance has been in optimizing the performance for predicting the prices of individual securities. Instead, the focus in this chapter is on learning general representations, with a foundational pre-training task and multiple varied downstream tasks. While some previous works have studied representation learning for financial time series [145, 146, 147], these applications have typically been focused on learned representations being used for a narrow set of tasks highly aligned to the pre-training tasks. The use of deep reinforcement learning for directly learning investment strategies and optimal portfolios has been extensive [148, 149], but similar to other deep learning work in this space, representation learning and cross-applicability to diverse tasks in finance remains under-explored. The encoder block of the network, described in detail in the Network Architecture section creates intermediate representations of individual time-series that are then concatenated for the final correlation prediction task. As shown in the Applications section, this does indeed generate improved performance across a variety of downstream tasks.

With the increasing availability of computing power and the abundance of data, deep learning techniques have generated significant interest in the financial time-series domain, but with mixed results to date [150]. Time-series being just a special type of sequence data, three kinds of sequential deep neural architectures, and their variants, are mainly used for time-series forecasting: recurrent neural network (RNN) [151], Transformer [87], and Temporal Convolutional Network (TCN) [144]. RNN's capture temporal dependency by storing past information in *memory cells*, which suffers from error accumulation and vanishing/exploding gradient problems. Transformers have naturally replaced RNN's in many sequential modelling tasks due the effectiveness and scalability of self-attention mechanism, and have gained popularity for time-series forecasting tasks [152]. TCN's also demonstrate comparable performance, and can often to do with a smaller computational costs due in part to the dilated convolution structure [144] employed by these networks. The authors experiment with all of these approaches for the network's encoder, with the final architecture summarized in the Network Architecture section.

7.3 PRE-TRAINING OF FINANCIAL TIME-SERIES NETWORKS

Designing a deep-learning approach for extracting useful general representation from a dataset relies on two key design decisions: the choice of pre-training task, and the choice of neural network architecture that embeds the input data and produces the desired representation. All the experiments for model pre-training were conducted using a total of under 1,000 hours of NVIDIA V100 GPU computing time, with the final architecture selected after conducting approximately 200 rounds of experiments at a maximum 5 hours of GPU time per experiment.

Table 7.1: List of final hyperparameter choices of the model

HYPERPARAMETER DESCRIPTION	FINAL CHOICE
LAG WINDOW OF seq_len TRADING DAYS	378
NUMBER OF FEATURE TIME-SERIES d_{input}	26×2
NUMBER OF TARGET METRICS d_{target}	3
EMBEDDING DIMENSION d_{embed}	256
MARKET EMBEDDING DIMENSION d_{market_embed}	64
INDUSTRY CLASSIFICATION EMBEDDING DIMENSION d_{ind_embed}	32
NUMBER OF ENCODER LAYERS $n_{encoder_layer}$	3
ENCODER EMBEDDING DIMENSION $d_{encoder}$	600
NUMBER OF TCN CHANNELS $n_{channel}$	[600, 600]
TCN KERNEL SIZE d_{kernel}	3
TCN DROPOUT RATE P_{TCN}	0.1
NUMBER OF (SUBNETWORK) TRANSFORMER BLOCKS n_{block}	1
NUMBER OF (SUBNETWORK) TRANSFORMER HEADS n_{head}	16
TRANSFORMER (ATTENTION & ACTIVATION) DROPOUT RATE P_{trans}	0.1
ENCODER HIDDEN LAYER DIMENSION d_{hidden}	[512, 256]
POST-ENCODER HIDDEN LAYER DIMENSION d_{post_hidden}	512
HIDDEN LAYER DROPOUT RATE P_{hidden}	0.25
LOSS FUNCTION	L_1 LOSS
OPTIMIZER	RADAM [91]
LEARNING RATE γ	5×10^{-4}
BATCH SIZE B	64
EVALUATION FREQUENCY	250 STEPS
EVALUATIONS WITHOUT IMPROVEMENT FOR EARLY STOPPING	3

Pre-Training Task

The choice of pre-training task is correlation prediction, due to the importance of the correlation structure of financial markets across many common modelling tasks performed in quantitative finance. The main dataset leveraged is the equity price dataset, which was obtained from Compustat [153], starting in 1998 and ending with mid-2020 (since S&P index data is no longer available readily accessible past this date).

In order to encourage the network to learn the correlation structure between securities, the authors leverage a *Siamese* network architecture [154]. *Siamese* networks are used for pair-wise learning tasks, such as distance learning, where an identical network (shared architecture and weights) is used to pair-wise encode two sets of input data. Recently, similar pair-wise pre-training tasks have proven extremely effective in the domain of NLP in scenarios when the amount of available labelled data is low [155]. In this study, a learned encoder f is used in order to create pair-wise representation of two time-series (s_1, s_2) , as $f(s_1)$ and $f(s_2)$, using the combination of embeddings $f(s_1)$ and $f(s_2)$ to make a prediction about the future correlation of input time-series s_1 and s_2 . The detailed pre-training approach is as follows:

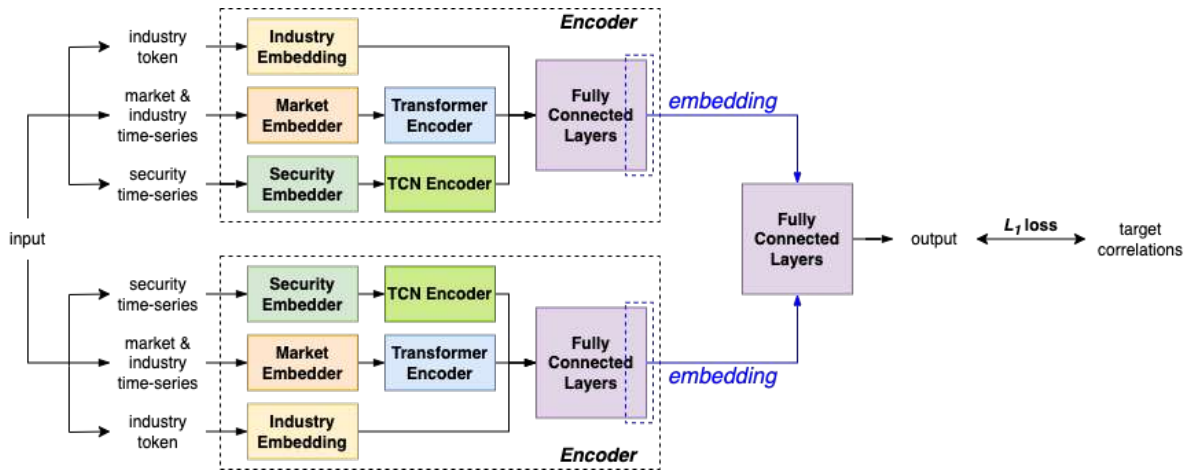


Figure 7.1: RIFT network architecture

1. Pick an *anchor date* within the training time period, and sample a pair of securities at the given date ($t = 0$)
2. Pick appropriately lagged time-series (s_1, s_2) to use as inputs into the neural network, with sequence length of seq_len steps each
3. Embed each of s_1 and s_2 using the learned encoder f
4. Combine the embeddings of $f(s_1)$ and $f(s_2)$ to predict several target variables (t_1, \dots, t_k) jointly

The authors make the following choices for the purposes of the experiments outlined below:

- A sequence length seq_len of 378 trading days (1.5 years) as of the anchor date for s_1 and s_2 ; this time period is chosen to incorporate more information than a full trading year, while retaining sufficient information in the training and validation period
- Training data is sampled from the period starting April 1998 and ending December 2014, and the validation period is from January 2015 to June 2020
- Three pre-training tasks are used: correlation between the two sampled securities 1, 3, and 6 weeks into the future
- The L_1 loss function is used to optimize the neural network

7.4 NETWORK ARCHITECTURE

Initial Embedding Layer

The general architecture of sequential encoder blocks, for example as used in the Transformer architecture [87], is symmetric in the sense that the input and output tensor shapes are identical, with input and output tensors for an encoder block having dimensionality $B \times seq_len \times d_{encoder}$, where $d_{encoder}$ is the embedding dimension of the encoder block, and B is the batch size. In

order to leverage a similar architecture for stacking encoder blocks and creating larger, more expressive networks, the authors first create intermediate representations of input time-series (s_1, s_2). These representations can be thought of as having the encoder embedding dimension $d_{encoder}$ corresponding to the word embedding dimensional of a text segment for a Transformer network.

In order to create this initial representation, the authors utilize a two-step process. First, the authors calculate several simple transformations of the input time-series s_1 and s_2 , utilizing simple lags, moving averages and moving standard deviations at different time steps. These are applied both to log-returns as well as original normalized prices of the input time-series, creating tensors of dimensions $B \times seq_len \times input_dim$. These are then encoded using an embedding layer to create a higher-dimensional representation of shape $B \times seq_len \times d_{encoder}$. In the authors' final experiments, 25 input transforms are used, corresponding to a $input_dim$ value of 52 (2 input time-series with 25 transforms applied to each), and an $d_{encoder}$ corresponding to the channel size of 600 for the TCN models that yielded the best results, seen in Exhibit 7.1.

Encoder Blocks

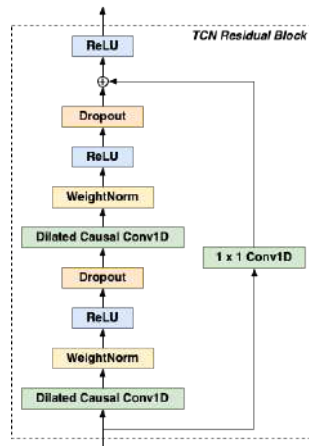
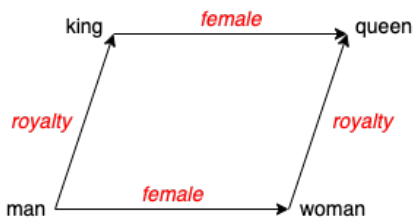


Figure 7.2: TCN block



(a) Word embedding



(b) Financial embedding

Figure 7.3: Hidden relationships like (a) gender and royalty or (b) asset composition can be found in linear analogies of embedding vectors.

Table 7.2: A summary statistics of HRP and Neural-HRP portfolio cumulative performance from January 2015 to June 2020. We use the S&P 500 index as the market benchmark.

PORTFOLIO	ANNUALIZED RETURN	VOLATILITY	SHARPE RATIO	INFORMATION RATIO	TURNOVER
HRP	5.26%	14.1%	0.107	-0.0140	28.0%
NEURAL-HRP	5.58%	14.2%	0.113	0.0137	22.4%
SPY	5.39%	14.3%	0.108	-	-

The encoder blocks employ TCN residual blocks, illustrated in Exhibit 7.2, to convert the transformed input time-series into dense embedding vectors. The final architecture stacks 3 TCN residual blocks in series, all of which have identical input and output dimensions of 600, kernel size of 3, and dilation factor of 2.

The authors append three additional data inputs to allow the network to learn the interactions between input time series and overall markets conditions & industry dynamics: time-series of the returns of the overall market, the industry corresponding to each input time series, and an industry token with a learned embedding. Market and industry time-series of seq_len time steps are concatenated together into one tensor, and then concatenated together with the same “feature engineering” transforms as described in the Initial Embedding Layer section. A single Transformer encoder block with 16 attention heads is used to generate a representation from the transformed tensor, as Transformers performed marginally better than TCN’s for the smaller sub-networks in some experiments. The industry token, recorded in North American Industry Classification System (NAICS) code system, is embedded using a linear embedding layer. All three transformed data and the input time-series are concatenated together to undergo two linear layers of dimension 512 and 256 respectively. The output of the last fully connected layer in the encoder block, indicated by blue arrow in Exhibit 7.1, is the intended representation of the time-series, which can be extracted upon completion of the pre-training task.

Two identical encoder blocks, each taking in input time-series s_1 and s_2 , are coupled by a hidden linear layer of dimension 512 and an output layer of dimension 3, aligning with the number of the pre-training target tasks. Finally the output is rescaled between -1 and 1 under \tanh activation function which is then compared with the target correlations using L_1 loss. The final model achieves an average L_1 loss of 0.24699 on the three pre-training targets, with details described in the appendix. The overall architecture is summarized in Exhibit 7.1.

7.5 APPLICATIONS

The following section demonstrates applications of RIFT embeddings for general dimensionality reduction, portfolio optimization, and asset replication; these are some of the most frequent and important tasks performed in quantitative finance industry. Python library *PyPortfOpt* [156] is mainly employed to implement HRP and its variant introduced in this chapter.

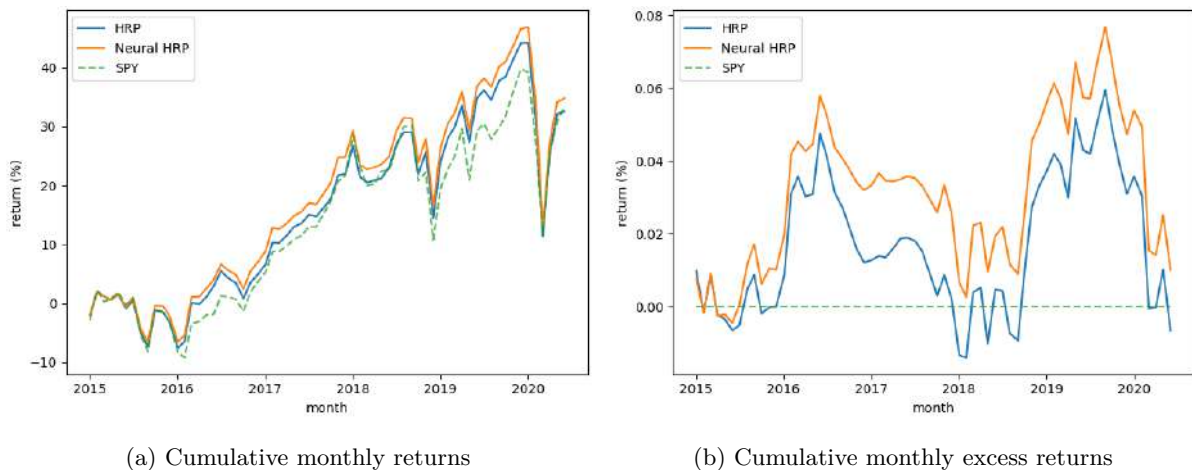


Figure 7.4: HRP and Neural-HRP portfolio cumulative performance from January 2015 to June 2020. We use the S&P 500 index as the market benchmark.

Representations of Financial Time-Series and Portfolios

Many distributed word representation models, such as *word2vec* [114] and *GloVe* [157], exhibit an unintended "side-effect" of compositionality, i.e., the property that meaning can be composed by adding or subtracting the embedding vectors. The most famous example being *king* – *man* + *woman* = *queen* as shown in Exhibit 7.3a, it illustrates how the models are capable of capturing hidden semantic relationships through relative positioning in the embedding space. Embeddings of financial time-series generated by RIFT display a similar feature where expected relationships among securities can be inferred from simple vector arithmetic; namely, an embedding of a stock index (e.g. DJIA, a subset of 30 S&P 500 constituents) can be approximated as a weighted sum of its constituents' embedding vectors with high accuracy as shown in Exhibit 7.3b. Note that in this case, the input time series used are a multi-dimensional input of 52 transformed time series, coupled with industry and market information as described in the Initial Embedding Layer section; the simplified vector representations still behave in intuitive ways, reminiscent of properties held by word embedding vectors.

This property is extremely valuable in the field of finance as an alternative to commonly used dimensionality reduction approaches such as PCA [137]. When dealing with multiple time-series embeddings, the property above allows for simply mean-pooling of multiple time-series to create representations of time-series combinations / portfolios. Mean-pooling has been incredibly effective when dealing with natural sequences, from simpler approaches like Deep Averaging Networks [158] and mean-pooling of token embeddings commonly used for classification tasks in language models [159]. Similarly, these results allows for working with a fixed dimensionality regardless of the number of time-series inputs being studied, by simply mean-pooling the embeddings of multiple time-series across the embedding and time-step dimensions, taking an input of any N time series enriched with additional information of shape $N \times input_dim \times seq_len$, and creating representations of fixed dimensionality $d_{embed} \times seq_len$.

Table 7.3: A summary statistics of the S&P 500 index replication using the largest $N \in \{5, 10, 15\}$ asset constituents under rolling LR and neural rolling LR respectively from January 2015 to June 2020.

	REPLICATION	TRACKING ERROR	CORRELATION	DAILY VOLATILITY
$N = 5$	ROLLING LR	0.631%	0.887	1.36%
	NEURAL ROLLING LR	0.603%	0.893	1.34%
$N = 10$	ROLLING LR	0.387%	0.950	1.24%
	NEURAL ROLLING LR	0.358%	0.955	1.20%
$N = 15$	ROLLING LR	0.298%	0.969	1.21%
	NEURAL ROLLING LR	0.294%	0.969	1.19%
	S&P 500	-	-	1.17%

Neural Hierarchical Risk Parity

The original HRP algorithm that Lopez de Prado presented in his paper [135] uses correlation-based distance measure to form hierarchical clusters. Formally, given N assets and their corresponding time-series s_1, \dots, s_N , the distance between asset i and j is defined as: $d_{ij} = \sqrt{\frac{1}{2}(1 - \rho_{ij})}$, where ρ_{ij} is the Pearson correlation between s_i and s_j . Although such formulation brings the mathematical benefit of establishing a proper metric space, it only relies on historical statistics and is not forward looking, leaving room for improvement of out-of-sample performance. The authors propose an alternative distance metric for hierarchical clustering using the time-series embeddings, which can incorporate future risk estimates into portfolio construction. For any two assets i and j and their corresponding time-series s_i and s_j , the distance between them is defined as: $d_{ij} = \|f(s_i) - f(s_j)\|$, the Euclidean distance between the embedding vectors of the two assets.

The out-of-sample performance of both approaches are compared under the following backtesting conditions: portfolios are rebalanced on the first trading day of every month from January 2015 to June 2020 (5.5 years); and at each rebalancing date, all securities that are included in the S&P 500 index are considered as the asset universe. The two following modifications of the HRP algorithm suggested by more recent works in quantitative finance are incorporated into the backtest:

- Ward criterion, a general agglomerative hierarchical clustering procedure seeking to minimize the total within-cluster variance, has shown to significantly outperform single-linkage method when applied to HRP [160].
- Instead of sample covariance matrix, exponentially-weighted covariance matrix is used for assigning inverse-variance weights to better address the heteroskedasticity of asset returns by giving more emphasis to recent observations [161]. The decay factor is set to $\alpha = \frac{2}{seq_len+1} = \frac{2}{379}$.

Neural-HRP outperforms the correlation-based HRP in all key metrics - annualized returns, total turnover, information ratio, and Sharpe ratio - summarized in Exhibit 7.2. The lower average turnover indicates greater stability of the algorithms, as this metric is defined as the sum of absolute difference between asset weights in the composition of the portfolio at each rebalancing date. The result above is based on a single experiment, a purposeful limitations imposed in order to avoid the dangers of P-Hacking common within quantitative finance [44]. These results indicate that while the embeddings effectively capture the general correlation structure, they also provide a slight consistent edge by incorporating information about future risks.

Asset Replication

Asset replication is an important topic for institutional investors seeking returns of assets that cannot be directly invested, such as market indices, or are not desirable to directly invest, such as hedge funds with high management fees. A common approach to tackle to this problem is to construct “linear clones” from individual assets or factors using a rolling linear regression (rolling LR) [162]. Let R_t and F_t be respectively the returns of the asset to be replicated and the returns of N individual factors at time t . Then the rolling LR model assumes $R_t = F_t w_t + \epsilon_t$ where w_t is the vector of factor weights such that $\sum_{i=1}^N w_{t_i} = 1$ and ϵ_t is the error between the asset and its linear replication returns. Note that the intercept term is commonly dropped in the regression under the assumption that the factors can fully recover the target asset, and to ensure the solution is investable.

The return vectors R_t and F_t can be replaced by their corresponding embedding vectors $f(R_t)$ and $f(F_t)$, reformulating the regression problem as $f(R_t) = f(F_t)w_t + \epsilon_t$. To demonstrate this, two approaches are applied to the task of replicating the S&P 500 index with a subset of its asset constituents. Specifically, the largest $N \in \{5, 10, 15\}$ securities of S&P 500 by market capitalization are used to track the index; both replicating portfolios are rebalanced on the first trading day of every month from January 2015 to June 2020 (5.5 years).

To address the numerical instability of inverting a matrix with highly correlated predictor variables - asset returns or embedding vectors in this case - the authors add a ridge regularization term $\lambda \|w_t\|^2$ to the regression model, a common practice in asset replication task [163]. The ridge regularization coefficient λ is selected using 10-fold cross-validation for each value of N . The optimization problem, at time t , can be formulated as follows:

$$\min_{w_t} \|X_t w_t - y_t\|^2 + \lambda \|w_t\|^2 \quad \text{s.t.} \quad \sum_{i=1}^N w_{t_i} = 1 \quad (7.1)$$

where $X_t = F_t, y_t = R_t$ for the conventional method and $X_t = f(F_t), y_t = f(R_t)$ for the newly proposed approach. As seen in Exhibit 7.3, neural rolling LR demonstrates superiority over the traditional rolling LR replication technique in all key metrics - tracking error, correlation between the replication and the benchmark, and daily volatility - for all examined values of $N \in \{5, 10, 15\}$ but show diminishing improvement as N increases. These results indicate that forward-looking nature of RIFT representations provides additional insights for asset replication tasks, with the highest benefit associated with replicating a portfolio using a low number of assets.

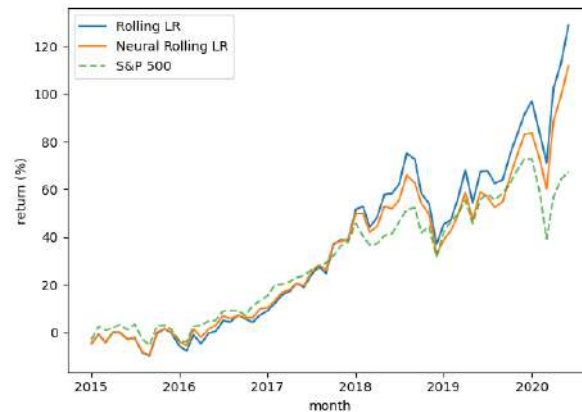


Figure 7.5: S&P 500 index replication cumulative performance for $N = 5$ from January 2015 to June 2020. We use the S&P 500 index as the market benchmark.

7.6 ADDITIONAL RESEARCH AVENUES

In conclusion, the authors demonstrate a robust scheme and network architecture for pre-training of deep neural networks that create useful representations of financial time-series data which generalize well across a variety of tasks in finance. This is the first work of its kind in the quantitative finance domain. Working with embeddings of financial time-series can simplify or improve many common tasks in finance, such as dimensionality reduction or prediction tasks, and portfolio optimization more broadly.

Transfer learning is an incredibly under-explored area in the domain quantitative finance, and many fruitful research directions can be explored to build on this work. The deep neural network models used to embed financial time-series can potentially be scaled up to be significantly larger and more expressive, especially by expanding the training data set by introducing richer data across multiple asset classes and geographies, to model more complex correlation structures across asset belonging to different asset classes. In addition, other financial applications not explored in this chapter, such as hedging, options pricing, and general predictive tasks can potentially be improved by leveraging representation of the input time-series typically used for such tasks.

THE TOPOLOGY OF LEARNED RIFT REPRESENTATIONS

This chapter considers how our project combines ideas from modern natural language processing (and more broadly sequence modeling) in deep learning, as well as topological ideas and topological data analysis, in order to study the structure of markets through examining learned representations of financial time-series data. This is studied via a two-stage approach, first by designing and training a network to create neural representations of financial time-series data, and then using topological data analysis to compare the structure of the raw time series to their learned representations.

8.1 TRANSFORMER MODELS FOR FINANCIAL TIME SERIES DATA

We begin our discussion by starting with a brief survey of deep learning approaches for sequence modeling.

Historically, the key evolution in sequence modeling was achieved primarily in the field of Natural Language Processing (NLP), and specifically machine translation. Recurrent networks have long been used for sequence modeling. The seminal work by Sutskever et. al. [130] introduced the idea of using recurrent networks with the *encoder-decoder* architectures in order to perform machine translation, which was a start of a series of important technical evolutions of deep learning models for sequences.

Key to these was the attention mechanism, was first introduced to address this long-term dependency issue encountered by recurrent networks [122]. Figure 8.2 shows how a recurrent network used for machine translation is augmented with the *attention* mechanism. The diagram depicts the usage of the attention mechanism in an *encoder-decoder* architecture used for a translation task. Here, the encoder maps an input sequenced into an intermediate representation the decoder then uses to output a new sequence, in this sequence, in this case in a different language. In the diagram, s_1, \dots, s_t indicate hidden states or the “intermediate” representations of the decoder at sequence index t , while the h'_t s indicate the hidden states of the encoder. The attention weights are learned, and allow for the hidden state representations to be to be *pooled*, or averaged, in order to allow the decoder to use information that may be far removed in the sequence.

The key innovation for sequence modeling in deep learning came with the seminal work of Vaswani et. al. [87], which demonstrated that recurrence can be abandoned altogether in order of stacking *Transformer* layers which combine self-attention with fully-connected layers, as depicted in Figure 8.3. The transformer can be thought of as a standard multi-layer perceptron (MLP) made up fully-connected layers operating individually on the tokens within the sequence, but

with intermediate averaging or “pooling” layers performed via self-attention. This allows for contextual representations of the input tokens, enriched by their relationship to other tokens in the sequence. Figure 8.4 illustrates this mechanism, whereby the representations of the tokens “Thinking” and “Machines” are pooled using weights generated by the softmax function applied to learned attention weights. Below is an overview of how this model architecture was adopted to modelling financial time series.

In this chapter, the transformer architecture is applied to financial time-series data. We consider tensors of financial time series data of dimension $l \times d$, where l is the sequence lengths (corresponding to the number of tokens for a transformer model applied to text), and d is the number of input time series (e.g. log returns, windowed variance, raw prices, etc.). We add an initial linear embedding layer with a ReLU activation to embed the input tensor into a higher $l \times d_{model}$ dimension due to the symmetrical nature of the transformer encoder.

A neural network our network on a correlation prediction task, as described in Chapter 7: given two input time series x_1 and x_2 corresponding to two securities, we predict the future correlation of these securities 5, 10, and 21 days into the future, as well as predict the future returns and volatilities of the individual securities. We do this using a *Siamese* network structures, first introduced in [154] by Bromley and LeCun, whereby the same encoder weights are used to create embeddings $f(x_1)$ and $f(x_2)$, which are subsequently concatenated prior to the full network’s output layer. This approach was chosen in order to generate embeddings that can be useful for *transfer learning*, similar to how Transformers are used to generate general-purpose embeddings for text sequences [84].

Our architecture uses n_{blocks} transformer layers with n_{heads} attention heads each. Inner residual connections are removed from the first transformer block in order to retain the full information in the first layer encodings, which are aggregated to capture the global time series structure. We use an embedding dimension of n_{embed} and encoder feed-forward network dimension $n_{feedforward}$, and a dropout of $P_{attention_drop}$ for the transformer attention layer, and $P_{residual_drop}$ for the transformer residual layer. This is all consistent with the Transformer network used in the original paper [87], with slightly lower network dimensionality, which was found to be the appropriate size through hyperparameter experimentation and given the size of our training dataset and the dynamics of our problem. The full hyperparameter choices are summarized in table 8.1:

8.2 TOPOLOGY BACKGROUND

In order to describe the topological complexity of a topological space, we define several background concepts from the field of algebraic topology. This section gives an introduction of the relevant topological background, all notations in line with [164].

Simplicial Complex and Topological Complexity

A k -dimensional simplex, or k -simplex in \mathbb{R}^d , is the convex hull of $k + 1$ affinely independent points $v_0, \dots, v_k \in \mathbb{R}^d$. For example, a 0-simplex is a point, a 1-simplex is a line segment and a 2-simplex is a triangle etc. A k -simplex is represented by listing the set of its $k + 1$ vertices and denoted by $\sigma = [v_0, \dots, v_k]$. The faces of a k -simplex are simplices of dimensions 0 to $k - 1$

formed by convex hulls of proper subsets of its vertex set v_0, \dots, v_k . For example, the faces of a line segment are its end points, which are 0-simplices; the faces of a triangle are its three sides, which are 1-simplices, and its three vertices, which are 0-simplices. An *abstract simplicial complex* is a list of simplices $K = [\sigma_1, \sigma_2, \dots, \sigma_n]$ with the property that if $\tau \subset \sigma \in K$, then $\tau \in K$.

Let $K^{(k)}$ denote the set of all k -simplices of K . The k -chain group is the free \mathbb{F}_2 -vector space $C_k(K) = \text{span}_{\mathbb{F}_2}(K^{(k)})$, whose basis elements are the simplices in $K^{(k)}$. We define linear map $\partial_k : C_k(K) \rightarrow C_{k-1}(K)$ by

$$\partial_k \sigma := \sum_{j=0}^k [v_0, \dots, \hat{v}_j, \dots, v_k]$$

for $\sigma \in K^{(k)}$. Note signs are omitted because computations are over \mathbb{F}_2 . One can check that

$$\partial_k \circ \partial_{k+1} = 0$$

for $k = 0, 1, \dots, d$. Then $\{\partial_k : k = 0, \dots, d+1\}$ form a chain complex. The k -th homology of the simplicial complex K is $H_k(K) = \ker(\partial_k) / \text{im}(\partial_{k+1})$. The k -th Betti's number of K , defined as the dimension of an \mathbb{F}_2 vector space is given as

$$\beta_k(K) = \dim(H_k(K)) = \text{nullity}(\partial_k) - \text{rank}(\partial_{k+1})$$

for $k = 0, 1, \dots, d$. Note ∂_k is a matrix of size $|K^{(k-1)}| \times |K^{(k)}|$. Therefore $\beta_k(K)$ can be computed with linear algebra on \mathbb{F}_2 .

For a given topological space $M \subset \mathbb{R}^d$, the *topological complexity* of M is defined to be

$$\beta(M) := \sum_{k=0}^d \beta_k(M) \tag{8.1}$$

Point Cloud Data and Vietoris-Rips Complex

Given a point cloud data set $X \subset \mathbb{R}^d$, we can build an abstract simplicial complex with vertex set X in the following manner. Consider metric δ on \mathbb{R}^d . The *Vietoris-Rips complex* at scale $\epsilon \geq 0$ on X is denoted by $\text{VR}_\epsilon(X)$ and defined by

$$\text{VR}_\epsilon(X) := \{[x_0, \dots, x_k] : \delta(x_i, x_j) \leq 2\epsilon, x_0, \dots, x_k \in X, k = 0, 1, \dots, n\}$$

When $\epsilon = 0$, $\text{VR}_0(X) = \{|x| : x \in X\}$ is a collection of 0-dimensional simplices with $\beta_0 = |X|$ and all other Betti numbers zero. As ϵ increases, more and more edges are introduced to form higher and higher-dimensional simplices and $\text{VR}_\epsilon(X)$ has richer topology. As $\epsilon \rightarrow \infty$, all points in X becomes a vertex of a single simplex which is contractible as a topological space.

Computing the homology groups of $\text{VR}_\epsilon(X)$ for the full range of ϵ would give us the *persistent barcodes*.

8.3 EXPERIMENTAL APPROACH AND RESULTS

In [164], the authors performed a number of empirical experiments with both simulated and real-world data (MNIST etc.) to investigate how deep neural nets operate on topologies of data. Their findings support the view that well-trained deep neural networks simplify the topology (with respect to the topological complexity defined in (8.1)) of the input data as the input data is passed through the layers of the network.

Among these experiments, the authors of [164] monitored the topological complexity of the data at different layer and found that

- Well-trained neural nets gradually simplify the complicated topology of the input data until it becomes linearly separable in the output space
- The reduction in topological complexity β is much faster for ReLU activation compared to hyperbolic tangent activation, because many-to-one maps (like ReLu) are nonhomeomorphic.
- Deeper neural nets are better at topology simplification as the reduction of topological complexity mainly happens in the later layers.

Experimental Approach

Our goal is to investigate the simplification of topology by our well-trained neural network (as described in section 8.1) on time series data. We will follow the experimental approach for the real-world data adopted by the authors of [164], except that we monitor the topological complexities for the input data and the output data (embeddings) only, rather than the complexity at any intermediate layers.

First, a consistent constant δ needs to be specified as the data is passed through all layers of the neural network, taking into account that the input data would have different scale and density when it is passed through the layers of the network. One reasonable choice for δ is the the graph geodesic distance δ_k on the k -nearest neighbors graph determined by the point cloud data X . δ_k of a pair of vertices is defined to be the minimal number of edges between them. The metric δ_k normalizes the distances across all layers of the neural network and allows us to compare the topological complexities of the point clouds embedded in the Euclidean spaces of different dimensions.

The Vietoris-Rips complexes of the inputs and the embeddings are then built. The Vietoris-Rips complexes have two parameters ϵ and k . For Simulated data, i.e. n points X sampled from a known manifold $M \subset \mathbb{R}^d$, one could always tune the ϵ and k to match the true topology of M that are known in advance by directly comparing the Betti numbers one after another. For real world data X , we cannot calibrate ϵ and k with respect to the true topology of X before we have the persistent diagram. Computing the persistent diagram requires computing the homology at full range of scale which would be extremely computationally intensive, especially for large high dimensional data set in deep learning.

However, we can still compare the topological complexity of the inputs and that of the embeddings with a fixed list of (ϵ, k) pairs; the results of this comparison are shown below.

Results

Time series data X of size 20000×365 (20000 samples with 365 features) was used as the input. As the forward pass $F_n(X) = Y$ through the neural network F_n as described above is computed, the dimension of the embedded data Y is of size 20000×265 .

Before we calculate the topological complexity on both inputs and embeddings, we will need to denoise the data, reduce the dimension and downsample the data to a reasonable size so that the data is usable for homology calculation.

For this analysis, Principal Component Analysis (PCA) was applied to both data sets to compute the first 100 principal components. See figure 8.5 for a visualization of the first three principal components.

The data is then randomly downsampled to 500 data points each by a random selection of indices. We denote the downsampled, PCA transformed data sets by \hat{X} and \hat{Y} .

Now we can construct k -NN graphs on \hat{X} and \hat{Y} and obtain δ_k the graph geodesic distance on each graph. For a fixed pair ϵ and k , we construct the Vietorik-Rips complexes $\text{VR}_{\epsilon,k}(\hat{X})$ and $\text{VR}_{\epsilon,k}(\hat{Y})$ with the topological data analysis library *Ghudi*. We then use *Ghudi* to calculate the Betti numbers and the topological complexities of the Vietorik-Rips complexes. Topological complexities for $(\epsilon, k) \in \{1.5, 2.5, 3.5, 5.5\} \times \{5, 8, 10, 12\} := I$ (figure 8.6) were calculated.

As the topological complexity got reduced, namely $\beta^{(\epsilon,k)}(\hat{X}) > \beta^{(\epsilon,k)}(\hat{Y})$, for all (ϵ, k) in I , where $\beta^{(\epsilon,k)}$ denotes the topological complexity for $\text{VR}_{\epsilon,k}$. Even though we didn't exhaust the (ϵ, k) to get the persistent diagrams, the evidence supports the view that a well-trained neural net simplifies the topology of the input data. While the paper by Naizat et al. [164] experiments with feedforward networks at multiple layers, we are interested with the final (embedding) layer of a transformer network.

8.4 CONCLUSION AND ADDITIONAL RESEARCH AVENUES

The experimental results confirm the hypothesis of topological simplification between the input time series space and the learned embeddings space generated by using the training task described in Section 8.1. In the discussion to follow, we used the terms “distance” and “kernels” somewhat synonymously, due to the practice (prevalent in machine learning) of using positive-definite kernels as a more general notion of distances [165]. Euclidean distance was used to analyze both our input and embedding space, but additional analysis is required to understand whether the topological simplification achieved by the network is meaningful in a practical sense. That is because Euclidean distance is not commonly used as a kernel for analyzing time series data, and a more fair comparison may be to analyze Euclidean distance in embedding space compared to commonly used metrics like correlation / cosine similarity, or covariance distance. It could also be interesting to explore the relation of the embedding distance to other kernels, such as Earth-Mover's or Wasserstein distance [166].

The experiments above, as well as experimentation performed in Chapter 7, have also shown that transformer networks can struggle to learn “global” features such as average variance of the time series over time, which was observed by concatenating some of these features to the

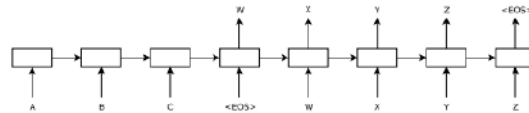


Figure 8.1: Illustration of a sequence-to-sequence machine translation model, adopted from [130]

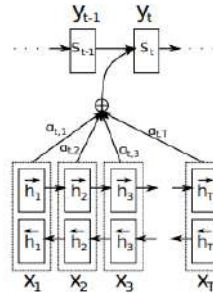


Figure 8.2: Illustration of a recurrent sequence-to-sequence model augmented with attention, adopted from [122]

neurons prior to the output layer. It also stands to reason that improving the performance of the Siamese encoder should in turn generate better embeddings (and topologically simpler). As such, further experiments with the network architecture itself can be an interesting direction of research, applying architectures like temporal convolutional networks, which have recently been showing promising results on other complex time series tasks [167] and should perform better in capturing the global structure of the time series. A mix of transformer and TCN layers was found to be the optimal setup in [15], but further experimentation with more computational resources and larger data samples may reveal that different architectures perform at a large scale, as was found in many other areas of primary machine learning research.

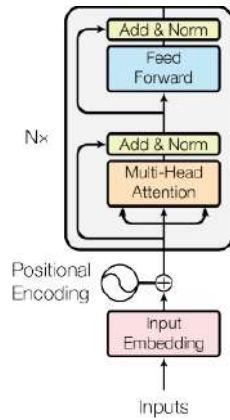


Figure 8.3: Illustration of a recurrent sequence-to-sequence model augmented with attention, adopted from [87]

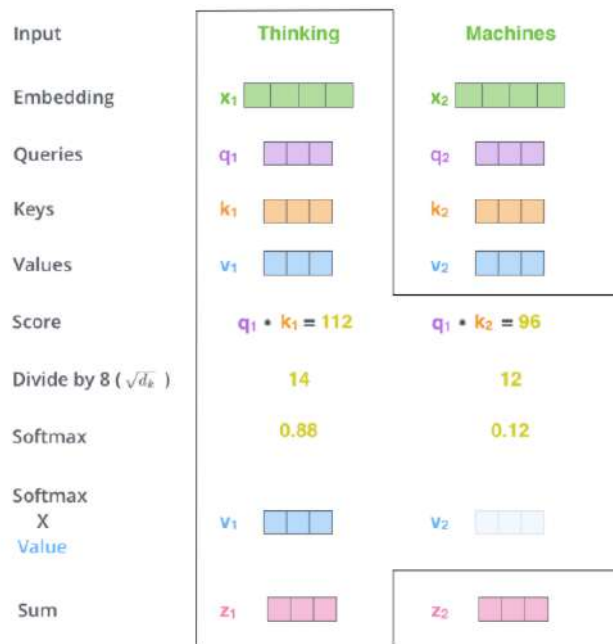


Figure 8.4: Illustration of token representation mixing via the self-attention mechanism, adopted from [168]

Table 8.1: Choice of Hyperparameters

Hyperparameter Description	Final Choice
Lag window of T days	128
Number of <i>feature time series</i> d_{feats}	7
Number of <i>target metrics</i> d_{trgt}	11
Embedding dimension n_{embed}	256
Number of transformer blocks n_{blocks}	6
Transformer block number of heads n_{heads}	16
Transformer block feed-forward network dimension $n_{\text{feedforward}}$	1024
Transformer block attention dropout $P_{\text{attention_drop}}$	0.1
Transformer block residual dropout $P_{\text{residual_drop}}$	0.1
N_{tokens} for feature computation	3
Internal features computed based on token embeddings	Standard deviation, L1, L2, and L3 norms ($N_{\text{features}} = 4$)
Hidden layer dimensions d_{hidden}	256, 128
Hidden layer dropout $P_{\text{hidden_drop}}$	0.25
Optimizer	RAadam
Learning rate r_{learning}	5×10^{-5}
Batch size	32
Evaluate every	200 steps
Early stopping	no improvement after 3 consecutive evaluations

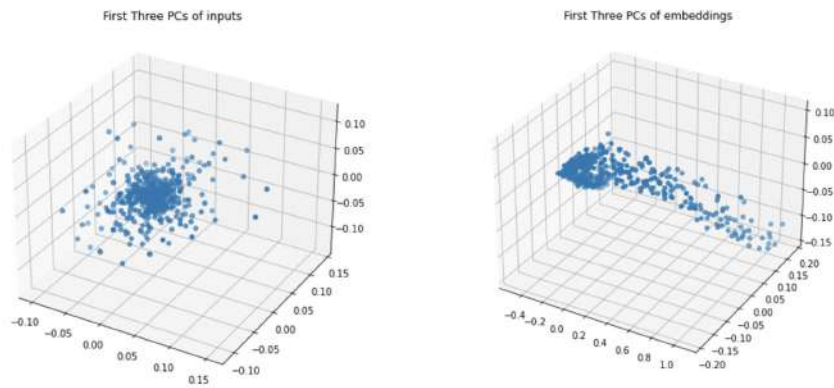


Figure 8.5: PCA transformed inputs and embeddings: First three principal components

epsilon	1.5				2.5				3.5				5.5			
	5	8	10	12	5	8	10	12	5	8	10	12	5	8	10	12
inputs	396.0	613.0	668.0	684.0	380.0	165.0	171.0	88.0	243.0	87.0	60.0	5.0	49.0	4.0	4.0	4.0
embeddings	118.0	133.0	112.0	68.0	101.0	39.0	21.0	11.0	54.0	6.0	1.0	1.0	17.0	2.0	1.0	2.0

Figure 8.6: Topological complexities for inputs and embeddings

BIBLIOGRAPHY

- [1] L. Bachelier. “Théorie de la spéculation”. In: *Annales Scientifiques de l’Ecole Normale Supérieure* 17 (1900). Reprinted in P. H. Cootner (ed.), 1964, *The Random Character of Stock Market Prices*, MIT Press, Cambridge, Mass., pp. 21–88.
- [2] H. Markowitz. “Portfolio Selection”. In: *Journal of Finance* 7.1 (1952), pp. 77–91.
- [3] A. C. Pigou. *The Economics of Welfare*. Introduces the concept of externalities and argues that market prices, guided by the “invisible hand,” fail to reflect social costs, leading to market inefficiencies. London: Macmillan, 1920.
- [4] A. Smith. *An Inquiry into the Nature and Causes of the Wealth of Nations*. Introduces the concept of the “invisible hand,” describing the self-regulating nature of markets. London: W. Strahan and T. Cadell, 1776.
- [5] Upright Project. *Quantifying the Net Impact of Companies*. White Paper. Upright Project, 2020. URL: <https://www.uprightproject.com/whitepapers/model/>.
- [6] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.
- [7] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Interscience, 2007.
- [8] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. 2nd. MIT Press, 2018.
- [9] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.
- [10] W. H. Fleming and H. M. Soner. *Controlled Markov Processes and Viscosity Solutions*. New York: Springer-Verlag, 2006.
- [11] K. Miettinen. *Nonlinear Multiobjective Optimization*. Boston: Kluwer Academic Publishers, 1999.
- [12] Z. S. Jiang, D. Xu, and J. Liang. “A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem”. In: *arXiv preprint arXiv:1706.10059* (2017).
- [13] A. Sokolov et al. “Weak Supervision and Black-Litterman for Automated ESG Portfolio Construction”. In: *The Journal of Financial Data Science* 3.3 (2021), pp. 129–138. DOI: [10.3905/jfds.2021.1.070](https://doi.org/10.3905/jfds.2021.1.070).
- [14] A. Sokolov et al. “Neural Embeddings of Financial Time-Series Data”. In: *The Journal of Financial Data Science* 2.4 (Oct. 2020), pp. 33–43. DOI: [10.3905/jfds.2020.1.041](https://doi.org/10.3905/jfds.2020.1.041).
- [15] A. Sokolov et al. “RIFT: Pretraining and Applications for Representations of Interrelated Financial Time Series”. In: *The Journal of Financial Data Science* 5.4 (Sept. 2023), jfds.2023.1.138. DOI: [10.3905/jfds.2023.1.138](https://doi.org/10.3905/jfds.2023.1.138).
- [16] A. Sokolov et al. “Towards Automating Causal Discovery in Financial Markets and Beyond”. In: *SSRN Electronic Journal* (Jan. 2024). DOI: [10.2139/ssrn.4679414](https://doi.org/10.2139/ssrn.4679414).

- [17] G. S. Becker. “A Theory of the Allocation of Time”. In: *The Economic Journal* 75.299 (1965), pp. 493–517.
- [18] M. Fleurbaey. “Beyond GDP: The Quest for a Measure of Social Welfare”. In: *Journal of Economic Literature* 47.4 (2009), pp. 1029–1075.
- [19] F. P. Ramsey. “A Mathematical Theory of Saving”. In: *The Economic Journal* 38.152 (1928), pp. 543–559.
- [20] N. Stern. *The Economics of Climate Change: The Stern Review*. Cambridge University Press, 2006.
- [21] A. Sokolov et al. “Building Machine Learning Systems for Automated ESG Scoring”. In: *The Journal of Impact and ESG Investing* 1.3 (Feb. 2021), pp. 39–50. DOI: [10.3905/jesg.2021.1.010](https://doi.org/10.3905/jesg.2021.1.010).
- [22] J. Mossin. “Optimal Multiperiod Portfolio Policies”. In: *Journal of Business* 41.2 (1968), pp. 215–229.
- [23] P. A. Samuelson. “Lifetime Portfolio Selection by Dynamic Stochastic Programming”. In: *Review of Economics and Statistics* 51.3 (1969), pp. 239–246.
- [24] R. C. Merton. “Lifetime Portfolio Selection under Uncertainty: The Continuous-Time Case”. In: *Review of Economics and Statistics* 51.3 (1969), pp. 247–257.
- [25] J. M. Poterba and L. H. Summers. “Mean Reversion in Stock Prices: Evidence and Implications”. In: *Journal of Financial Economics* 22.1 (1988), pp. 27–59.
- [26] M. Grebeck and S. T. Rachev. “Stochastic Programming Methods in Asset-Liability Management”. In: *Investment Management and Financial Innovations* 2.1 (2005), pp. 82–90.
- [27] V. V. Acharya and L. H. Pedersen. “Asset Pricing with Liquidity Risk”. In: *Journal of Financial Economics* 77.2 (2005), pp. 375–410.
- [28] W. D. Nordhaus et al. *The DICE Model: Background and Structure of a Dynamic Integrated Climate-Economy Model of the Economics of Global Warming*. Tech. rep. Cowles Foundation Discussion Paper No. 1009, Yale University, 1992.
- [29] L. Pástor, R. F. Stambaugh, and L. A. Taylor. “Sustainable Investing in Equilibrium”. In: *Journal of Financial Economics* 142.2 (2021), pp. 550–571. DOI: [10.1016/j.jfineco.2020.12.011](https://doi.org/10.1016/j.jfineco.2020.12.011).
- [30] H. Hong and M. Kacperczyk. “The Price of Sin: The Effects of Social Norms on Markets”. In: *Journal of Financial Economics* 93.1 (2009), pp. 15–36.
- [31] R. Heinkel, A. Kraus, and J. Zechner. “The Effect of Green Investment on Corporate Behavior”. In: *Journal of Financial and Quantitative Analysis* 36.4 (2009), pp. 431–449.
- [32] L. H. Pedersen, S. Fitzgibbons, and L. Pomorski. “Responsible Investing: The ESG-Efficient Frontier”. In: *Journal of Financial Economics* 142.2 (2021), pp. 572–597.
- [33] E. F. Fama and K. R. French. “Common risk factors in the returns on stocks and bonds”. In: *Journal of Financial Economics* 33.1 (1993), pp. 3–56. ISSN: 0304-405X. DOI: [https://doi.org/10.1016/0304-405X\(93\)90023-5](https://doi.org/10.1016/0304-405X(93)90023-5). URL: <https://www.sciencedirect.com/science/article/pii/0304405X93900235>.

- [34] J. H. Cochrane. “Production-Based Asset Pricing and the Link Between Stock Returns and Economic Fluctuations”. In: *Journal of Finance* 46.1 (1991), pp. 209–237.
- [35] L. Zhang. “The Value Premium”. In: *Journal of Finance* 60.1 (2005), pp. 67–103.
- [36] F. Black and R. Litterman. “Global Portfolio Optimization”. In: *Financial Analysts Journal* 48.5 (1992), pp. 28–43.
- [37] J. M. Mulvey and W. T. Ziemba, eds. *Worldwide Asset and Liability Modelling*. Cambridge, UK: Cambridge University Press, 1998.
- [38] H. Markowitz. “Consumption, Investment and Insurance in the Game of Life”. In: *Journal of Investment Management* 13.3 (2015), pp. 5–23.
- [39] T. I. Jensen, B. T. Kelly, and L. H. Pedersen. “Is There a Replication Crisis in Finance?” In: *The Journal of Finance* 78.5 (2023), pp. 2465–2518. DOI: <https://doi.org/10.1111/jofi.13249>.
- [40] Y. Bengio, A. Courville, and P. Vincent. “Representation Learning: A Review and New Perspectives”. In: *IEEE transactions on pattern analysis and machine intelligence* 35 (Aug. 2013), pp. 1798–1828. DOI: [10.1109/TPAMI.2013.50](https://doi.org/10.1109/TPAMI.2013.50).
- [41] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Cambridge, MA: MIT Press, 2016.
- [42] J. Peters, D. Janzing, and B. Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. Cambridge, MA: MIT Press, 2017.
- [43] N. Brown and T. Sandholm. “Libratus: The Superhuman AI for No-Limit Poker”. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017) – Demonstration Track*. 2017, pp. 5226–5230. DOI: [10.24963/ijcai.2017/765](https://doi.org/10.24963/ijcai.2017/765).
- [44] de Prado, Marcos Lopez. “Clustered Feature Importance (Presentation Slides)”. In: (2020). SSRN: <https://ssrn.com/abstract=3517595>.
- [45] M. L. de Prado. “Causal Factor Investing”. In: *Elements in Quantitative Finance*. Cambridge University Press, 2023. ISBN: 9781009397315. DOI: <https://doi.org/10.1017/9781009397315>.
- [46] E. F. Fama and K. R. French. “A Five-Factor Asset Pricing Model”. In: *Journal of Financial Economics* 116.1 (2015), pp. 1–22. DOI: [10.1016/j.jfineco.2014.10.010](https://doi.org/10.1016/j.jfineco.2014.10.010).
- [47] X. Zhang et al. “A causal feature selection algorithm for stock prediction modeling”. In: *Neurocomputing* 142.1 (2014), pp. 48–59. DOI: <https://doi.org/10.1016/j.neucom.2014.01.057>.
- [48] D. Polakow, T. Gebbie, and E. Flint. *Epistemic Limits of Empirical Finance: Causal Reductionism and Self-Reference*. arXiv preprint arXiv:2311.16570. 2023. DOI: [10.48550/arXiv.2311.16570](https://doi.org/10.48550/arXiv.2311.16570). URL: <https://arxiv.org/abs/2311.16570>.
- [49] Z. Hao et al. “Causal discovery on high dimensional data”. In: *Applied Intelligence* 42.3 (2015), pp. 594–607. DOI: <https://doi.org/10.1007/s10489-014-0607-0>. URL: <https://dl.acm.org/doi/abs/10.1007/s10489-014-0607-0>.
- [50] U. Hasan and M. O. Gani. “KCRL: A Prior Knowledge Based Causal Discovery Framework with Reinforcement Learning”. In: *Proceedings of the 7th Machine Learning for Healthcare Conference, PMLR 182:691-714*. 2022.

- [51] A. K. Lampinen et al. *Passive Learning of Active Causal Strategies in Agents and Language Models*. 2023. DOI: [10.48550/arXiv.2305.16183](https://doi.org/10.48550/arXiv.2305.16183). arXiv: [2305.16183](https://arxiv.org/abs/2305.16183) [cs.AI].
- [52] C. Zhang et al. *Understanding Causality with Large Language Models: Feasibility and Opportunities*. 2023. DOI: [10.48550/arXiv.2304.05524](https://doi.org/10.48550/arXiv.2304.05524). arXiv: [2304.05524](https://arxiv.org/abs/2304.05524) [cs.CL].
- [53] Z. Jin et al. *Can Large Language Models Infer Causation from Correlation?* 2023. DOI: [10.48550/arXiv.2306.05836](https://doi.org/10.48550/arXiv.2306.05836). arXiv: [2306.05836](https://arxiv.org/abs/2306.05836) [cs.CL].
- [54] E. Kiciman et al. *Causal Reasoning and Large Language Models: Opening a New Frontier for Causality*. 2023. DOI: [10.48550/arXiv.2305.00050](https://doi.org/10.48550/arXiv.2305.00050). arXiv: [2305.00050](https://arxiv.org/abs/2305.00050) [cs.AI].
- [55] N. Naik, A. Khandelwal, M. Joshi, et al. *Applying Large Language Models for Causal Structure Learning in Non Small Cell Lung Cancer*. 2023. DOI: [10.48550/arXiv.2311.07191](https://doi.org/10.48550/arXiv.2311.07191). arXiv: [2311.07191](https://arxiv.org/abs/2311.07191) [cs.AI].
- [56] S. Long et al. *Causal Discovery with Language Models as Imperfect Experts*. 2023. DOI: [10.48550/arXiv.2307.02390](https://doi.org/10.48550/arXiv.2307.02390). arXiv: [2307.02390](https://arxiv.org/abs/2307.02390) [cs.AI].
- [57] T. Ban et al. *From Query Tools to Causal Architects: Harnessing Large Language Models for Advanced Causal Discovery from Data*. 2023. DOI: [10.48550/arXiv.2306.16902](https://doi.org/10.48550/arXiv.2306.16902). arXiv: [2306.16902](https://arxiv.org/abs/2306.16902) [cs.AI].
- [58] N. Hollmann, S. Müller, and F. Hutter. *Large Language Models for Automated Data Science: Introducing CAAFE for Context-Aware Automated Feature Engineering*. 2023. DOI: [10.48550/arXiv.2305.03403](https://doi.org/10.48550/arXiv.2305.03403). arXiv: [2305.03403](https://arxiv.org/abs/2305.03403) [cs.LG].
- [59] X.-Y. Liu et al. *FinGPT: Democratizing Internet-scale Data for Financial Large Language Models*. 2023. DOI: [10.48550/arXiv.2307.10485](https://doi.org/10.48550/arXiv.2307.10485). arXiv: [2307.10485](https://arxiv.org/abs/2307.10485) [q-fin.CP].
- [60] B. Yu. *Benchmarking Large Language Model Volatility*. 2023. DOI: [10.48550/arXiv.2311.15180](https://doi.org/10.48550/arXiv.2311.15180). arXiv: [2311.15180](https://arxiv.org/abs/2311.15180) [q-fin.TR].
- [61] H. Kang and X.-Y. Liu. *Deficiency of Large Language Models in Finance: An Empirical Examination of Hallucination*. 2023. DOI: [10.48550/arXiv.2311.15548](https://doi.org/10.48550/arXiv.2311.15548). arXiv: [2311.15548](https://arxiv.org/abs/2311.15548) [q-fin.CP].
- [62] J. Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: [2201.11903](https://arxiv.org/abs/2201.11903) [cs.CL].
- [63] OpenAI. *openai/openai-cookbook: Examples and guides for using the openai api*. 2023. URL: <https://github.com/openai/openai-cookbook> (visited on 10/30/2023).
- [64] J. Pearl. “Causal diagrams for empirical research.” In: *Biometrika* 82 (4 1995), pp. 669–710.
- [65] I. Shpitser and J. Pearl. “Identification of joint interventional distributions in recursive semi-Markovian causal models”. In: *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. AAAI Press. Menlo Park, CA, 2006, pp. 1219–1226.
- [66] Y. Huang and M. Valtorta. “Pearl’s calculus of intervention is complete”. In: *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*. Ed. by R. Dechter and T. Richardson. AUAI Press. Corvallis, OR, 2006, pp. 217–224.
- [67] A. Sharma and E. Kiciman. *DoWhy: An End-to-End Library for Causal Inference*. arXiv preprint arXiv:2011.04216. 2020. DOI: [10.48550/arXiv.2011.04216](https://doi.org/10.48550/arXiv.2011.04216). URL: <https://arxiv.org/abs/2011.04216>.

- [68] E. F. Fama and J. D. MacBeth. “Risk, Return, and Equilibrium: Empirical Tests”. In: *Journal of Political Economy* 81.3 (1973), pp. 607–636. ISSN: 00223808, 1537534X. URL: <http://www.jstor.org/stable/1831028> (visited on 12/05/2023).
- [69] K. Hou, C. Xue, and L. Zhang. “Replicating Anomalies”. In: *The Review of Financial Studies* 33.5 (Dec. 2018), pp. 2019–2133. ISSN: 0893-9454. DOI: [10.1093/rfs/hhy131](https://doi.org/10.1093/rfs/hhy131). eprint: <https://academic.oup.com/rfs/article-pdf/33/5/2019/33710468/hhy131.pdf>. URL: <https://doi.org/10.1093/rfs/hhy131>.
- [70] F. Black and M. Scholes. “The Pricing of Options and Corporate Liabilities”. In: *Journal of Political Economy* 81.3 (1973), pp. 637–654.
- [71] R. C. Merton. “On the Pricing of Corporate Debt: The Risk Structure of Interest Rates”. In: *Journal of Finance* 29.2 (1974), pp. 449–470.
- [72] P. Collin-Dufresne, R. S. Goldstein, and J. S. Martin. “The Determinants of Credit Spread Changes”. In: *The Journal of Finance* 56.6 (2001), pp. 2177–2207.
- [73] L. Chen, P. Collin-Dufresne, and R. S. Goldstein. “On the Relation Between the Credit Spread Puzzle and the Equity Premium Puzzle”. In: *Review of Financial Studies* 21.6 (2008), pp. 2205–2243.
- [74] S. Gilchrist and E. Zakrajšek. *Replication Data for: Credit Spreads and Business Cycle Fluctuations*. Ann Arbor, MI: Inter-university Consortium for Political and Social Research [distributor]. Distributed 2019-10-11. Nashville, TN, 2012. DOI: [10.3886/E112536V1](https://doi.org/10.3886/E112536V1). URL: <https://doi.org/10.3886/E112536V1>.
- [75] G. Friede, T. Busch, and A. Bassen. “ESG and Financial Performance: Aggregated Evidence from More than 2000 Empirical Studies”. In: *Journal of Sustainable Finance & Investment* Volume 5.Issue 4 (2015), pp. 210–233. DOI: [10.1080/20430795.2015.1118917](https://doi.org/10.1080/20430795.2015.1118917).
- [76] C. Pinney, S. Lawrence, and S. Lau. “Sustainability and Capital Markets—Are We There Yet?” In: *Journal of Applied Corporate Finance* 31.2 (2019), pp. 86–91. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/jacf.12350>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/jacf.12350>.
- [77] R. I. Association. *2018 Canadian Responsible Investment: Trends Report*. URL: <https://www.riacanada.ca/content/uploads/2018/10/2018-RI-Trends-Report-FINAL-WEB-1.pdf>. (accessed: 30.11.2019).
- [78] A. Pierron. *ESG Data: Mainstream Consumption, Bigger Spending*. 2019. URL: <http://www.opimas.com/research/428/detail/>. (accessed: 30.11.2019).
- [79] B. Beardsley et al. *Global Asset Management 2018: The Digital Metamorphosis*. Report. Boston Consulting Group, July 2018.
- [80] R. Henriksson et al. “Integrating ESG in Portfolio Construction”. In: *The Journal of Portfolio Management* 45 (4) (2019), pp. 67–81.
- [81] F. Climent and P. Soriano. “Green and Good? The Investment Performance of US Environmental Mutual Funds”. In: *Journal of Business Ethics* 103.2 (2011), pp. 275–287. DOI: [10.1007/s10551-011-0865-2](https://doi.org/10.1007/s10551-011-0865-2).
- [82] A. Nematzadeh et al. *Empirical Study on Detecting Controversy in Social Media*. arXiv preprint arXiv:1909.01093. 2019. DOI: [10.48550/arXiv.1909.01093](https://doi.org/10.48550/arXiv.1909.01093). URL: <https://arxiv.org/abs/1909.01093>.

- [83] D. Chen et al. “Incorporating Fine-grained Events in Stock Movement Prediction”. In: *Proceedings of the Second Workshop on Economics and Natural Language Processing*. Hong Kong: Association for Computational Linguistics, 2019, pp. 31–40. DOI: [10.18653/v1/D19-5105](https://doi.org/10.18653/v1/D19-5105). URL: <https://aclanthology.org/D19-5105>.
- [84] J. Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1*. Minneapolis, MN: Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: [10.18653/v1/N19-1423](https://doi.org/10.18653/v1/N19-1423). URL: <https://aclanthology.org/N19-1423>.
- [85] Y.-H. Lee, W.-J. Tsao, and T.-H. Chu. “Use of Ontology to Support Concept-Based Text Categorization”. In: *Designing E-Business Systems. Markets, Services, and Networks (WEB 2008)*. Ed. by C. Weinhardt, S. Luckner, and J. Stöber. Vol. 22. Lecture Notes in Business Information Processing. Berlin, Heidelberg: Springer, 2009, pp. 201–213. DOI: [10.1007/978-3-642-01256-3_17](https://doi.org/10.1007/978-3-642-01256-3_17).
- [86] A. Wang et al. “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: *International Conference on Learning Representations*. 2019. URL: <https://openreview.net/forum?id=rJ4km2R5t7>.
- [87] A. Vaswani et al. “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS’17*. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.
- [88] F. Zhuang et al. *A Comprehensive Survey on Transfer Learning*. 2019. DOI: [10.48550/ARXIV.1911.02685](https://doi.org/10.48550/ARXIV.1911.02685). URL: <https://arxiv.org/abs/1911.02685>.
- [89] Z. Lan et al. “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations”. In: *International Conference on Learning Representations*. 2020. URL: <https://openreview.net/forum?id=H1eA7AEtvs>.
- [90] T. Wolf et al. “Huggingface’s transformers: State-of-the-art natural language processing”. In: (2019). arXiv: [1910.03771](https://arxiv.org/abs/1910.03771) [cs.CL].
- [91] L. Liu et al. “On the Variance of the Adaptive Learning Rate and Beyond”. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL: <https://openreview.net/forum?id=rkgz2aEKDr>.
- [92] J. Deng et al. “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: [10.1109/CVPR.2009.5206848](https://doi.org/10.1109/CVPR.2009.5206848).
- [93] F. Black and R. Litterman. “Asset Allocation”. In: *The Journal of Fixed Income* 1.2 (Sept. 1991), pp. 7–18. DOI: [10.3905/jfi.1991.408013](https://doi.org/10.3905/jfi.1991.408013). URL: <https://doi.org/10.3905/2Fjfi.1991.408013>.
- [94] A. Bryan et al. *Passive Sustainable Funds: The Global Landscape 2020*. Tech. rep. Sept. 2020.

- [95] S. M. Hartzmark and A. B. Sussman. “Do Investors Value Sustainability? A Natural Experiment Examining Ranking and Fund Flows”. In: *The Journal of Finance* 74.6 (2019), pp. 2789–2837. DOI: [10.1111/jofi.12841](https://doi.org/10.1111/jofi.12841). eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/jofi.12841>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.12841>.
- [96] J.P. Morgan. “Why COVID-19 Could Prove to Be a Major Turning Point for ESG Investing”. In: *JP Morgan* (July 2020). URL: <https://www.jpmorgan.com/insights/research/covid-19-esg-investing> (visited on 10/12/2020).
- [97] D. Balji. “Sustainable investing surges in Canada amid pandemic, protests”. In: *BNN Bloomberg* (June 2020). URL: <https://www.bnnbloomberg.ca/sustainable-investing-surges-in-canada-amid-pandemic-protests-1.1453425> (visited on 10/12/2020).
- [98] E. Whieldon, R. Clark, and M. Copley. “ESG funds outperform S&P 500 amid COVID-19, helped by tech stock boom”. In: *S&P Global Market Intelligence* (Aug. 2020). URL: <https://www.spglobal.com/marketintelligence/en/news-insights/latest-news-headlines/esg-funds-outperform-s-p-500-amid-covid-19-helped-by-tech-stock-boom-59850808> (visited on 10/12/2020).
- [99] B. Leitao. “How ESG ETFs Have Performed in the Sell-Off”. In: *Morningstar ETF Research & Insights* (Apr. 2020). URL: <https://www.morningstar.co.uk/uk/news/201154/how-esg-etfs-have-performed-in-the-sell-off.aspx> (visited on 10/12/2020).
- [100] S. Kotsantonis and G. Serafeim. “Four Things No One Will Tell You About ESG Data”. In: *Journal of Applied Corporate Finance* 31.2 (2019), pp. 50–58. DOI: [10.1111/jacf.12346](https://doi.org/10.1111/jacf.12346). eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/jacf.12346>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/jacf.12346>.
- [101] D. M. Christensen, G. Serafeim, and A. Sikochi. “Why Is Corporate Virtue in the Eye of the Beholder? The Case of ESG Ratings”. In: *The Accounting Review* 97.1 (2022), pp. 147–175. DOI: [10.2308/TAR-2019-0506](https://doi.org/10.2308/TAR-2019-0506).
- [102] F. Berg, J. F. Kölbel, and R. Rigobon. “Aggregate Confusion: The Divergence of ESG Ratings”. In: *Review of Finance* 26.6 (2022), pp. 1315–1344. DOI: [10.1093/rof/rfac033](https://doi.org/10.1093/rof/rfac033).
- [103] New York Times. *Article Search API*. 2020. URL: <https://developer.nytimes.com/docs/articlesearch-product/1/overview>.
- [104] A. Ratner, B. Hancock, and C. Ré. “The Role of Massively Multi-Task and Weak Supervision in Software 2.0”. In: *CIDR*. 2019.
- [105] Y. Liu et al. “RoBERTa: A Robustly Optimized BERT Pretraining Approach”. In: *Proceedings of the International Conference on Learning Representations (ICLR 2020)*. 2020. arXiv: [1907.11692](https://arxiv.org/abs/1907.11692) [cs.CL]. URL: <https://openreview.net/forum?id=SyxS0T4tvS>.
- [106] Z. Yang et al. “XLNet: Generalized Autoregressive Pretraining for Language Understanding”. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 5753–5763. URL: <http://papers.nips.cc/paper/8812-xlnet-generalized-autoregressive-pretraining-for-language-understanding.pdf>.

- [107] A. Conneau et al. “Unsupervised Cross-lingual Representation Learning at Scale”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020, pp. 8440–8451. DOI: [10.18653/v1/2020.acl-main.747](https://doi.org/10.18653/v1/2020.acl-main.747). URL: <https://aclanthology.org/2020.acl-main.747>.
- [108] T. Brown et al. “Language Models are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>.
- [109] V. Sanh et al. “DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter”. In: *Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC²) at NeurIPS*. arXiv:1910.01108. 2019. DOI: [10.48550/arXiv.1910.01108](https://doi.org/10.48550/arXiv.1910.01108). URL: <https://arxiv.org/abs/1910.01108>.
- [110] X. Jiao et al. “TinyBERT: Distilling BERT for Natural Language Understanding”. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. 2020, pp. 4163–4174. DOI: [10.18653/v1/2020.findings-emnlp.372](https://doi.org/10.18653/v1/2020.findings-emnlp.372).
- [111] W. Li et al. *Online Deep Metric Learning*. 2018. arXiv: [1805.05510](https://arxiv.org/abs/1805.05510) [cs.CV].
- [112] X. Li et al. “Semi-supervised clustering with deep metric learning and graph embedding”. In: *World Wide Web* (2019). DOI: [10.1007/s11280-019-00723-8](https://doi.org/10.1007/s11280-019-00723-8).
- [113] Bengio, Yoshua. “Deep Learning of Representations for Unsupervised and Transfer Learning”. In: *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*. Ed. by I. Guyon et al. Vol. 27. Proceedings of Machine Learning Research. Bellevue, Washington, USA: PMLR, Feb. 2012, pp. 17–36.
- [114] T. Mikolov et al. “Distributed Representations of Words and Phrases and Their Compositionality”. In: *NIPS’13*. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 3111–3119.
- [115] D. Cer et al. “Universal Sentence Encoder for English”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 169–174. DOI: [10.18653/v1/D18-2029](https://doi.org/10.18653/v1/D18-2029). URL: <https://www.aclweb.org/anthology/D18-2029>.
- [116] K. He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016, pp. 770–778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [117] K. Varaku. *Stock Price Forecasting and Hypothesis Testing Using Neural Networks*. 2019. arXiv: [1908.11212](https://arxiv.org/abs/1908.11212) [q-fin.ST].
- [118] D. León et al. “Clustering algorithms for Risk-Adjusted Portfolio Construction”. In: *Procedia Computer Science* 108 (2017). International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland, pp. 1334–1343. ISSN: 1877-0509. DOI: <https://doi.org/10.1016/j.procs.2017.05.185>. URL: <http://www.sciencedirect.com/science/article/pii/S187705091730772X>.
- [119] W. Bao, J. Yue, and Y. Rao. “A deep learning framework for financial time series using stacked autoencoders and long-short term memory”. In: *PLOS One* (2017). DOI: [10.1371/journal.pone.0180944](https://doi.org/10.1371/journal.pone.0180944).

- [120] D. Silver et al. “A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play”. In: *Science* 362.6419 (2018), pp. 1140–1144. ISSN: 0036-8075. DOI: [10.1126/science.aar6404](https://doi.org/10.1126/science.aar6404). eprint: <https://science.sciencemag.org/content/362/6419/1140.full.pdf>. URL: <https://science.sciencemag.org/content/362/6419/1140>.
- [121] S. Siami-Namini, N. Tavakoli, and A. Siami Namin. *A Comparative Analysis of Forecasting Financial Time Series Using ARIMA, LSTM, and BiLSTM*. 2019. arXiv: [1911.09512](https://arxiv.org/abs/1911.09512) [cs.LG].
- [122] D. Bahdanau, K. Cho, and Y. Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2014. arXiv: [1409.0473](https://arxiv.org/abs/1409.0473) [cs.CL].
- [123] S. Li et al. *Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting*. 2019. arXiv: [1907.00235](https://arxiv.org/abs/1907.00235) [cs.LG].
- [124] J. Ward Jr. “Hierarchical Grouping to Optimize an Objective Function”. In: *Journal of the American Statistical Association* 58.301 (1963), pp. 236–244. DOI: [10.1080/01621459.1963.10500845](https://doi.org/10.1080/01621459.1963.10500845). eprint: <https://www.tandfonline.com/doi/pdf/10.1080/01621459.1963.10500845>. URL: <https://www.tandfonline.com/doi/abs/10.1080/01621459.1963.10500845>.
- [125] P. Sermanet et al. “Pedestrian Detection with Unsupervised Multi-stage Feature Learning”. In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 3626–3633. DOI: [10.1109/CVPR.2013.465](https://doi.org/10.1109/CVPR.2013.465).
- [126] N. Tajbakhsh et al. “Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?” In: *IEEE Transactions on Medical Imaging* 35 (2016), pp. 1299–1312.
- [127] Z. Xinxi. “Single task fine-tune BERT for text classification”. In: *2nd International Conference on Computer Vision, Image, and Deep Learning*. Ed. by B. H. bin Ahmad and F. Cen. Vol. 11911. International Society for Optics and Photonics. SPIE, 2021, pp. 434–439. DOI: [10.1117/12.2604768](https://doi.org/10.1117/12.2604768). URL: <https://doi.org/10.1117/12.2604768>.
- [128] R. M. Rao et al. “MSA Transformer”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila and T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, pp. 8844–8856. URL: <https://proceedings.mlr.press/v139/rao21a.html>.
- [129] A. Senior et al. “Improved protein structure prediction using potentials from deep learning”. In: *Nature* 577 (Jan. 2020), pp. 1–5. DOI: [10.1038/s41586-019-1923-7](https://doi.org/10.1038/s41586-019-1923-7).
- [130] I. Sutskever, O. Vinyals, and Q. V. Le. “Sequence to Sequence Learning with Neural Networks”. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS’14. Montreal, Canada: MIT Press, 2014, pp. 3104–3112.
- [131] C. Raffel et al. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer”. In: *J. Mach. Learn. Res.* 21.1 (June 2022). ISSN: 1532-4435.
- [132] B. Bi et al. “PALM: Pre-training an Autoencoding&autoregressive Language Model for Context-conditioned Generation”. In: *Conference on Empirical Methods in Natural Language Processing*. 2020.

- [133] OpenAI. *ChatGPT: Optimizing Language Models for Dialogue*. Nov. 2022. URL: <https://openai.com/blog/chatgpt/>.
- [134] E. Qian. “Risk Parity and Diversification”. In: *The Journal of Investing* 20.1 (2011), pp. 119–127. ISSN: 1068-0896. DOI: [10.3905/joi.2011.20.1.119](https://doi.org/10.3905/joi.2011.20.1.119).
- [135] M. López de Prado. “Building Diversified Portfolios that Outperform Out of Sample”. In: *The Journal of Portfolio Management* 42.4 (2016), pp. 59–69. ISSN: 0095-4918. DOI: [10.3905/jpm.2016.42.4.059](https://doi.org/10.3905/jpm.2016.42.4.059).
- [136] J. Tenreiro Machado, F. Duarte, and G. Duarte. “Analysis of financial data series using fractional Fourier transform and multidimensional scaling”. In: *Nonlinear Dynamics* 65 (Aug. 2011), pp. 235–245. DOI: [10.1007/s11071-010-9885-1](https://doi.org/10.1007/s11071-010-9885-1).
- [137] J. Tan. *Principal Component Analysis and Portfolio Optimization*. SSRN Working Paper. 2012. DOI: [10.2139/ssrn.2213687](https://doi.org/10.2139/ssrn.2213687). URL: <https://ssrn.com/abstract=2213687>.
- [138] R. B. Litterman and J. Scheinkman. “Common Factors Affecting Bond Returns”. In: *The Journal of Fixed Income* 1.1 (1991), pp. 54–61. ISSN: 1059-8596. DOI: [10.3905/jfi.1991.692347](https://doi.org/10.3905/jfi.1991.692347).
- [139] G. Zumbach. “A Gentle Introduction to the RM2006 Methodology”. In: *SSRN Electronic Journal* (Jan. 2007). DOI: [10.2139/ssrn.1420183](https://doi.org/10.2139/ssrn.1420183).
- [140] O. Ledoit and M. Wolf. “Improved estimation of the covariance matrix of stock returns with an application to portfolio selection”. In: *Journal of Empirical Finance* 10.5 (2003), pp. 603–621. ISSN: 0927-5398. DOI: [https://doi.org/10.1016/S0927-5398\(03\)00007-0](https://doi.org/10.1016/S0927-5398(03)00007-0).
- [141] R. F. Engle, S. M. Focardi, and F. J. Fabozzi. “ARCH/GARCH Models in Applied Financial Econometrics”. In: *Encyclopedia of Financial Models*. John Wiley & Sons, Ltd, 2012. ISBN: 9781118182635. DOI: <https://doi.org/10.1002/9781118182635.efm0062>.
- [142] P. Fiszeder and W. Orzeszko. “Covariance matrix forecasting using support vector regression”. In: *Applied Intelligence* 51 (2021). DOI: [10.1007/s10489-021-02217-5](https://doi.org/10.1007/s10489-021-02217-5).
- [143] J. Sun, Y. Jiang, and J. Lin. “Convolutional LSTM Network for forecasting correlations between stocks based on spatiotemporal sequence”. In: *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*. 2021, pp. 1–6. DOI: [10.1109/INDIN45523.2021.9557538](https://doi.org/10.1109/INDIN45523.2021.9557538).
- [144] S. Bai, J. Z. Kolter, and V. Koltun. *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. arXiv preprint arXiv:1803.01271. 2018. DOI: [10.48550/arXiv.1803.01271](https://doi.org/10.48550/arXiv.1803.01271). URL: <https://arxiv.org/abs/1803.01271>.
- [145] X. Wang et al. *Stock2Vec: A Hybrid Deep Learning Framework for Stock Market Prediction with Representation Learning and Temporal Convolutional Network*. arXiv preprint arXiv:2010.01197. 2020. DOI: [10.48550/arXiv.2010.01197](https://doi.org/10.48550/arXiv.2010.01197). URL: <https://arxiv.org/abs/2010.01197>.
- [146] P. Chatigny, J.-M. Patenaude, and S. Wang. *Spatiotemporal Adaptive Neural Network for Long-term Forecasting of Financial Time Series*. arXiv preprint arXiv:2003.12194. 2020. DOI: [10.48550/arXiv.2003.12194](https://doi.org/10.48550/arXiv.2003.12194). URL: <https://arxiv.org/abs/2003.12194>.
- [147] Z. Yue et al. “TS2Vec: Towards Universal Representation of Time Series”. In: *AAAI Conference on Artificial Intelligence*. 2021.

- [148] S. Mohammadi and A. Nazemi. “On portfolio management with value at risk and uncertain returns via an artificial neural network scheme”. In: *Cognitive Systems Research* 59 (Sept. 2019). DOI: [10.1016/j.cogsys.2019.09.024](https://doi.org/10.1016/j.cogsys.2019.09.024).
- [149] Y. Deng et al. “Deep Direct Reinforcement Learning for Financial Signal Representation and Trading”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28.3 (2017), pp. 653–664. DOI: [10.1109/TNNLS.2016.2522401](https://doi.org/10.1109/TNNLS.2016.2522401).
- [150] S. Makridakis, E. Spiliotis, and V. Assimakopoulos. “Statistical and Machine Learning forecasting methods: Concerns and ways forward”. In: *PLOS ONE* 13.3 (Mar. 2018), pp. 1–26. DOI: [10.1371/journal.pone.0194889](https://doi.org/10.1371/journal.pone.0194889).
- [151] J. J. Hopfield. “Neural networks and physical systems with emergent collective computational abilities”. In: *Proceedings of the National Academy of Sciences* 79.8 (1982), pp. 2554–2558. DOI: [10.1073/pnas.79.8.2554](https://doi.org/10.1073/pnas.79.8.2554).
- [152] Q. Wen et al. *Transformers in Time Series: A Survey*. 2022. DOI: [10.48550/ARXIV.2202.07125](https://doi.org/10.48550/ARXIV.2202.07125). URL: <https://arxiv.org/abs/2202.07125>.
- [153] Compustat. *S&P Daily Index Price Data*. Data retrieved from Wharton Research Data Services, <https://wrds-web.wharton.upenn.edu/wrds/>. 2020.
- [154] J. Bromley et al. “Signature Verification Using a "Siamese" Time Delay Neural Network”. In: *Proceedings of the 6th International Conference on Neural Information Processing Systems*. NIPS’93. Denver, Colorado: Morgan Kaufmann Publishers Inc., 1993, pp. 737–744.
- [155] R. Karimi Mahabadi et al. “Prompt-free and Efficient Few-shot Learning with Language Models”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3638–3652. DOI: [10.18653/v1/2022.acl-long.254](https://doi.org/10.18653/v1/2022.acl-long.254). URL: <https://aclanthology.org/2022.acl-long.254>.
- [156] R. A. Martin. “PyPortfolioOpt: portfolio optimization in Python”. In: *Journal of Open Source Software* 6.61 (2021), p. 3066. DOI: [10.21105/joss.03066](https://doi.org/10.21105/joss.03066).
- [157] J. Pennington, R. Socher, and C. D. Manning. “GloVe: Global Vectors for Word Representation”. In: *EMNLP*. 2014.
- [158] M. Iyyer et al. “Deep Unordered Composition Rivals Syntactic Methods for Text Classification”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for Computational Linguistics, July 2015, pp. 1681–1691. DOI: [10.3115/v1/P15-1162](https://doi.org/10.3115/v1/P15-1162). URL: <https://aclanthology.org/P15-1162>.
- [159] T. Wolf et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: [10.18653/v1/2020.emnlp-demos.6](https://doi.org/10.18653/v1/2020.emnlp-demos.6). URL: <https://aclanthology.org/2020.emnlp-demos.6>.

- [160] J. Pfitzinger and N. Katzke. *A constrained hierarchical risk parity algorithm with cluster-based capital allocation*. Tech. rep. 14/2019. Stellenbosch University, Department of Economics, 2019.
- [161] P. Jain and S. Jain. “Can Machine Learning-Based Portfolios Outperform Traditional Risk-Based Portfolios? The Need to Account for Covariance Misspecification”. In: *Risks* 7.3 (2019). ISSN: 2227-9091. DOI: [10.3390/risks7030074](https://doi.org/10.3390/risks7030074).
- [162] J. Hasanhodzic and A. Lo. “Can Hedge-Fund Returns Be Replicated?: The Linear Case”. In: *Journal of Investment Management* 5.2 (Aug. 2007), pp. 5–45. DOI: [10.2139/ssrn.924565](https://doi.org/10.2139/ssrn.924565).
- [163] J. Chen and M. L. Tindall. “Hedge Fund Replication Using Shrinkage Methodologies”. In: *The Journal of Alternative Investments* 17.2 (2014), pp. 26–49. ISSN: 1520-3255. DOI: [10.3905/jai.2014.17.2.026](https://doi.org/10.3905/jai.2014.17.2.026).
- [164] G. Naitzat, A. Zhitnikov, and L.-H. Lim. *Topology of deep neural networks*. 2020. arXiv: [2004.06093](https://arxiv.org/abs/2004.06093) [cs.LG].
- [165] T. Hofmann, B. Schölkopf, and A. J. Smola. “Kernel Methods in Machine Learning”. In: *The Annals of Statistics* 36.3 (2008), pp. 1171–1220. DOI: [10.1214/009053607000000677](https://doi.org/10.1214/009053607000000677). URL: <https://doi.org/10.1214/009053607000000677>.
- [166] G. Peyre and M. Cuturi. “Computational Optimal Transport”. In: *Foundations and Trends in Machine Learning* 11.5-6 (2019), pp. 355–607.
- [167] D. Wang, Y. Yang, and S. Ning. “DeepSTCL: A Deep Spatio-temporal ConvLSTM for Travel Demand Prediction”. In: *2018 International Joint Conference on Neural Networks (IJCNN)*. 2018, pp. 1–8. DOI: [10.1109/IJCNN.2018.8489530](https://doi.org/10.1109/IJCNN.2018.8489530).
- [168] J. Alammar. *The Illustrated Transformer*. 2018. URL: <http://jalammar.github.io/illustrated-transformer/>.