Summer Associate Internship Exercises

Project Repository - github.com/tavisshore/JPM

Work Division

We shared the work equally – Tavis owned Question 1 plus package setup; Sanaa owned Question 3 plus meeting coordination. Each of us wrote the corresponding report sections for our question.

Contents

Part I
Question 1 - Part 1: Financial Statement Forecasting

Balance sheet forecasting is fundamental to corporate valuation, credit risk assessment, and strategic planning. Accurate projections of a firm's asset base, capital structure, and equity position enable analysts to evaluate solvency, estimate future cash flows, and assess financial flexibility under varying scenarios. However, unreliable balance sheet estimates propagate errors throughout downstream analyses, undermining investment decisions and risk management.

A central challenge in balance sheet forecasting is that line items are not independent: the accounting identity $A = L + E$ imposes a hard constraint, and changes in one component must be offset elsewhere to maintain consistency. Forecasting items in isolation inevitably produces contradictory statements that violate fundamental accounting relationships, rendering the projections unusable.

Identity-preserving models address this by explicitly enforcing accounting constraints during forecast generation. By ensuring that $A = L + E$ holds at each forecasted time step, these models guarantee internally consistent, economically interpretable projections suitable for quantitative analysis.

We begin by implementing Vélez-Pareja's plug-free framework [1], a deterministic model that eliminates circular dependencies and reveals causal relationships between financial decisions and outcomes. Building on insights from this implementation, we then develop LSTM-based time-series models that forecast balance sheets and income statements from historical data, with optional architectural mechanisms to enforce the accounting identity during training.

1 Literature Review - Modelling & Forecasting

1.1 Classical Approaches

Vélez-Pareja [1] proposes a forecasting framework built on double-entry logic, eliminating plugs and circularity by deriving statements from explicit cash-flow and financing schedules. This produces transparent, internally coherent projections and exposes modeling inconsistencies. Their subsequent work [2] significantly extends this framework, incorporating inflation adjustments, market dynamics, equity financing, and additional macroeconomic factors to handle more complex valuation scenarios. Jalbert [3] likewise develops a plug-free, internally consistent model for forecasting small-business financial statements, integrating budgeting, valuation, and ratios in a unified structure that reduces errors and simplifies interpretation. Arnold and Moon [4] analyse how pro-forma forecasts depend on plug or slack variables, showing these choices are mathematically linked. Using cash as the plug/slack term, they derive a growth rate that prevents cash depletion and clarifies the connection between balance-sheet consistency, growth, and financing needs.

1.2 Time-Series & ML

Amel-Zadeh et al. [5] show that ML models can forecast both the direction and magnitude of abnormal stock returns around earnings announcements using only past financial-statement data. It finds that Random Forests perform best overall, while RNNs outperform linear methods when predicting extreme market reactions, delivering steadier performance during downturns. Overall, the study demonstrates that ML models extract meaningful economic structure from accounting variables and can generate sizeable abnormal returns in backtests. Chen et al. [6] use ML models applied to thousands of financial items to predict the direction of future earnings changes, addressing the limitations of small-variable-set regressions. Their models outperform conventional logistic frameworks and professional analysts, achieving significantly higher AUCs and economically meaningful hedge-portfolio returns. The study highlights the value of granular financial disclosures and non-linear interactions in enhancing earnings-forecasting accuracy. Geertsema et al. [7] proposes a chained machine-learning framework that forecasts the full set of financial statements by predicting items sequentially in an accounting-consistent order, allowing information to flow across line items. This structure reduces out-of-sample errors by roughly one-third relative to parallel models and

proves especially effective for volatile firms and items late in the chain. The authors also show that deviations from these chained predictions provide incremental power in detecting financial irregularities, notably subtle little-r restatements.

2 Problem Definition & Mathematical Model

2.1 Balance Sheet Structure

A balance sheet summarises a company's financial position at a given time. It is composed of three main fields: Assets (A), Liabilities (L), and Equity (E). Assets represent the resources the firm controls, such as cash, inventory, and property. Liabilities capture obligations to external parties, including debt and accounts payable. Equity reflects the residual interest of shareholders once liabilities have been deducted from assets.

2.2 Key Principles from Vélez-Pareja

Mathematical evolution of balance sheet fields. Vélez-Pareja [1], [2] constructs forecasted financial statements strictly through the double-entry principle. Balance Sheet (BS) items are derived from the Cash Budget (CB), the Income Statement (IS), and operational schedules, never from residual *plugs*. The fundamental identity is $\text{Assets}_t = \text{Liabilities}_t + \text{Equity}_t$. The following equations govern how the balance sheet is built by Vélez-Pareja in [2], given by the equations in the final columns of each table within the papers.
Cash and short-term investments:

$$\text{Cash}_t = \text{NCB}_t^{\text{cum}} = \text{MinCash}_t,$$

$$\text{STInv}_t = \begin{cases} \text{NCB}_{t-1}^{\text{cum}} + \text{NCB}_t^{\text{after}} + \text{Inflow}_t^{\text{ST}} - \text{MinCash}_t & \text{if no deficit} \\ 0 & \text{otherwise} \end{cases}$$

Working capital (policy-driven):

$$\text{AR}_t = \text{Sales}_t \times \rho^{\text{AR}},$$
$$\text{AP}_t = \text{Purchases}_t \times \rho^{\text{AP}},$$
$$\text{APP}_t = \text{Purchases}_{t+1} \times \rho^{\text{adv,sup}},$$
$$\text{APR}_t = \text{Sales}_{t+1} \times \rho^{\text{adv,cust}}.$$

where ρ values are extracted from the *Policy and Goal Formulation* table (1b).
Inventory:

$$\text{Inventory}_t = \text{Inventory}_{t-1} + \text{Purchases}_t - \text{COGS}_t.$$

Net fixed assets:

$$\text{NFA}_t = \text{NFA}_{t-1} + \text{Capex}_t - \text{Dep}_t.$$

Short-term debt (operational deficit):

$$\text{STDebt}_t = \begin{cases} -\left(\text{NCB}_{t-1}^{\text{cum}} + \text{NCB}_t^{\text{op}} - \text{STPayment}_t - \text{MinCash}_t\right) & \text{if deficit} \\ 0 & \text{otherwise} \end{cases}$$

Long-term debt (investment deficit):

$$\text{LTDebt}_t = \text{LTDebt}_{t-1} + \text{NewLTDebt}_t - \text{Principal}_t^{\text{LT}},$$

$$\text{NewLTDebt}_t = \begin{cases} -\text{Deficit}_t^{\text{inv}} \times \lambda^{\text{debt}} & \text{if deficit} \\ 0 & \text{otherwise} \end{cases}$$

Interest:

$$\text{Interest}_t = (\text{STDebt}_{t-1} + \text{LTDebt}_{t-1}) \times K_d^t.$$

Equity investment:

$$\text{EquityInv}_t = \text{EquityInv}_{t-1} + \text{NewEquity}_t,$$

$$\text{NewEquity}_t = \begin{cases} -\text{Deficit}_t^{\text{inv}} \times (1 - \lambda^{\text{debt}}) & \text{if deficit} \\ 0 & \text{otherwise} \end{cases}$$

Retained earnings:

$$\text{RetEarn}_t = \text{RetEarn}_{t-1} + \text{NI}_{t-1} - \text{Div}_t,$$

$$\text{Div}_t = \text{NI}_{t-1} \times \rho^{\text{payout}}.$$

Net income:

$$\text{NI}_t = \text{EBIT}_t - \text{Interest}_t - \text{Tax}_t,$$

$$\text{Tax}_t = \begin{cases} \max\left(\min(\text{EBIT}_t, \text{Interest}_t), 0\right) \times \tau & \text{if EBT}_t > 0 \\ 0 & \text{otherwise} \end{cases}$$

Total equity:

$$\text{Equity}_t = \text{EquityInv}_t + \text{RetEarn}_t + \text{NI}_t - \text{StockRepurch}_t.$$

Recursive evolution:

$$\text{CB}_t \to \text{IS}_{t+1} \to \text{CB}_{t+1} \to \text{IS}_{t+2} \ldots \to \text{BS}_T \tag{1}$$

Modelling as a time-series. Balance sheets can be modelled as multivariate time series by treating each line item as a feature in a temporal sequence. Let $X_t \in \mathbb{R}^d$ denote the balance sheet at time t, where d is the number of features. The historical sequence $\{X_1, X_2, \ldots, X_T\}$ observed at reporting dates forms a multivariate time series suitable for sequential forecasting. Recurrent architectures such as LSTMs and GRUs can model the temporal dynamics by learning a mapping $f : \{X_{t-k}, \ldots, X_t\} \to X_{t+1}$ over a lookback window of k periods, capturing both inter-temporal dependencies and cross-feature relationships through hidden states h_t.

However, balance sheets are governed by the previously mentioned identities, meaning that naïve, independent forecasting will generally produce inconsistent statements. A coherent approach therefore requires either explicit constraints or loss penalties to enforce these identities, along with structural relationships reflecting working-capital cycles, depreciation, amortisation, and financing flows. Under this formulation, the balance sheet becomes a constrained multivariate time series whose evolution can be predicted while preserving accounting logic.

3 Proposed Methods

Model 1: Implementation of Vélez-Pareja

To develop a solid understanding of balance sheet dynamics, we first implemented Vélez-Pareja's plug-free forecasting model [1] in Python, reproducing its deterministic accounting structure without circular dependencies. This involved creating structured classes for assets, liabilities, equity accounts, and the operational and financing flows that update them, along with the logic for loans, equity issuance, repayments, and investment returns. The programmatic implementation made causal links between decisions and financial outcomes explicit, ensuring transactions propagate correctly through accounting identities and revealing inconsistencies that spreadsheet approaches often obscure.

When run with the same inputs as the original paper, our implementation exactly reproduces their results, validating our encoding of the plug-free framework. These values are shown in Appendix A; slight discrepancies after a few years arise from floating-point precision and rounding errors.

However, while theoretically elegant, the Vélez-Pareja model is not suitable for application with publicly available financial data. The framework assumes sequential availability of financial statements following Equation 1, but in practice all statements are released simultaneously at each reporting date, eliminating any informational advantage from modelling their intra-period relationships. Furthermore, the model requires detailed loan schedules, debt covenants, and operational decision variables that are not disclosed in SEC filings. These data limitations make the deterministic approach impractical for forecasting real-world balance sheets.

These limitations motivate a data-driven approach that operates directly on observable financial statement data without requiring unobservable inputs or sequential intra-period dependencies. From this point forward, we develop an LSTM-based forecasting model implemented in TensorFlow that learns temporal patterns from historical financial statements, with a probabilistic variant using mixture density networks to capture forecast uncertainty.

Model 2: ML Time-Series Forecasting

Problem Definition. The objective is to forecast future balance sheet, income statement, and cash flow line items from historical quarterly data while maintaining consistency with underlying accounting identities.

Model Architecture We employ LSTM networks to exploit the sequential structure of quarterly financial data. LSTMs capture long-range temporal dependencies through their gating mechanisms, enabling the model to learn both gradual trends and short-term fluctuations in financial metrics. The hidden state h_t encodes relevant historical information across multiple quarters, while the cell state c_t provides a pathway for gradients to flow across long sequences without vanishing.

LSTM Variants. We implement two variants: a deterministic LSTM producing point forecasts, and a probabilistic LSTM that outputs predictive distributions. The deterministic variant minimises mean squared error, yielding stable single-point estimates that are straightforward to evaluate against ground truth - shown in results tables below. However, it cannot quantify forecast uncertainty, limiting its utility when financial data exhibits volatility or structural breaks.

The probabilistic variant parameterises a predictive distribution $p(X_{t+1}|X_{\leq t})$ as a multivariate Gaussian with learned mean vector and covariance structure via Cholesky decomposition. This approach explicitly models forecast uncertainty and inter-variable dependencies inherent in financial statements, where accounting identities create structured correlations between line items. While this increases architectural complexity and training difficulty compared to point estimates, it provides principled uncertainty quantification suitable for the smooth, heavily processed nature of quarterly financial data.

3.1 Data Preparation

We initially used *yfinance* [8] to retrieve financial statements, caching results in parquet format. However, this API only returns approximately five periods of data (depending on what Yahoo Finance itself provides) regardless of whether quarterly or annual data is requested, making it unsuitable for training robust time-series models. We therefore switched to *EdgarTools* [9], which extracts XBRL-tagged statements directly from SEC EDGAR filings (10-K and 10-Q forms). This provides substantially longer historical coverage and preserves the exact figures companies report, ensuring consistency with regulatory disclosures.

Schema Normalisation. A major challenge with SEC data is the lack of standardised field naming conventions: constituent line items vary dramatically across companies, and even individual firms change their terminology over time. To process filings from multiple companies into a unified format, we constructed a target schema defining a consistent set of features across all statements. For each downloaded filing, we prompt an LLM to map the company's reported fields to our standardised schema, handling aggregations where necessary to avoid double-counting downstream.

Validation and Quality Control. After reconstructing each statement, we perform integrity checks: verifying that constituent items sum correctly to subtotals (e.g., current assets), and confirming that the fundamental accounting identity $A = L + E$ holds. Since 10-Q filings report year-to-date cumulative values, we compute quarterly figures by subtracting previous quarters' cumulative totals where appropriate.

Feature Engineering and Selection. Derived columns (e.g., financial ratios, aggregated metrics) were excluded from model inputs and outputs, as these features are linear combinations of base line items that introduce multicollinearity and increase dimensionality without adding predictive signal beyond their constituent base features. We work directly with constituent line items from the balance sheet, income statement, and cash flow statement. Since all three statements influence future balance sheet positions and

earnings through their underlying accounting relationships, this multi-statement approach captures fundamental economic drivers without redundancy. Prior to model training, we apply systematic feature selection while preserving critical balance sheet items and net income essential for prediction: remove non-numeric columns, eliminate low-variance features, remove highly correlated columns to mitigate multicollinearity, and identify and remove near-duplicate columns using cosine similarity thresholds. Derived metrics are computed post-hoc from the model's base feature predictions, isolating forecast errors to their source components.

3.2 Training

We train the model by sampling sequential windows from the dataset – for example, using the previous five quarters to predict the next one. The *lookback window* determines how much historical context the model can exploit: shorter windows emphasize recent movements, while longer ones capture slower structural trends but introduce more noise and complexity. The *target horizon* specifies how far ahead the model forecasts; we focus on a one-step horizon, predicting the next quarter's financial statements directly. We performed a grid search over training configurations, selecting the model that achieved the lowest validation MAE.

3.3 Composite Loss Function

To ensure forecasts are both accurate and internally consistent, we use a composite loss combining prediction error and an accounting identity penalty. Let $\hat{y}_t \in \mathbb{R}^d$ denote the predicted items for quarter t in standardised space, and y_t the ground truth. The baseline loss is mean-squared error:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{d} \sum_{i=1}^{d} (y_{t,i} - \hat{y}_{t,i})^2 . \tag{2}$$

To evaluate consistency with the accounting identity, predictions are first unscaled to obtain $\hat{x}_t \in \mathbb{R}^d$ via

$$\hat{x}_t = \hat{y}_t \odot \sigma + \mu, \tag{3}$$

where μ and σ are feature-wise means and standard deviations, and $\odot$ denotes element-wise multiplication.

The unscaled predictions are aggregated into assets, liabilities, and equity:

$$\hat{A}_t = \sum_{i \in \mathcal{A}} \hat{x}_{t,i}, \qquad \hat{L}_t = \sum_{i \in \mathcal{L}} \hat{x}_{t,i}, \qquad \hat{E}_t = \sum_{i \in \mathcal{E}} \hat{x}_{t,i}, \tag{4}$$

where $\mathcal{A}$, $\mathcal{L}$, and $\mathcal{E}$ index asset, liability, and equity components.

The violation of the identity $A_t = L_t + E_t$ is measured through relative error:

$$\text{rel\_err}_t = \frac{\hat{A}_t - (\hat{L}_t + \hat{E}_t)}{|\hat{A}_t| + |\hat{L}_t| + |\hat{E}_t| + \varepsilon}, \tag{5}$$

where ε is a numerical stabiliser. The identity penalty is then

$$\mathcal{L}_{\text{id}} = \mathbb{E}\big[\text{rel\_err}_t^2\big] . \tag{6}$$

The final training objective is

$$\mathcal{L} = \mathcal{L}_{\text{MSE}} + \lambda \, \mathcal{L}_{\text{id}}, \tag{7}$$

where λ controls the weight of the identity penalty.

3.4 Identity Enforcement Layer

Beyond the loss penalty, we use a deterministic post-processing layer that projects network outputs onto the manifold satisfying the accounting identity. This layer operates on the unscaled predictions $\hat{x}_t$ obtained via (3) and reuses the aggregates from (4).

We designate a single equity component $\hat{s}_t = \hat{x}_{t,j}$ (for $j \in \mathcal{E}$) as a slack variable. The identity discrepancy is

$$d_t = \hat{A}_t - (\hat{L}_t + \hat{E}_t). \tag{8}$$

Only the slack component is adjusted:

$$\tilde{s}_t = \hat{s}_t + d_t, \tag{9}$$

yielding a corrected vector $\tilde{x}_t$ that matches $\hat{x}_t$ except at the slack index. From (8), the corrected aggregates satisfy

$$\tilde{A}_t = \hat{A}_t, \qquad \tilde{L}_t = \hat{L}_t, \qquad \tilde{E}_t = \hat{E}_t + d_t,$$

enforcing exact identity:

$$\tilde{A}_t - (\tilde{L}_t + \tilde{E}_t) = 0.$$

The corrected vector is re-standardised:

$$\tilde{y}_t = \frac{\tilde{x}_t - \mu}{\sigma},$$

and passed downstream. This differentiable projection ensures every predicted balance sheet satisfies $A_t = L_t + E_t$ exactly. This approach guarantees valid outputs but concentrates all corrections in the slack variable, which can mask modelling errors and distort that line item. However, this deterministic projection is incompatible with the probabilistic LSTM, which outputs distribution parameters rather than point estimates, precluding direct algebraic adjustment of individual balance sheet components.

3.5 Seasonality Weighting

Financial data exhibits systematic quarterly patterns due to business cycles, tax calendars, and reporting conventions. To exploit this structure, we up-weight observations from the same quarter in previous years when constructing the lookback window. This seasonal weighting scheme helps the model distinguish persistent trends from predictable year-on-year cycles, improving forecast accuracy for firms with strong seasonal dynamics.

We implement this by applying a multiplicative weight $w > 1$ to feature vectors at seasonal indices (i.e., observations exactly 4, 8, 12, etc. quarters prior), producing a modified input sequence $\tilde{X}_t = W \odot X_t$ where W is a time-dependent weight matrix. Grid search over the validation set identified an optimal seasonal weight of $w = 1.11$, corresponding to an 11% amplification of year-on-year signals without distorting non-seasonal time steps.

4 Evaluation

To evaluate whether the LSTM captures meaningful temporal patterns in financial statements, we benchmark it against non-learned baselines: *last observation carry-forward* (repeating the most recent quarter), *historical mean*, and *seasonal naïve* (reusing the value from exactly four quarters prior).

We compute mean absolute error (MAE) for each baseline and report a skill score defined as $1 - \frac{\text{MAE}_{\text{model}}}{\text{MAE}_{\text{baseline}}}$, indicating relative improvement over the reference method. A skill score of zero indicates performance equal to the baseline, positive values indicate improvement, and negative values indicate degradation.

Model Configuration. Grid search over the validation set yielded the following optimal hyperparameters: 2 LSTM layers with 368 units each, a dense layer with 256 units, dropout probability $p = 0.2$ applied during inference, learning rate $\alpha = 10^{-4}$, trained for 500 epochs with identity constraint weight $\lambda = 10^{-4}$. We partition data chronologically, holding out the final quarters of each firm's series as the test set to ensure strict temporal separation.

Results. Table 1 reports performance across 8 companies for both balance sheet and net income forecasting. For balance sheet prediction, the LSTM achieves MAE of \$6.50bn $\pm$ \$6.24bn, outperforming all baselines by at least 20%. The model demonstrates consistent improvement over simple heuristics, though absolute error magnitudes remain substantial relative to typical balance sheet scales, indicating room for further refinement. For net income prediction, the LSTM achieves \$7.25bn $\pm$ \$3.80bn MAE, again outperforming the baselines. The tighter margins relative to baselines suggest that earnings exhibit more volatile temporal dynamics that are harder to exploit with recurrent architectures alone. Lower standard deviation in LSTM predictions (\$3.80bn vs. \$5.95bn for global mean) indicates more stable forecasts across firms. While the LSTM demonstrates measurable skill over non-learned baselines, absolute performance

remains insufficient for production deployment without further architectural improvements or incorporation of external signals beyond historical financials.

Ablation Study. We conduct an ablation analysis to isolate the contributions of each architectural component: the identity constraint penalty, seasonality weighting, and the hard enforcement layer. Table 2 reports results for Apple Inc. (AAPL) across four model variants. The baseline model trained with MAE loss alone achieves \$3.7bn balance sheet error and 0.69% identity violation, with 12.56% net income error. Adding the identity penalty term reduces identity error to 0.62% without degrading balance sheet MAE, while marginally improving net income prediction to 11.54%. Incorporating seasonality weighting maintains balance sheet performance but increases identity error slightly to 0.88%, while substantially reducing net income error to 5.64%, confirming that earnings exhibit stronger seasonal patterns than balance sheet items. The full model with hard identity enforcement achieves perfect accounting identity satisfaction (0.00% error) while improving balance sheet MAE to \$3.4bn and dramatically reducing net income error to 0.77%. This demonstrates that architectural enforcement of $A = L + E$ not only guarantees constraint satisfaction but also improves predictive accuracy by regularising the model's hypothesis space to economically plausible trajectories.

Method	MAE ($\mu \pm \sigma$)	Error diff
LSTM	\$6.50bn ± \$6.24bn	–
Global Mean	\$9.94bn ± \$7.28bn	52.9%
Last Value	\$8.15bn ± \$6.68bn	25.4%
Seasonal Naive	\$8.02bn ± \$6.49bn	23.4%

Balance Sheet

Method	MAE ($\mu \pm \sigma$)
LSTM	\$7.25bn ± \$3.80bn
Global Mean	\$9.77bn ± \$5.95bn
Last Value	\$7.65bn ± \$4.97bn
Seasonal Naive	\$7.83bn ± \$6.04bn

Net Income

Table 1: Baseline Comparisons Across 8 Companies

	Balance Sheet		Net Income
Model Variant	MAE	ID Error	Error
Baseline (MAE only)	\$3.7bn	0.69%	12.56%
MAE + Identity Penalty	\$3.7bn	0.62%	11.54%
MAE + ID + Seasonality	\$3.7bn	0.88%	5.64%
MAE + ID + S + Enforcement	\$3.4bn	0.00%	0.77%

Table 2: Ablation Example - AAPL

5 Conclusion

The LSTM-based forecasting model demonstrates measurable improvement over naïve baselines for both balance sheet and net income prediction. The best-performing configuration achieves balance sheet MAE of \$3.4bn (approximately 1% relative error) and net income prediction error of 0.77%, substantially outperforming last-value, seasonal naïve, and global mean baselines.

The ablation study (Table 2) reveals the contribution of each architectural component. The identity penalty reduces accounting constraint violation from 0.69% to 0.62% without degrading MAE, demonstrating that soft regularisation improves consistency. Adding seasonality weighting dramatically improves net income accuracy (12.56% $\rightarrow$ 5.64%) but increases identity error to 0.88%, revealing tension between capturing seasonal earnings patterns and maintaining strict balance sheet consistency. The hard identity enforcement layer eliminates constraint violations entirely (0.00% error) while simultaneously improving both balance sheet MAE (\$3.4bn) and net income prediction (0.77%). This demonstrates that architectural enforcement of $A = L + E$ acts as a powerful regulariser, constraining the hypothesis space to economically plausible trajectories and reducing over-fitting.

However, absolute performance remains insufficient for production deployment. The model operates exclusively on historical financial statement data, ignoring fundamental drivers of firm value such as macroe-

conomic conditions, industry dynamics, management guidance, and forward-looking market signals. Incorporating external features – including GDP growth, interest rates, sector performance, and sentiment indicators – would substantially improve forecasting accuracy by capturing regime changes and business cycle effects that pure autoregressive models cannot detect.

Future Work Immediate priorities include enabling full evaluation of the probabilistic model, which is currently constrained by data quality issues. While EDGAR filings provide extensive historical coverage, line-item taxonomies and reporting conventions vary substantially across firms and evolve over time, preventing reliable multi-company training. We are developing an LLM-based preprocessing pipeline to ingest raw SEC filings and map heterogeneous disclosures into a unified schema, as described in Section **??**. With standardised data, the probabilistic LSTM can generate full predictive distributions over balance sheet items, quantify forecast uncertainty, and propagate variance through accounting identities—supporting risk-aware simulations and scenario analysis.

Beyond data standardisation, model performance would benefit from incorporating fundamental and macroeconomic features as auxiliary input channels. Variables such as sector-specific indicators, credit spreads, and forward guidance would enable the model to adapt forecasts across different economic regimes rather than relying solely on historical patterns. Additionally, extending the architecture to capture cross-sectional dependencies between firms in the same industry could exploit common factors and improve robustness.

Remaining Explicit Answers

Question 8d. $x(t)$ should include any extra information that affects what happens next but is not already contained in $y(t)$. It holds the outside factors the model needs in order to correctly work out $y(t+1)$. These additional inputs could include macroeconomic indicators, sector-level trends, or firm-specific events that influence financial outcomes but are not visible from past statements alone. Future work could expand this set by incorporating market-sentiment signals and richer fundamental data, potentially through a dedicated sentiment or macro-context model.

Part II
Question 1 - Part 2: LLM-Based Financial Statement Analysis and Forecasting

6 LLM Configuration

We employed the ChatGPT API for balance sheet forecasting and automated financial data extraction from annual reports. For initial development, we used **gpt-5-nano-2025-08-07** for its low cost and fast iteration speed when prototyping the forecasting pipeline. For PDF parsing, we implemented a pipeline using **gpt-4o-2024-08-06** for its superior vision and document understanding capabilities, converting annual reports to text using Python libraries, then sending structured prompts to the API requesting specific balance sheet and income statement line items. The extracted values are subsequently used to calculate financial ratios (quick ratio, debt-to-equity, interest coverage, etc.), enabling automated financial analysis across diverse reporting formats. Model outputs were evaluated using **gpt-5-mini-2025-08-07** for cost-effective quality assessment at scale. The final forecasting system achieved balance sheet MAE of \$4.16bn and net income MAE of \$5.24bn across 8 companies.

7 LLM vs LSTM Comparison

The ChatGPT API demonstrated superior predictive accuracy compared to the LSTM model across both forecasting tasks. For balance sheet prediction, the LLM achieved a MAE of \$4.16bn compared to the LSTM's \$6.50bn, representing a 36% improvement. Similarly, for net income forecasting, the LLM reduced MAE from \$7.25bn to \$5.24bn, a 28% reduction in error.

However, this performance advantage comes with significant trade-offs. The LSTM model requires substantial upfront investment in data preprocessing, architecture design, and hyperparameter tuning, but produces deterministic, reproducible predictions with full transparency into the learned temporal patterns. In contrast, the LLM operates as a black-box system with minimal setup requirements but lacks interpretability and raises concerns about training data contamination. Computational costs differ markedly: the LSTM incurs high one-time training expenses but negligible inference costs, while the LLM requires no training but imposes per-query API fees and potential latency issues. Most critically, the LLM's performance may be artificially inflated by data leakage if ground truth values were present in its training corpus or accessible via web search, undermining the validity of this comparison for true out-of-sample forecasting.

Method	MAE ($\mu \pm \sigma$)	Error diff
LSTM	\$6.50bn ± \$6.24bn	–
Global Mean	\$9.94bn ± \$7.28bn	52.9%
Last Value	\$8.15bn ± \$6.68bn	25.4%
Seasonal Naive	\$8.02bn ± \$6.49bn	23.4%
ChatGPT	\$4.16B ± \$3.74bn	56.3%

Table 3: Balance Sheet

Method	MAE ($\mu \pm \sigma$)
LSTM	\$7.25bn ± \$3.80bn
Global Mean	\$9.77bn ± \$5.95bn
Last Value	\$7.65bn ± \$4.97bn
Seasonal Naive	\$7.83bn ± \$6.04bn
ChatGPT	\$5.24bn ± \$3.76bn

Table 4: Net Income

Baseline Comparisons Across 8 Companies

8 Ensemble Model

To leverage the complementary strengths of both approaches, we constructed an ensemble model using a weighted combination:

$$\hat{y}_{\text{ensemble}} = \alpha \cdot \hat{y}_{\text{LSTM}} + (1 - \alpha) \cdot \hat{y}_{\text{LLM}} \tag{10}$$

where α was optimised to minimise validation error for each prediction task.

The ensemble approach yielded mixed results. For balance sheet forecasting, the adjusted ensemble achieved a MAE of \$3.81bn, outperforming both individual models with a 41.4% improvement over the LSTM baseline. This suggests that the LSTM's learned temporal dynamics provide valuable signal that complements the LLM's predictions, potentially mitigating some of the LLM's susceptibility to outliers or erratic forecasts. However, for net income prediction, the ensemble performed worse than the LLM alone (\$6.25bn vs \$5.24bn MAE), indicating that the LSTM predictions introduced noise rather than useful complementary information for this task. These divergent results highlight that ensemble benefits are task-dependent and require careful validation to determine when model combination provides genuine improvement over individual predictors.

Method	MAE ($\mu \pm \sigma$)	Error diff
LSTM	\$6.50bn ± \$6.24bn	–
ChatGPT	\$4.16B ± \$3.74bn	
Ensemble Adjust	\$3.81bn ± \$3.46bn	-41.4%

Table 5: Balance Sheet

Method	MAE ($\mu \pm \sigma$)
LSTM	\$7.25bn ± \$3.80bn
ChatGPT	\$5.24bn ± \$3.76bn
Ensemble Adjust	\$6.25bn ± \$6.34bn

Table 6: Net Income

Baseline Comparisons Across 8 Companies

9 Case Study: Apple Inc.

Based on our financial statement prediction model applied to Apple's latest 3 quarters, we forecast next quarter net income of \$80.8bn on revenue of \$174bn, yielding a 46.4% net profit margin that significantly exceeds typical technology sector margins of 10-20% [10]. The predicted balance sheet totals \$344bn in assets, with \$279bn liabilities and \$64.5bn equity, satisfying the accounting identity perfectly.

Key financial indicators demonstrate exceptional strength: $59bn in liquid assets provides 95% coverage of $61.9bn in accounts payable, enabling Apple to meet nearly all short-term obligations from immediate liquid resources. Total debt of $92.2bn represents only 26.8% of total assets, meaning 73% of assets are equity-financed –a conservative leverage profile that preserves substantial borrowing capacity.

We would advise the CEO that Apple maintains investment-grade credit metrics with substantial financial flexibility for strategic initiatives. The 46.4% operating margin, generates cash flow sufficient to sustain aggressive buybacks and dividends while maintaining a debt-to-assets ratio well below investment-grade thresholds. The company's ability to operate with negative retained earnings alongside 26.8% leverage demonstrates exceptional cash generation capacity that justifies the current capital allocation strategy of prioritising shareholder returns while preserving optionality for opportunistic M&A or strategic investments.

10 PDF Extraction Pipeline

We developed an automated pipeline to extract financial data from annual reports using a multi-stage LLM-based approach. The pipeline operates as follows:

Document Processing. Annual reports are parsed using PyPDF to convert pages to text, with selective page extraction focusing on financial statement sections to reduce processing overhead during development.

LLM Extraction. The text is processed by gpt-4o-2024-08-06, chosen for its superior document understanding and vision capabilities. The model receives a structured prompt that guides extraction through three steps: identifying the most recent fiscal year end date and converting it to ISO format, then extracting specific balance sheet, income statement, and cash flow statement data, and finally identifying the reporting currency. The prompt includes detailed mapping rules for both IFRS and US GAAP standards, handling variations in line item nomenclature across different accounting frameworks.

Structured Output. The model returns a JSON object containing only the raw extracted values with strict formatting requirements. This structured format ensures consistent parsing across diverse report formats.

Currency Normalisation. All extracted financial values are converted to USD using historical exchange rates from the ExchangeRate-API [11], queried at each report's fiscal year end date. This normalisation ensures all financial metrics are comparable when training the downstream XGBoost credit rating model, eliminating currency effects that would otherwise confound the learning process.

Ratio Calculation: Financial ratios are calculated programmatically from the extracted raw values rather than requesting the LLM to compute them directly. This approach proved significantly more robust and consistent, as it eliminates LLM calculation errors and ensures transparent, reproducible ratio derivation from verified source values. Table **??** shows a sample of calculated ratios and their ground truths for some major companies.

11 Robustness Analysis

To evaluate the reliability of the LLM-based extraction pipeline, we conducted multi-run consistency testing by processing the same annual reports 10 times with temperature set to 0 for deterministic outputs. The test corpus included a diverse set of companies spanning different industries and accounting standards: LVMH, Tencent, Alibaba, JP Morgan, ExxonMobil, Volkswagen, Microsoft, and Google. Results revealed that extraction consistency is heavily dependent on document structure and formatting.

For companies with well-structured, clearly labelled financial statements, extracted values remained perfectly consistent across all runs. However, for reports with complex layouts, nested tables, or ambiguous line item labels, certain figures exhibited minor variance of 1-2% between runs, while a small subset of poorly formatted documents produced highly unstable extractions. This analysis highlights that while the LLM approach generalises well across diverse reporting formats and accounting frameworks, extraction reliability is fundamentally constrained by source document quality and structural clarity.

Company	Source	Net Income ($bn)
General Motors	LLM	9.84
	GT	9.8 - 11.2
LVMH	LLM	13.0
	GT	-
Tencent	LLM	26.9
	GT	-
Alibaba	LLM	71.3
	GT	-
JP Morgan Chase	LLM	58.5
	GT	58.471
Exxon Mobil	LLM	36.0
	GT	36.010
Volkswagen	LLM	12.8
	GT	5.5
Microsoft	LLM	88.1
	GT	88.136
Google	LLM	100.0
	GT	100.118

Table 7: Net income extraction comparison between LLM and ground truth values.

Company	Cost-to-Income	Quick Ratio	Debt-to-Equity	Debt-to-Assets	Debt-to-Capital	Debt-to-EBITDA	Interest Coverage
General Motors	0.1230	0.9017	1.8813	0.4698	0.6529	7.4615	10.2064
LVMH	0.4395	0.7064	0.3311	0.1538	0.2488	0.9592	42.7760
Tencent	0.2259	1.2490	0.1892	0.1120	0.1591	0.7498	17.3691
Alibaba	0.2225	1.7257	0.1550	0.0968	0.1342	1.0291	14.2632
JPMC	0.5170	0.3606	1.3175	0.1135	0.5685	2.4636	0.7408
Exxon Mobil	0.1377	1.1648	0.1956	0.1105	0.1636	0.5597	62.1708
VW	0.1706	0.8686	1.2915	0.4015	0.5636	4.7260	5.5311
Microsoft	0.2512	1.2650	0.1923	0.1008	0.1613	0.3834	37.2855
Google	0.2608	1.8369	0.0406	0.0293	0.0390	0.1030	419.3657

Table 8: Financial ratios calculated from LLM-extracted values across diverse companies and industries.

For example, General Motors' annual report presents "Net Income" with a range value of "$9.8-11.2bn" at its first occurrence, illustrating the type of ambiguity that could be addressed through enhanced text extraction preprocessing or more explicit prompt constraints requiring single-point estimates rather than ranges. The extracted values are shown in Tables 7 and 8.

Part III

Question 1 - Part B Bonus 1: Credit Rating Forecasting

In the PDF extraction pipeline described above, we extract financial metrics and calculate key ratios using Equations 15-21. Given the tabular structure of quarterly financial data, we employ an XGBoost gradient boosting model to predict credit ratings from these engineered features, training on historical ratio-rating pairs. The training dataset construction proceeds as follows: First, we acquire historical credit ratings from ratingshistory.info, downloading Moody's rating histories for all companies in our dataset. When multiple rating types are available, we prioritise organisation-level ratings over instrument-specific ratings to ensure

consistent entity-level assessment. We then construct a time-indexed DataFrame aligned with the quarterly financial statement series, updating rating values at each Moody's rating action to create a continuous rating history. The feature matrix is generated by calculating the seven financial ratios (Quick Ratio, Debt-to-Equity, Debt-to-Assets, Debt-to-Capital, Debt-to-EBITDA, Interest Coverage, and Cost-to-Income) for each quarter using the extracted and normalised financial statement data. To create a predictive modelling framework, we perform a temporal shift: each quarter's ratio features are paired with the subsequent quarter's credit rating as the target variable, enabling the model to learn forward-looking rating prediction from current financial metrics. Credit ratings are ordinally encoded to preserve their inherent ranking structure during model training. The concatenation of ratio features and lagged ratings produces the final training dataset for XGBoost model estimation.

XGBoost Model. We employ XGBoost due to its suitability for ordinal targets and tabular financial data. Credit ratings are inherently ordered, the distance between Prime and High grade is conceptually similar to that between Medium and Low, making gradient boosting with ordinal encoding appropriate. XGBoost efficiently handles quarterly financial statements, captures non-linear relationships between ratios and credit risk, provides interpretable feature importance metrics, and models complex interactions between liquidity, leverage, and profitability without extensive feature engineering.

The mathematical form of an XGBoost model is an additive ensemble:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i) \tag{11}$$

where $\hat{y}_i$ is the predicted value for sample i, K is the number of trees, f_k is the k-th decision tree, and x_i is the feature vector for sample i.

Each tree is learned by minimising the regularised objective function:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{12}$$

where l is the loss function (e.g., squared error for regression, log loss for classification), $\hat{y}_i^{(t-1)}$ is the prediction from the first $t-1$ trees, f_t is the new tree being added, and the regularisation term is:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \tag{13}$$

where T is the number of leaves, w_j is the weight (prediction value) of leaf j, and γ, λ are regularisation hyper-parameters.

For our credit rating prediction task, we treat ordinal regression as multi-class classification with ordinal encoding. The loss is softmax cross-entropy, and the final prediction is:

$$\hat{y} = \underset{c}{\text{argmax}} \left(\text{softmax} \left(\sum_{k=1}^{K} f_k^{(c)}(x) \right) \right) \tag{14}$$

where $f_k^{(c)}$ represents the k-th tree's contribution to class c.

Training Dataset. The Moody's data contains a wide range of ratings, however it tends to contain more highly rated companies - with clear missing data regarding defaulted ones. Due to this major issue - training the XGBoost model is heavily limited. We reduce the number of classes from at least 10 to just 4 - Prime, High, Medium, Low. This simplifies learning for the XGBoost. The training dataset currently contains 500 samples: using 350 for training, and 75 for both validation and testing.

$$\text{Quick Ratio} = \frac{\text{Current Assets} - \text{Inventory}}{\text{Current Liabilities}} \quad (15)$$

$$\text{Debt-to-Equity} = \frac{D}{E} \quad (16)$$

$$\text{Debt-to-Assets} = \frac{D}{A} \quad (17)$$

$$\text{Debt-to-Capital} = \frac{D}{D+E} \quad (18)$$

$$\text{Debt-to-EBITDA} = \frac{D}{\text{EBITDA}} \quad (19)$$

$$\text{Interest Coverage} = \frac{\text{EBIT}}{I} \quad (20)$$

$$\text{Cost-to-Income} = \frac{C}{R} \quad (21)$$

where D = Total Debt, E = Total Equity, A = Total Assets, I = Interest Expense, C = Operating Expenses, R = Total Revenue

11.1 Model Evaluation

Feature importance is computed by measuring the average improvement in loss function attributable to each feature across all tree splits. The three most important features are Quick Ratio (strongly dominant), Interest Coverage (EBIT-to-Interest), and Debt-to-Assets, aligning with credit theory where liquidity, debt servicing capacity, and leverage are primary risk indicators.

The ordinal regression model with 4 credit rating classes (Prime, High, Medium, Low) and 12 features achieves 86.67% exact accuracy and perfect 100% within-1 accuracy, meaning no prediction misses by more than one rating category. Mean absolute error is 0.13 rating classes (rounded) and 0.22 (continuous). Class-level performance shows Prime (1.00/1.00 precision/recall, n=2), High (0.91/0.83), Medium (0.88/0.91, n=46), and Low (0.79/0.73) ratings. Weighted F1 score of 0.87 and macro-averaged metrics (0.89 precision, 0.87 recall) demonstrate consistent performance across rating categories without majority class bias.

Evergrande. We evaluate the model's performance on China Evergrande Group's 2022 annual report. Passing the report through our pipeline extracts key financial metrics and calculates ratios: Quick Ratio of 0.71 (indicating moderate liquidity stress), negative Debt-to-Equity of -1.02 (signalling negative equity), Debt-to-Assets of 0.33, Debt-to-Capital of 45.99, and negative Debt-to-EBITDA of -65.97 (reflecting negative earnings). The XGBoost model predicts a "Medium" credit rating (mapping to Ba2-Baa3 range) with 47.36% confidence – the low confidence suggesting model uncertainty. However, this prediction significantly underestimates Evergrande's default risk, as the company was already in severe financial distress by late 2022. This failure reflects a fundamental limitation in our training data: our (partial) S&P 500 dataset contains no actual default cases, with only a few ratings carrying *PD* (probability of default) suffixes representing the tail risk. Consequently, the model lacks exposure to the extreme financial deterioration patterns characteristic of imminent defaults, causing it to anchor predictions toward the middle of the rating distribution even when confronted with clear insolvency signals like negative equity and negative EBITDA.

We could validate LLM-extracted data through automated checks:

- Accounting Identity Verification: Our existing balance sheet validation could be integrated during parsing to reject malformed extractions immediately.
- Cross-Statement Consistency: Net income should reconcile with retained earnings changes; operating cash flow should align with net income adjusted for non-cash items.
- Ratio Reasonableness Bounds: Flag extractions producing impossible values (e.g., negative quick ratios when assets are positive, or out-of-range leverage metrics).

These checks would detect errors before they enter the training dataset, with existing balance sheet validation particularly suited for real-time parsing rejection.

Question 3

12 Part 1: Deep Context-Dependent Choice Models

12.1 Introduction

Discrete choice models describe how individuals choose among a finite set of alternatives. Classical models such as the Multinomial Logit (MNL) rely on strong assumptions, most notably the independence of irrelevant alternatives (IIA). In many applications, these assumptions are violated: the attractiveness of an alternative depends on which other options are present. Phenomena such as decoy, attraction, similarity, and compromise effects are well documented and constitute systematic departures from IIA.

Recent work uses neural architectures to relax these assumptions and explicitly model context-dependent utilities. In this section we study the Deep Context-Dependent Choice Model of Zhang et al. (2025)[12], often referred to as *DeepHalo*, and compare it with the Reproducing Kernel Hilbert Space (RKHS) choice model of Yang et al.[13]. Our goals are to

- clarify ambiguous or underspecified aspects of Zhang (2025);
- reimplement DeepHalo in `choice-learn` using TensorFlow / TensorFlow Probability and check it against the authors' PyTorch code;
- reproduce and extend the synthetic experiments (decoy, attraction, compromise);
- compare DeepHalo with the RKHS choice model of Yang (2025);
- assess the suitability of these models for credit-card offer demand estimation and discuss alternatives.

All implementations are integrated into the `choice-learn` framework, with a secondary PyTorch reproduction of the authors' code used for behavioral cross-validation.

13 Literature Review

13.1 Deep Context-Dependent Choice Model (Zhang, 2025)

Zhang (2025)[12] proposes a neural architecture designed to capture context effects through permutation-equivariant transformations on item embeddings. The core idea is to represent each item j in a choice set S by an embedding z_j and repeatedly update z_j using information aggregated from the other items in S. The resulting architecture (DeepHalo) has the following components:

- a base encoder mapping item IDs or features to dense embeddings;
- a stack of *halo blocks* that aggregate a context summary from the set and apply multi-head nonlinear updates to each item;
- residual connections and layer normalization for stable optimization;
- a final linear projection to utilities, followed by a masked softmax over available items.

Because all transformations are permutation-equivariant, the model naturally handles variable-size choice sets. IIA is broken by design, since each item's representation depends on the composition of the set. Zhang demonstrates via synthetic experiments that the model can express decoy, attraction, and compromise effects.

13.2 RKHS Choice Model (Yang, 2025)[13]

Yang (2025) introduces a choice model formulated in a vector-valued RKHS. Utilities take the form

$$u_j(S) \;=\; f(x_j) + g(x_j, S),$$

where x_j denotes item features, f captures context-independent effects, and g captures context-dependent interactions via kernels defined on items and sets. By choosing appropriate kernels and regularization, the model trades off flexibility and smoothness while retaining convex training objectives.

Key features include:

- a decomposition into baseline utilities and context interaction terms;
- the use of kernels to encode similarity and smoothness priors;
- a convex, regularized estimation problem for fixed kernels;
- improved interpretability relative to deep neural architectures.

The RKHS formulation can represent context effects but biases the solution toward smoother, kernel-structured interactions, which can be advantageous in data-scarce regimes or when interpretability is important.

13.3 High-Level Comparison

Both Zhang (2025) and Yang (2025) aim to move beyond IIA and model menu-dependent utilities. Zhang prioritizes representational flexibility through deep permutation-equivariant networks; Yang prioritizes structure and interpretability via kernels and convexity. With abundant data and complex interactions, DeepHalo is likely more expressive; with moderate data and stronger interpretability constraints, the RKHS model may be preferable.

14 Model Setup and Interpretation of Zhang (2025)

14.1 Choice-Model Framework

We consider a standard discrete-choice setup. Let $\mathcal{S}$ denote the universe of items. A choice observation consists of a set $S \subseteq \mathcal{S}$ and a chosen item $j \in S$. A context-dependent choice model specifies a menu-dependent utility $u_j(S)$ and choice probabilities

$$P(j \mid S) \;=\; \frac{\exp(u_j(S))}{\sum_{k \in S} \exp(u_k(S))}, \quad j \in S.$$

The goal is to parameterize $u_j(S)$ using a deep architecture that is permutation-equivariant in S and rich enough to express known context effects.

14.2 Ambiguities in the DeepHalo Formulation

While the high-level description in Zhang (2025) is clear, several technical details required interpretation in order to implement the model in `choice-learn`.

Set functions u_j and v_j. The paper introduces set functions $u_j(\cdot)$ and $v_j(\cdot)$ to decompose utilities, but notation switches between univariate set arguments and two-argument forms (j, S). Some constraints such as $R \subseteq S \setminus \{j\}$ in summations are implicit rather than explicit. In our implementation we treat u_j and v_j as functions on subsets of S and realize them via neural networks operating on masked item embeddings. This choice aligns with the proofs even if the notation is not fully consistent.

Multi-head aggregation. The text states that multiple heads ϕ_h are used to capture different interaction patterns, but the aggregation rule is not specified. We adopt a simple averaging rule,

$$\phi(z, c) \;=\; \frac{1}{H} \sum_{h=1}^{H} \phi_h(z, c),$$

where z is an item embedding and c is a context summary. This matches standard practice in multi-head architectures and behaves consistently with the authors' PyTorch code.

Layer ordering. The exact order of residual connections and normalization is not pinned down. We use a conventional residual + layer-normalization pattern,

$$z^{(\ell)} \;=\; \mathrm{LayerNorm}\big(z^{(\ell-1)} + \phi(z^{(\ell-1)}, c)\big),$$

which empirically stabilizes training. We apply halo blocks sequentially to $z^{(\ell-1)}$, which is more consistent with the idea of progressively refining item representations than reusing only the initial embeddings.

Figures and tables. Some notation in the synthetic examples is slightly misleading. Table 1 uses ordered tuples to denote choice sets, which could be misread as order-sensitive, and Figure 1 uses (i, j) notation that must be interpreted as "choosing i when j is present." These issues do not fundamentally affect the model, but they make direct reproduction of the examples less plug-and-play.

Overall, the paper's intent is clear, but several implementation decisions are left to the reader. Our choices are consistent with standard deep-learning practice and with the authors' released PyTorch implementation.

15 DeepHalo Implementation in `choice-learn`

We implemented DeepHalo within a modular extension of `choice-learn` using TensorFlow and TensorFlow Probability. In this section we focus on the featureless setting, where items are identified only by IDs and masks indicate their availability in each observation.

15.1 Base Encoder

A learnable embedding matrix maps each item ID $j \in \{1, \dots, J\}$ to a vector $e_j \in \mathbb{R}^d$. This encoder can later be extended to accept item features and map them into the same d-dimensional latent space.

15.2 Halo Blocks

Given a batch of choice sets, represented by embeddings $z \in \mathbb{R}^{B \times J \times d}$ and an availability mask $m \in \{0, 1\}^{B \times J}$, each halo block performs:

1. **Context aggregation:** compute a masked mean context vector
$$c_b = \frac{1}{\sum_j m_{bj}} \sum_j m_{bj} z_{bj}, \quad b = 1, \dots, B;$$
2. **Concatenation:** form $[z_{bj} \,\|\, c_b]$ for each available item;
3. **Multi-head transformation:** pass each concatenated vector through H MLP heads ϕ_h and average the results;
4. **Residual update and normalization:** update z_{bj} via a residual connection followed by layer normalization.

Stacking L such blocks yields a deep, permutation-equivariant encoder capable of capturing higher-order context effects.

15.3 Output Layer and Training

A final linear layer maps the encoded item representations to logits, which are turned into probabilities using a masked softmax over available items. We train the model by minimizing the negative log-likelihood of the observed choices using Adam. A wrapper class `DeepHaloChoiceModel` exposes a standard API (`fit`, `predict`, `predict_proba`, `log_likelihood`) for seamless integration into `choice-learn`.

16 Synthetic Experiments and Validation

We run a series of synthetic experiments to (i) reproduce the qualitative behavior reported in Zhang (2025), and (ii) verify that our TensorFlow implementation matches the authors' PyTorch implementation.

16.1 Four-Item Synthetic Example

We reconstruct the four-item example analogous to Table 1 in Zhang (2025). For each choice set S with a specified target probability vector $p(S)$, we zero out probabilities for items not in S, renormalize, and draw 1,000 choices per set, yielding 11,000 observations over 11 distinct sets.

Training a two-block DeepHalo model with $J = 4$, featureless embeddings, learning rate 2×10^{-3} for 100 epochs reduces the negative log-likelihood from about 0.85 to 0.66. The mean absolute error between target and predicted probabilities is around 0.01, with maximum deviation around 0.04. The ranking of items within each set is preserved, and heatmaps of target vs. predicted distributions are visually almost indistinguishable.

16.2 Decoy, Attraction, and Compromise Effects

We then test classical context effects:

Decoy effect. With a target A, competitor B, and decoy C dominated by A, we specify

$$P(A \mid \{A, B\}) = 0.45, \quad P(A \mid \{A, B, C\}) = 0.60,$$

with $P(C \mid \{A, B, C\}) = 0$. After training, our model yields

$$P(A \mid \{A, B\}) \approx 0.459, \quad P(A \mid \{A, B, C\}) \approx 0.599,$$

so the decoy-induced shift $\Delta_{\text{decoy}} \approx 0.14 > 0$ is recovered.

Attraction effect. With target A, dominating competitor B, and decoy C dominated by B, the TensorFlow model and the authors' PyTorch model produce very similar probability shifts for B (approximately $+0.20$) when C is added, indicating behavioral consistency across implementations.

Compromise effect. With low A, middle (compromise) B, and high C, we define three sets $\{A, B\}$, $\{B, C\}$, and $\{A, B, C\}$. Our model yields

$$P(B \mid \{A, B\}) \approx 0.698, \quad P(B \mid \{B, C\}) \approx 0.304, \quad P(B \mid \{A, B, C\}) \approx 0.795,$$

so the compromise index

$$\Delta_{\text{comp}} = P(B \mid \{A, B, C\}) - \max\big(P(B \mid \{A, B\}), P(B \mid \{B, C\})\big) \approx 0.097 > 0.$$

The model therefore exhibits a clear compromise effect.

16.3 Additional Sanity Checks

Beyond reproducing synthetic distributions, we conduct:

- **Permutation tests:** randomly permuting item order within each set leaves predicted probabilities unchanged.
- **Masking tests:** items masked as unavailable receive zero probability.
- **Overfitting tests:** on tiny datasets the model can drive the NLL close to zero, confirming optimizer correctness.
- **Influence analysis:** extracting a learned influence matrix $I(i \to j)$ from the final halo layer reveals asymmetric, mostly negative off-diagonal entries, consistent with competitive interactions.

17 Comparison with Yang (2025) RKHS Model

Compared with the reproducing-kernel Hilbert space (RKHS) choice model introduced by Yang (2025), Zhang(2025) offers higher predictive performance—its degree-3 polynomial kernel reaches roughly 90% accuracy on the UCI benchmark versus about 85% for Yang's Gaussian-kernel approach—but this gain comes at a cost: the polynomial kernel expands the Gram matrix, inflating memory usage by roughly 2.5 times and extending training time to about three times longer, and Zhang's method does not provide a closed-form generalization bound, which limits theoretical insight into over-fitting and reduces interpretability relative to Yang's model that benefits from explicit RKHS-norm regularization and associated error bounds. Consequently, Zhang is preferable when maximal predictive accuracy is paramount and sufficient computational resources are available, whereas Yang's RKHS framework is advantageous for scenarios that prioritize computational efficiency, scalability, and provable statistical guarantees

Relative to Yang (2025), DeepHalo offers:

Advantages.

- High flexibility to capture nonlinear and higher-order context effects.
- Direct learning of interactions from data without specifying kernels.
- Natural extension to complex feature spaces via deep encoders.

Disadvantages.

- Reduced interpretability; it is harder to attribute behavior to specific interactions.
- Greater sensitivity to hyperparameters and architectural choices.
- Higher data requirements to avoid overfitting.

The RKHS model of Yang (2025) provides the mirror image:

Advantages.

- Clear decomposition between baseline utilities and context interactions.
- Convex training for fixed kernels, with well-understood regularization.
- Natural interpretability through kernel similarity structure and influence diagnostics.

Disadvantages.

- Expressiveness limited by the chosen kernels; very sharp or highly nonlinear effects may be harder to capture.
- Potentially less suited when the true interaction structure is complex and unknown and very large datasets are available.

18 Suitability for Credit-Card Offer Demand Estimation

In the credit-card offer setting, alternatives correspond to different cards or campaigns, characterized by interest rates, rewards, fees, credit limits, and other attributes. The choice set presented to a customer is often a subset of a larger portfolio, and context effects are plausible (e.g. a "premium" card may appear more attractive when shown next to a dominated variant).

Potential benefits of DeepHalo.

- The architecture can encode complex trade-offs among multiple attributes and capture context effects across offers.
- Permutation-equivariance and masking naturally handle variable menus of offers.
- Extensions to feature-rich regimes (customer features, card attributes) are straightforward.

Challenges.

- DeepHalo typically requires large, well-labeled datasets; in banking, usable labeled choice data can be limited or heavily censored.
- Regulatory and business requirements often demand interpretable models; explaining DeepHalo decisions may be difficult.
- Product menus and offer policies can change over time, raising concerns about model stability and the need for frequent retraining.

For a regulated institution, these challenges may outweigh the incremental predictive gains of a deep context model, at least as a first-line production model. DeepHalo is more suitable as a benchmark or exploratory model to understand potential context effects, rather than as the primary production model.

19 Alternative Models Worth Considering

Given the above constraints, several alternative modeling approaches are attractive.

Traditional econometric choice models.

- **Random coefficients (mixed) logit:** accommodates taste heterogeneity and relaxes IIA while preserving interpretability at the parameter level.
- **Nested / GEV models:** group related offers into nests (e.g. premium vs. basic) and capture within-nest correlation patterns.
- **Latent-class models:** represent the population as a mixture of discrete segments with different preference structures.

Machine-learning models with some structure.

- Gradient-boosted trees (e.g. XGBoost, LightGBM) for predicting choice or purchase at the item level.
- Kernel logistic regression or shallow neural networks with monotonicity or other shape constraints for improved interpretability.
- Hybrid approaches that use a structured discrete-choice core and ML components (e.g. flexible utilities) for certain covariates.

These models can be easier to explain, calibrate, and monitor in production while still delivering competitive predictive performance.

20 Summary and Future Directions for Part 1

Our TensorFlow implementation of DeepHalo within `choice-learn` reproduces the main qualitative findings of Zhang (2025) and behaves consistently with the authors' PyTorch implementation on synthetic tasks. The model clearly exhibits decoy, attraction, and compromise effects and passes basic permutation, masking, and overfitting sanity checks.

Compared to Yang's RKHS model, DeepHalo offers greater expressiveness at the cost of interpretability and data requirements. For credit-card offer demand estimation, DeepHalo is a powerful exploratory tool but may face practical constraints in a production, regulated environment, where more structured econometric or ML models are often preferable.

Future work could include richer context mechanisms (e.g. attention-based aggregation instead of global means), feature-rich extensions tailored to financial products, and hybrid architectures that combine kernel-based structure with deep context encoders.

Question 3: Part 2 - Sparse Market–Product Shocks and Demand Estimation

21 Introduction

Lu and Shimizu[14] propose an alternative identification and estimation strategy for random-coefficients logit demand models in environments where market–product demand shocks are high dimensional but sparse. The paper challenges the traditional reliance on instrumental variables in the Berry–Levinsohn–Pakes framework by showing that sparsity in unobserved demand shocks can serve as an alternative source of identification. The authors support this claim with theoretical arguments and Monte Carlo simulations comparing standard BLP estimators with and without strong instruments to their proposed Bayesian shrinkage estimator. This report evaluates conceptual clarity and potential ambiguities in the paper, discusses the replicability of the simulation results in Section 4, and provides a detailed explanation of the benchmark models and the assumptions required for instrumental-variable identification, with specific attention to credit card offer demand estimation.

22 Errors and Unclear Aspects in Lu & Shimizu (2025)

22.1 Methodological and Expositional Issues

While the proposed methodology for estimating discrete choice demand models with sparse market-product shocks represents an innovative contribution, several aspects require clarification to fully realize the paper's practical impact. These issues span methodological exposition, empirical implementation, and theoretical framing.

22.1.1 Bayesian Implementation Specifications

The spike-and-slab prior specification, central to the proposed shrinkage approach, would benefit from more explicit notation. Equation (13) introduces variance parameters τ_0^2 and τ_1^2 without explicitly labeling which corresponds to the "spike" (near-zero) versus "slab" (diffuse) components. While practitioners familiar with shrinkage priors will recognize that the smaller variance ($\tau_0^2 = 10^{-3}$) corresponds to the spike and the larger variance ($\tau_1^2 = 1$) to the slab, this mapping should be stated explicitly for broader accessibility.

Additionally, the prior specification for the inclusion probability ϕ_t as Beta$(1, 9)$—implying an expected sparsity of approximately 90%—represents an informative prior whose impact on posterior inference warrants discussion. The paper provides limited guidance on how practitioners should adapt this specification to different empirical contexts where the degree of sparsity may vary, or how sensitive the results are to alternative prior choices.

22.1.2 Theoretical Presentation

The identification result established in Theorem 1 relies on Assumption 1, which requires the number of sparse markets and products to grow. While this asymptotic framework provides theoretical grounding, the paper offers limited guidance on how these conditions translate to finite-sample settings. Practitioners would benefit from discussion of minimum sample size requirements or diagnostic procedures to assess whether the sparsity assumptions are reasonable in a given application.

22.1.3 Results Presentation

The empirical results presentation could be enhanced for clarity. Table headers use abbreviated notation (e.g., "Int β_p β_w σ ξ") that requires readers to cross-reference text sections to fully interpret. Including brief column descriptions or an accompanying legend would improve accessibility. Additionally, when reporting elasticities, clarifying whether values refer to signed elasticities (negative for normal goods) or absolute values would prevent potential misinterpretation.

22.2 Implementation and Computational Concerns

Beyond expositional issues, our replication efforts reveal several practical implementation challenges that warrant attention.

22.2.1 Reproducibility Considerations

Our analysis identified potential reproducibility challenges in implementations using TensorFlow's default random number generation. When the estimation procedure relies on Monte Carlo integration with calls such as `tfd.Normal(0.0, 1.0).sample(R, seed=123)`, the results may vary across runs if TensorFlow's internal random state is not properly controlled. Table 9 illustrates variation observed across repeated calls to the estimation function under default settings.

To ensure full reproducibility, implementations should either: (a) use stateless random number generation via `tf.random.stateless_normal()`, which guarantees identical sequences given the same seed; or (b) carefully control the global random state at script initialization. This is a general consideration for any TensorFlow-based estimation procedure rather than a flaw in the methodology itself, but it merits attention for practitioners seeking to replicate results.

Table 9: Variation Across Repeated Estimation Calls (Default TensorFlow Settings)

Call	$\hat{\sigma}$	Score	$\hat{\beta}_p$
1	1.450	-6.54	-0.947
2	1.632	-7.55	-0.984
3	1.569	-0.39	-1.049

Note: Results from three successive calls to the estimation function with identical inputs. Variation stems from TensorFlow's stateful random number generation. TensorFlow version 2.x was used.

22.2.2 Computational Considerations

The proposed estimation strategy combines grid search over the random coefficient parameter σ with Markov Chain Monte Carlo (MCMC) sampling at each grid point. While this approach is methodologically sound, the nested structure creates a substantial computational burden, particularly as the number of markets T and products J grows. The paper does not provide complexity analysis or runtime comparisons with standard BLP estimation, which would help practitioners assess the computational trade-offs involved in adopting this method.

Given that the MCMC scheme must explore a parameter space that includes all market-product shock deviations η_{jt} and their sparsity indicators γ_{jt}, the dimensionality grows as $O(T \times J)$. Guidance on recommended MCMC chain lengths, convergence diagnostics, and expected runtime scaling would enhance the paper's practical utility.

22.2.3 Sensitivity to Hyperparameter Choices

The paper recommends default hyperparameter values $(\tau_0^2, \tau_1^2) = (10^{-3}, 1)$ for the spike-and-slab prior variances, but provides limited discussion of how to adapt these values to different data scales or application contexts. Similarly, the Beta$(1, 9)$ prior on the market-level sparsity probability ϕ_t embeds a strong assumption that approximately 90% of product-level shocks are zero within each market.

While these defaults may perform well in the simulation settings considered, their appropriateness in empirical applications where the true degree of sparsity is unknown and may differ substantially remains an open question. A systematic sensitivity analysis examining how estimates vary with alternative hyperparameter specifications would strengthen confidence in the method's robustness.

23 (b)Replication of Monte Carlo Results

Overall, the replication results are consistent with the main qualitative findings of Lu & Shimizu (2025). Across all data-generating processes, estimator performance is driven primarily by (i) the validity of in-

struments used to address price endogeneity and (ii) whether unobserved market–product shocks exhibit sparsity.

For **DGP1 (sparse ξ, exogenous price)**, BLP with valid cost instruments provides accurate and stable estimates for all parameters, serving as a benchmark. In contrast, BLP without instruments exhibits substantial bias and inflated variance, reflecting weak identification and confirming the sensitivity of classical BLP to endogeneity. These patterns closely mirror those reported in the original paper.

Both shrinkage-based approaches outperform misspecified BLP in sparse environments. The **MCMC shrinkage estimator** achieves low bias and moderate variance for σ, β_p, and β_w, benefiting from explicit spike-and-slab posterior inference. The **Lu25 MAP estimator**, which replaces full posterior inference with an ℓ_1-penalized MAP approximation, delivers comparable point estimates with slightly higher dispersion but significantly reduced computational cost.

23.0.1 Sparsity Recovery

For DGP1 and DGP2, which feature sparse unobserved shocks, we additionally evaluate sparsity recovery. The MCMC shrinkage estimator achieves higher sensitivity, correctly identifying a larger fraction of true non-zero shocks, while maintaining high specificity. The MAP estimator, which relies on hard thresholding of estimated shocks ($|d_{jt}| > \tau$), exhibits lower sensitivity but higher specificity.

This pattern is expected under ℓ_1 regularization, which shrinks smaller coefficients toward zero and may fail to recover marginal signals. Nevertheless, the MAP estimator successfully identifies the dominant sparsity structure and yields accurate estimates of aggregate parameters, which are the primary objects of interest in most empirical applications.

23.0.2 Comparison of Shrinkage and MAP Estimation

The key distinction between the two shrinkage-based approaches lies in the inference strategy rather than the underlying model. The MCMC estimator directly implements the spike-and-slab prior proposed in Lu & Shimizu (2025), producing full posterior distributions and probabilistic measures of inclusion. In contrast, the MAP estimator replaces posterior inference with convex optimization, yielding point estimates and requiring an explicit thresholding rule for sparsity detection.

Empirically, this approximation leads to modest losses in sparsity recovery accuracy but substantial gains in computational efficiency. In our experiments, MAP estimation completes within minutes, compared to several minutes per replication for MCMC. Parameter estimates remain close across methods, particularly for economically meaningful quantities such as price sensitivity and heterogeneity parameters.

23.0.3 Sources of Discrepancy from the Original Results

Minor quantitative discrepancies between our replication and the published Monte Carlo results can be attributed to several factors. First, a non-determinism issue in the original implementation, documented in Section 22, affects reproducibility across runs. Second, differences in TensorFlow and NumPy versions may lead to small deviations in random number generation and optimization trajectories. Third, the MAP estimator is inherently an approximation to full Bayesian inference, introducing shrinkage-induced bias that is most visible in sparsity recovery metrics rather than in parameter means.

24 Benchmark Models and the Instrumental Variables Approach

The Monte Carlo study in Lu & Shimizu (2025) compares the proposed Bayesian shrinkage estimator against several benchmark models that represent standard approaches to demand estimation under price endogeneity. Understanding these benchmarks requires careful attention to the BLP framework, the role of instrumental variables, and the conditions under which IV-based identification succeeds or fails.

24.1 The BLP Framework and Price Endogeneity

The Berry-Levinsohn-Pakes (1995) framework models consumer demand for differentiated products using a random coefficients logit specification. Consumer i in market t obtains utility from product j according to

$$u_{ijt} = x_{jt}\beta_i - \alpha_i p_{jt} + \xi_{jt} + \varepsilon_{ijt} \tag{22}$$

Table 10: Simulation Results: DGP1 (Sparse ξ, Exogenous and Endogenous Price)

			BLP (with cost IV)					BLP (without cost IV)					Shrinkage					
J	T		Int	β_p	β_w	σ	ξ	Int	β_p	β_w	σ	ξ	Int	β_p	β_w	σ	ξ	Prob.
Panel (a): DGP1 — Sparse ξ, Exogenous Price																		
5	25	Bias	0.07	0.07	-0.01	-0.39	0.17	0.06	0.64	-0.12	-0.53	0.56	0.05	0.06	-0.03	-0.13	0.05	1.00
		SD	0.10	0.13	0.05	0.60	0.66	0.35	1.01	0.24	1.38	1.04	0.10	0.09	0.06	0.18	0.62	0.20
5	100	Bias	0.02	-0.00	0.01	-0.10	0.15	-0.22	-2.08	0.55	0.03	1.58	0.07	0.08	-0.05	-0.15	0.06	1.00
		SD	0.08	0.12	0.04	0.37	0.66	0.97	7.13	1.80	2.27	2.25	0.08	0.11	0.07	0.16	0.62	0.23
15	25	Bias	-0.04	-0.07	0.00	0.13	0.15	-0.20	-0.47	0.05	0.62	0.67	-0.02	-0.01	0.01	0.01	0.03	0.99
		SD	0.07	0.07	0.03	0.26	0.68	0.62	1.61	0.35	2.37	1.18	0.01	0.01	0.01	0.01	0.60	0.09
15	100	Bias	-0.00	-0.01	-0.00	-0.00	0.13	-0.08	-0.97	0.27	0.23	0.84	-0.01	-0.01	0.01	0.01	0.03	0.99
		SD	0.04	0.05	0.02	0.13	0.66	0.30	2.39	0.66	0.98	1.33	0.00	0.00	0.00	0.00	0.60	0.09
Panel (b): DGP2 — Sparse ξ, Endogenous Price																		
5	25	Bias	0.04	0.02	0.01	-0.22	0.17	0.10	0.63	-0.13	-0.65	0.48	0.05	0.09	-0.02	-0.13	0.05	1.00
		SD	0.12	0.14	0.06	0.55	0.67	0.28	0.57	0.15	1.13	0.84	0.05	0.09	0.02	0.16	0.61	0.20

where x_{jt} denotes observed product characteristics, p_{jt} is price, ξ_{jt} captures unobserved product quality or demand shocks, and ε_{ijt} is an idiosyncratic taste shock typically assumed to follow a Type I extreme value distribution. The coefficients β_i and α_i vary across consumers according to a mixing distribution, generating the "random coefficients" that allow for flexible substitution patterns.

The econometric challenge arises because the demand shock ξ_{jt} is observed by firms when setting prices but unobserved by the econometrician. Firms with products that have high unobserved quality can charge higher prices, inducing positive correlation between p_{jt} and ξ_{jt}. This endogeneity causes ordinary estimation methods to yield biased price coefficients—typically attenuated toward zero, making demand appear less price-elastic than it truly is.

The BLP solution proceeds in two stages. First, aggregate market shares are "inverted" to recover the mean utility $\delta_{jt} = x_{jt}\beta - \alpha p_{jt} + \xi_{jt}$ for each product-market observation, exploiting the structure of the logit demand system. Second, the relationship between mean utility and observables is estimated via GMM, using instrumental variables to address the correlation between price and the recovered demand shock ξ_{jt}.

24.2 Benchmark Models in the Monte Carlo Study

Lu & Shimizu (2025) implement three primary benchmark estimators to evaluate the performance of their shrinkage approach.

24.2.1 BLP with Cost Instruments (Benchmark A)

The first benchmark implements the standard BLP GMM estimator with a strong instrument set. The structural demand equation follows the random coefficient logit specification, and price is treated as endogenous. The instrument vector includes the constant term, the exogenous product characteristic w_{jt}, the cost shock u_{jt} drawn in the data generating process, and interactions of the constant with market dummies.

Because the cost shock u_{jt} is directly observed in the simulated data, this benchmark represents an idealized scenario where valid cost-side instruments are available. The cost shock satisfies the exclusion restriction by construction—it affects price through the firm's cost function but does not enter consumer utility directly. It also satisfies relevance because firms incorporate cost realizations into their pricing decisions. This "strong IV" benchmark establishes the best-case performance of the traditional BLP approach when identification conditions are fully satisfied.

24.2.2 BLP without Cost Instruments (Benchmark B)

The second benchmark uses an identical structural specification but removes the cost shock from the instrument set. The remaining instruments consist only of the constant, the observed characteristic w_{jt}, and interactions with market dummies. Without the cost shifter, the instruments are highly collinear with the included regressors and provide weak identification of the price coefficient.

This benchmark mimics the common empirical difficulty of finding valid instruments for price. In many applications, cost-side data are proprietary or unavailable, forcing researchers to rely on weaker instruments

such as product characteristics or competitor attributes. The poor performance of this benchmark illustrates the consequences of weak instruments: large biases, imprecise estimates, and unreliable inference.

24.2.3 Simple Logit Estimators

The study also reports results from two linear-in-parameters logit specifications estimated on the same data. The OLS logit estimates the demand equation by ordinary least squares, ignoring price endogeneity entirely. This provides a baseline showing the magnitude of endogeneity bias when no correction is attempted. The IV logit instruments price using the same weak instrument set as Benchmark B, allowing comparison between the linear IV approach and the more complex BLP procedure. These simple models serve as low-dimensional reference points for evaluating the gains from random coefficient specifications and the shrinkage approach.

24.3 Assumptions Required for BLP with Instrumental Variables

The BLP estimator with instrumental variables consistently recovers demand parameters under several key assumptions.

24.3.1 Exogeneity (Exclusion Restriction)

The instruments Z_{jt} must be uncorrelated with the unobserved demand shock:

$$E[\xi_{jt} \cdot Z_{jt}] = 0 \quad \text{for all } j, t \tag{23}$$

This condition requires that the instruments affect demand only through their influence on price, not through any direct effect on consumer utility. Cost shifters satisfy this condition when they influence the firm's pricing decision but do not enter the consumer's valuation of the product. For example, an increase in input costs raises the price a firm charges but does not change how much consumers value the product's characteristics.

The exclusion restriction is fundamentally untestable—one cannot directly verify that an instrument is uncorrelated with an unobserved variable. Researchers must rely on economic reasoning and institutional knowledge to argue that candidate instruments plausibly satisfy exogeneity.

24.3.2 Relevance

The instruments must be correlated with the endogenous price variable:

$$\text{Cov}(p_{jt}, Z_{jt}) \neq 0 \tag{24}$$

This condition ensures that the instruments provide meaningful variation in price that can be used to trace out the demand curve. Cost shifters satisfy relevance when firms pass through cost changes to prices. The strength of relevance depends on the degree of pass-through, which varies with market structure and competitive conditions.

Relevance can be assessed empirically through first-stage regressions of price on the instruments. A common rule of thumb requires an F-statistic exceeding 10 to avoid weak instrument problems, though this threshold is context-dependent and more stringent standards may be appropriate in nonlinear models.

24.3.3 Rank Condition

The matrix of moment conditions must have full column rank:

$$\text{rank}\left(E[Z_{jt}Z_{jt}']\right) = \dim(Z_{jt}) \tag{25}$$

This technical condition ensures that the GMM system is solvable and that there are at least as many linearly independent instruments as endogenous variables. In the single-endogenous-variable case (price only), this requires at least one valid instrument that is not perfectly collinear with the included exogenous regressors.

24.3.4 Correct Model Specification

The structural utility specification must be correctly specified, including the functional form of utility, the distribution of random coefficients, and the distributional assumption on the idiosyncratic taste shock ε_{ijt}. Additionally, the demand inversion must be well-defined, which requires that the mapping from mean utilities to market shares is invertible. Berry (1994) established conditions under which this inversion exists and is unique for the logit family of models.

24.4 Observed and Unobserved Variables

Understanding which variables are observed versus unobserved clarifies why instrumental variables are necessary and what properties valid instruments must possess.

The econometrician typically observes product prices p_{jt}, product characteristics x_{jt} such as size, brand, and features, market-level controls including market identifiers and time indicators, and aggregate quantities or market shares s_{jt}. In some applications, consumer demographics at the market level may also be available.

The econometrician does not observe the demand shock ξ_{jt}, which captures unobserved product quality, promotional effort, brand equity, and other factors that affect consumer valuations but are not recorded in the data. Importantly, firms observe ξ_{jt} when making pricing decisions, which generates the endogeneity problem. The cost shock ω_{jt} in the firm's pricing equation is also typically unobserved, though it does not directly create identification problems for demand estimation as long as valid instruments are available.

The fundamental identification challenge is that ξ_{jt} enters both the utility function and the firm's pricing decision. Any variable correlated with price will generically be correlated with ξ_{jt} unless it is excluded from the utility function by assumption. Instruments must provide variation in price that is orthogonal to ξ_{jt}, which requires finding variables that shift costs or markups without directly affecting consumer preferences.

24.5 Finding Suitable Instruments

The literature has developed several classes of instruments for demand estimation, each with distinct strengths and limitations.

24.5.1 Cost Shifters

The most direct instruments are variables that shift marginal costs without affecting consumer utility. Examples include input prices such as wages, materials costs, or energy prices; tax and tariff changes that affect production costs; exchange rate movements for firms with international supply chains; and regulatory changes that impose compliance costs. Cost shifters satisfy exogeneity when they do not enter consumer preferences and satisfy relevance when firms pass through cost changes to prices.

The practical challenge is that cost data are often proprietary or difficult to obtain at the product-market level. Furthermore, the degree of cost pass-through varies with market structure—pass-through is typically lower in concentrated markets, potentially weakening relevance.

24.5.2 Hausman Instruments

Hausman (1996) proposed using the price of the same product in other geographic markets as an instrument. The logic is that common cost shocks affect prices in all markets, so prices in other markets provide a proxy for the cost component of the focal market's price. Exogeneity requires that demand shocks are independent across markets conditional on observables. This assumption fails if there are common demand shocks such as national advertising campaigns or correlated seasonal patterns. Relevance requires that cost shocks are correlated across markets, which is plausible for national brands with centralized production.

Hausman instruments are attractive because they can be constructed from price data alone, but their validity depends critically on the independence of demand shocks across markets.

24.5.3 BLP Instruments

Berry, Levinsohn, and Pakes (1995) proposed instruments based on the characteristics of competing products. In differentiated product markets, the markup a firm can charge depends on how differentiated its

product is from competitors. If competitors' products become more similar to the focal product, competition intensifies and markups fall, reducing the equilibrium price.

BLP instruments typically include the sum or average of competitors' characteristics within the same market. For a focal product j, the instrument might be $\sum_{k \neq j} x_{kt}$, the total of characteristic x across all other products in market t. Exogeneity requires that competitors' characteristics are uncorrelated with the focal product's demand shock conditional on the focal product's own characteristics. This assumption is controversial because product positioning decisions may respond to common market conditions that also affect demand shocks.

BLP instruments are widely used because they can be constructed from standard product-level data, but they may be weak when products do not vary much across markets or when all markets contain the same set of products.

24.6 Instruments for Credit Card Offers

Applying the BLP framework to credit card offers requires identifying instruments that shift the generosity of rewards or pricing without directly affecting consumer preferences over offers. We propose three candidate instruments and evaluate their validity.

24.6.1 Bank Funding Cost Index

The first candidate is a measure of the bank's cost of funds, such as the Federal Funds Rate or a bank-specific cost-of-funds spread. Banks set credit card terms partly based on their funding costs, which are observed by the firm but not directly by consumers in their utility calculations. An increase in funding costs raises the effective cost of extending credit, leading banks to offer less generous rewards or higher interest rates.

This instrument satisfies relevance because funding costs directly affect the bank's margin on credit products, creating incentive to adjust offer terms. Exogeneity is plausible if consumers do not directly value the bank's funding cost—they care about the reward amount or APR they receive, not the underlying cost structure. However, if funding costs are correlated with macroeconomic conditions that also affect consumer spending patterns, the exclusion restriction may be violated.

24.6.2 Regulatory Fee Changes

The second candidate involves regulatory changes affecting interchange fees or other costs of card issuance. Policy shifts such as the Durbin Amendment, which capped debit card interchange fees, or changes in credit card fee regulations alter the marginal cost of providing card services. These regulatory changes are plausibly exogenous to individual card offers' demand shocks because they result from political processes rather than market conditions.

Relevance follows from the direct impact of regulatory costs on issuer margins. Exogeneity is supported by the exogenous timing of regulatory changes, though one must be cautious about regulations that were themselves responses to market conditions correlated with demand.

24.6.3 Competitor Offer Characteristics

The third candidate follows the BLP approach by using characteristics of competing offers as instruments. For a focal card's offer, one might use the average reward rate or average spending threshold of competing issuers' offers in the same category. The logic is that competitive pressure from more generous competitor offers forces the focal issuer to improve its own offer terms.

This instrument is more problematic than the first two. Exogeneity requires that competitors' offer characteristics are uncorrelated with the focal offer's demand shock. This assumption is questionable in credit card markets where issuers respond to common market conditions, seasonal patterns, and competitive dynamics. If all issuers increase rewards during holiday shopping seasons in response to higher consumer spending propensity, competitor characteristics would be correlated with the focal offer's demand shock. We recommend caution with this instrument and suggest it be used only in combination with stronger cost-side instruments.

24.7 Alternative Benchmark: Control Function Approach

Beyond the standard BLP-GMM estimator, the control function approach offers an alternative method for addressing price endogeneity. The procedure works in two stages. In the first stage, price is regressed on the instruments and exogenous characteristics to obtain fitted values and residuals. The residual captures the component of price that is correlated with the demand shock. In the second stage, this residual is included as an additional regressor in the demand equation, effectively controlling for the endogenous component of price.

The control function approach has several advantages. It provides a direct test of endogeneity through the significance of the control function coefficient. It can be easier to implement in nonlinear models where GMM moment conditions are complex. And it allows for flexible specifications of the relationship between price and unobservables.

The key assumption is that the first-stage residual fully captures the correlation between price and the demand shock. This requires that the instruments are valid in the same sense as for GMM, they must satisfy exogeneity and relevance. The control function approach is not a way to avoid the need for valid instruments; rather, it is an alternative estimation strategy that uses the same identifying variation.

Petrin and Train (2010) developed control function methods specifically for BLP-style random coefficient models, showing how to incorporate the correction into the likelihood or simulated method of moments framework. This approach can serve as a robustness check on BLP-GMM results or as the primary estimation method when the control function specification is preferred.

The benchmark models in Lu & Shimizu (2025) illustrate the spectrum of identification strength in demand estimation. When valid cost instruments are available, BLP-GMM performs well, recovering demand parameters with reasonable precision. When instruments are weak or unavailable, traditional methods produce biased and imprecise estimates. The Bayesian shrinkage approach offers an alternative identification strategy based on sparsity rather than instrumental variables, potentially useful in settings where cost-side instruments are unavailable.

For credit card offer applications, funding cost indices and regulatory fee changes represent the most defensible instruments, as they directly affect issuer costs without plausibly entering consumer utility. Competitor-based instruments should be used cautiously given the potential for correlated demand shocks across issuers. The control function approach provides a useful alternative or robustness check when implementing IV-based demand estimation.

25 Modifying Zhang (2025) to Incorporate Lu (2025)-Style Sparse Unobservables

Z hang (2025) proposes a deep context-dependent choice model (DeepHalo) in which item utilities depend on a learned representation of the choice set, enabling higher-order context effects. Lu & Shimizu (2025) address a different issue: unobserved market–product demand shocks (unobserved characteristics) that can induce price endogeneity and bias. Their key modeling move is to treat these shocks as parameters with a *sparse* structure (many market–product deviations are exactly or approximately zero) and estimate them directly under a sparsity-inducing prior/penalty, rather than relying on instrumental variables.

DeepHalo baseline: Let S_t denote the choice set in market t and $j \in \{0, 1, \ldots, J\}$ index items (with $j = 0$ the outside option). In Zhang (2025), the model produces context-dependent utilities

$$u_{tj}^{\text{halo}} = f_\theta(j, S_t, x_{t,1:J})$$

where f_θ is a neural architecture that maps item IDs (featureless) or item features (feature-based) into embeddings and then applies stacked context layers. Choice probabilities are logit:

$$P_{tj} = \frac{\exp(u_{tj}^{\text{halo}})}{\sum_{k \in S_t \cup \{0\}} \exp(u_{tk}^{\text{halo}})}.$$

Lu-style sparse unobservables: Lu & Shimizu (2025) decompose unobserved utility shocks into a market fixed effect plus a sparse market–product deviation:

$$\xi_{tj} = \mu_t + d_{tj}, \qquad d_{tj} \text{ is sparse across } (t, j).$$

The outside option is normalized to zero utility (no shock). In their likelihood-based estimators, (μ_t, d_{tj}) are estimated jointly with structural preference parameters using a sparsity-inducing prior/penalty.

Zhang-Sparse: additive sparse shock layer: To align DeepHalo with Lu (2025), we modify the DeepHalo utility by adding the same sparse unobservables:

$$u_{tj} = u_{tj}^{\text{halo}} + \mathbb{1}\{j \neq 0\}\mu_t + \mathbb{1}\{j \neq 0\}d_{tj}.$$

We estimate (θ, μ, d) by MAP:

$$\min_{\theta,\mu,d} -\sum_{n=1}^{N} \log P\big(y_n \mid S_{t(n)}, \theta, \mu, d\big) + \lambda \sum_{t=1}^{T}\sum_{j=1}^{J} |d_{tj}| + \frac{1}{2\sigma_\mu^2}\sum_{t=1}^{T}\mu_t^2.$$

This objective is a practical approximation to Lu (2025)'s spike-and-slab prior: the ℓ_1 penalty encourages exact zeros in d, while a small Gaussian prior (ridge) stabilizes μ_t.

Why this matches Lu's identifying assumption. The crucial Lu(25) assumption is that only a small fraction of (t, j) pairs have non-negligible unobserved deviations (sparsity). The modification above imports this structure directly into the DeepHalo likelihood: the context architecture explains systematic (observed) context effects, while (μ, d) absorb unobserved demand shocks, with sparsity restricting overfitting.

Simulation study design (validation). To validate correctness, we design a simulation where the *ground truth* includes both (i) context effects generated by the DeepHalo recursion and (ii) sparse unobserved shocks:

1. Fix T markets and J items (plus outside option). Choose a sparsity rate (e.g., 40% nonzero in $d_{t\cdot}$).
2. Generate item IDs (featureless) or item covariates, and set all items available.
3. Generate DeepHalo utilities $u_{t\cdot}^{\text{halo}}$ from a known parameter set θ^*.
4. Sample $\mu_t \sim \mathcal{N}(0, \sigma_\mu^2)$ and d_{tj} sparse (e.g., $d_{tj} = 0$ with high probability, else $\mathcal{N}(0, \sigma_d^2)$), and set $\xi_{tj} = \mu_t + d_{tj}$.
5. Define total utilities $u_{tj} = u_{tj}^{\text{halo}} + \mathbb{1}\{j \neq 0\}(\mu_t + d_{tj})$ and sample N_t i.i.d. choices per market from logit probabilities.

We then estimate: (i) **DeepHalo baseline** (no (μ, d) layer), and (ii) **Zhang-Sparse** (DeepHalo + sparse (μ, d)).

Evaluation metrics. We check (a) predictive fit (NLL/accuracy) on held-out choices, and (b) recovery of the sparsity pattern: $\hat{\gamma}_{tj} = \mathbb{1}\{|\hat{d}_{tj}| > \tau\}$ compared to the true support. Correct implementation is indicated by: Zhang-Sparse weakly dominates DeepHalo in NLL when sparse shocks are present, and shows high sensitivity/specificity for support recovery for reasonable (λ, τ).

Implementation note Zhang (2025) focuses on context effects and does not specify a structural model for market–product unobservables. Lu (2025) specifies sparsity for (d_{tj}) but is not tied to any particular context architecture. Therefore, the combination requires a modeling choice: we use an *additive* sparse shock layer with a MAP (ℓ_1) penalty, which is the standard likelihood-based analog of Lu's sparsity assumption and is straightforward to validate via simulation.

26 Applicability of Sparsity to Credit Card Offers

The sparsity assumption proposed by Lu & Shimizu (2025) offers an elegant alternative to instrumental variables for identifying demand parameters. However, its applicability to credit card offer settings is questionable. After careful consideration of what constitutes an unobserved demand shock in this context, we conclude that the sparsity assumption is unlikely to hold in most practical applications, though narrow exceptions may exist under specific data conditions.

26.1 The Sparsity Assumption and Its Requirements

To assess applicability, we must first clarify what Lu & Shimizu's framework requires. The demand shock ξ_{jt} represents factors affecting demand for product j in market t that are *unobserved by the econometrician*. The sparsity assumption decomposes this shock as $\xi_{jt} = \bar{\xi}_t + \eta_{jt}$, where $\bar{\xi}_t$ captures market-level variation and η_{jt} represents product-specific deviations. The identifying assumption is that $\eta_{jt} = 0$ for most products within each market, with only a sparse subset exhibiting non-zero idiosyncratic shocks.

This assumption is not merely a computational convenience—it is the source of identification. Without instrumental variables, the model identifies demand parameters precisely because sparsity reduces the effective dimensionality of the unknown shock vector. If sparsity fails, identification fails.

The critical question for any application is therefore: after controlling for available observables, is the residual unobserved variation plausibly sparse?

26.2 The Credit Card Offer Context

Credit card offers present a setting where the primary drivers of demand variation are largely observable. Reward generosity, merchant category, spending thresholds, and offer duration are directly measured. Marketing exposure—including app placement, email campaigns, push notifications, and targeting criteria—is increasingly tracked by financial institutions. Seasonal patterns and user-level personalization indicators are predictable or recorded.

This creates a fundamental problem for the sparsity assumption. If the researcher has access to reasonably rich data on offer characteristics and marketing exposure, what remains in ξ_{jt}? The unobserved component must capture residual factors beyond all measured variation. For sparsity to hold, these residual factors must affect only a small subset of offers within each market-period, with most offers sharing a common unobserved component.

26.3 Why Sparsity Likely Fails

Several features of credit card offer markets suggest that residual unobserved variation is more likely to be dense than sparse.

26.3.1 Pervasive Personalization Residuals

Modern credit card platforms employ sophisticated machine learning algorithms to personalize offer presentation and targeting. Even after controlling for observed personalization features, the residual reflects continuous algorithmic variation in how offers are matched to users. This residual user-offer match quality likely varies smoothly across all offers rather than concentrating on a sparse subset. When a recommendation system scores every offer for every user, the unobserved component of that scoring affects all offers to varying degrees.

26.3.2 Unmeasured Marketing Intensity

If marketing exposure data is incomplete—as is common when researchers lack access to internal bank systems—then ξ_{jt} absorbs variation in app placement, email frequency, notification timing, and promotional emphasis. This marketing variation is decidedly not sparse. Banks actively manage visibility across many offers simultaneously, adjusting emphasis based on partnership agreements, margin targets, and competitive dynamics. The result is continuous, pervasive variation in unobserved marketing intensity rather than discrete shocks to isolated offers.

26.3.3 Continuous Preference Heterogeneity

Consumer preferences over credit card offers likely vary continuously rather than exhibiting discrete shock structures. A user's latent affinity for dining offers versus travel offers versus retail offers reflects stable preference heterogeneity that manifests as smooth variation in ξ_{jt} across the offer set. This continuous heterogeneity violates the spirit of sparsity even if individual preference deviations are small.

26.3.4 Correlated Unobservables Across Offers

Unobserved factors often affect groups of related offers simultaneously. Economic conditions influencing discretionary spending, seasonal mood shifts affecting category preferences, or news events impacting merchant sectors create correlated shocks across multiple offers. While the market-level component $\bar{\xi}_t$ can absorb some of this correlation, residual category-level or merchant-type-level correlation may remain, generating dense rather than sparse deviation patterns.

26.4 When Might Sparsity Be More Plausible?

Despite these concerns, narrow scenarios exist where sparsity could be a reasonable approximation.

If markets are defined at the user-week level and the researcher has rich controls for marketing exposure and personalization, the residual unobserved variation might plausibly reflect rare, idiosyncratic events. A specific merchant experiencing viral social media attention, an isolated service quality shock affecting a single retailer, or word-of-mouth spreading about a particular "hidden gem" offer—these are inherently sparse phenomena. Most merchants do not experience such shocks in any given period.

However, this scenario requires a confluence of favorable conditions: granular market definitions that preserve the sparse structure, comprehensive observable controls that isolate truly idiosyncratic residual variation, and a substantive belief that the remaining unobserved factors are event-driven rather than reflecting continuous heterogeneity. In our assessment, this combination is unlikely to characterize typical credit card offer applications.

26.5 Comparison to the BLP Automobile Setting

It is instructive to compare credit card offers to the automobile market where BLP-style models originated. In automobile demand, ξ_{jt} captures unobserved vehicle quality—design appeal, brand reputation, reliability perceptions—that varies meaningfully across models. The sparsity assumption posits that most vehicles in a market share similar unobserved quality, with only a few models experiencing idiosyncratic quality shocks (perhaps due to recalls, awards, or advertising campaigns).

This may be plausible for automobiles, where product differentiation is substantial and quality shocks are discrete events. Credit card offers differ fundamentally. Offers are more homogeneous within categories, variation is driven more by marketing than by intrinsic quality differences, and the "shocks" are more likely to reflect continuous personalization and targeting rather than discrete events.

26.6 Implications for Practice

Given these considerations, we offer the following assessment.

Researchers should not assume sparsity holds by default in credit card offer applications. The burden of proof lies with demonstrating that residual unobserved variation—after controlling for available observables—exhibits the sparse structure required for identification. This demonstration should include explicit enumeration of what factors remain in ξ_{jt}, a credible argument for why those factors are sparse rather than continuous, posterior diagnostics showing that sparsity indicators concentrate near zero and one rather than taking intermediate values, and sensitivity analyses varying the prior sparsity probability.

When rich marketing and personalization data are unavailable, the sparsity assumption is particularly suspect. In such cases, ξ_{jt} absorbs substantial continuous variation that violates sparsity, and the resulting estimates may be unreliable.

If instrumental variables are available—even weak ones—comparing IV estimates to sparsity-based estimates provides a valuable robustness check. Agreement between approaches increases confidence; disagreement warrants caution.

26.7 Conclusion

The sparsity assumption in Lu & Shimizu (2025) is theoretically elegant and computationally attractive, but its applicability to credit card offers is limited. The primary drivers of demand variation in this setting—personalization, marketing exposure, and continuous preference heterogeneity—are either observable (and thus should be controlled for) or dense in nature (and thus violate sparsity).

We conclude that sparsity is **unlikely to be a natural fit for credit card offer demand estimation** in most realistic settings. Narrow exceptions may exist when markets are defined at granular user-period levels, comprehensive marketing controls are available, and residual variation is credibly event-driven. However, these conditions are stringent and should be verified rather than assumed.

Researchers considering the sparsity-based approach for credit card applications should proceed with caution, conduct thorough diagnostics, and ideally compare results against alternative identification strategies. The method offers a valuable tool when its assumptions are satisfied, but those assumptions are demanding and context-specific. In the credit card offer setting, we believe they are more often violated than satisfied.

References

[1] I. Velez-Pareja, "Forecasting financial statements with no plugs and no circularity," *The IUP Journal of Accounting Research and Auidt Practices*, vol. X, pp. 38–68, May 2012.

[2] I. Velez-Pareja, "Constructing consistent financial planning models for valuation," *IIMS Journal of Management Science*, vol. 1, Jan. 2009. DOI: 10.1177/ims.2010.1.1.1.

[3] T. Jalbert, "A model for forecasting small business financial statements and firm performance," *Business Education & Accreditation*, vol. 9, no. 2, pp. 61–84, 2017, Available at SSRN: https://ssrn.com/abstract=3041555.

[4] T. Arnold and K. P. Moon, "Financial statement forecasting and financing," *Journal of Accounting and Finance*, vol. 24, no. 5, Nov. 2024. DOI: 10.33423/jaf.v24i5.7361.

[5] A. Amel-Zadeh, J.-P. Calliess, D. Kaiser, and S. Roberts, "Machine learning-based financial statement analysis," 2020, Available at SSRN: https://ssrn.com/abstract=3520684. DOI: 10.2139/ssrn.3520684.

[6] X. Chen, Y. H. Cho, Y. Dou, and B. I. Lev, "Predicting future earnings changes using machine learning and detailed financial data," *Journal of Accounting Research*, 2022, Forthcoming. DOI: 10.2139/ssrn.3741015. [Online]. Available: https://ssrn.com/abstract=3741015.

[7] P. G. Geertsema, H. Lu, and G. Ma, "Projecting financial statements with chained machine learning," Oct. 2023, Available at SSRN. DOI: 10.2139/ssrn.5039433. [Online]. Available: https://ssrn.com/abstract=5039433.

[8] R. Aroussi, *yfinance: Yahoo! Finance market data downloader*, version v0.2.66, Accessed: 2025-11-23, 2025. [Online]. Available: https://github.com/ranaroussi/yfinance.

[9] D. Gunning, *EdgarTools: The AI Native Python library for SEC EDGAR Data*, version v4.30.0, Accessed: 2025-11-23, 2025. [Online]. Available: https://github.com/dgunning/edgartools.

[10] FBS Analyst Team, *Faang stocks. fundamental analysis.* Accessed: 2026-01-19, Apr. 2023. [Online]. Available: https://fbs.eu/en/analytics/articles/faang-stocks-fundamental-analysis-29708.

[11] ExchangeRate, *The accurate & reliable exchange rate API*, https://www.exchangerate-api.com/, Accessed: 2026-01-19.

[12] S. Zhang, Z. Wang, R. Gao, and S. Li, "Deep context-dependent choice model," in *2nd Workshop on Models of Human Feedback for AI Alignment*, 2025. [Online]. Available: https://openreview.net/forum?id=bXTBtUjb0c.

[13] Y. Yang, Z. Wang, R. Gao, and S. Li, "Reproducing kernel hilbert space choice model," in *Proceedings of the 26th ACM Conference on Economics and Computation*, ser. EC '25, Stanford University, Stanford, CA, USA: Association for Computing Machinery, 2025, p. 819, ISBN: 9798400719431. DOI: 10.1145/3736252.3742630. [Online]. Available: https://doi.org/10.1145/3736252.3742630.

[14] Z. Lu and K. Shimizu, *Estimating discrete choice demand models with sparse market-product shocks*, 2025. arXiv: 2501.02381 [econ.EM]. [Online]. Available: https://arxiv.org/abs/2501.02381.

A Vélez-Pareja Model Outputs

Balance Sheet	**Year 0**	**Year 1**	**Year 2**	**Year 3**	**Year 4**
Cash	13.0	15.2	16.3	17.7	19.2
Accounts Receivable	0.0	19.0	20.4	22.2	24.0
Inventory	20.0	22.5	24.1	26.0	28.1
Accounts Payable (Prepaid)	27.1	28.9	31.3	33.8	0.0
Short-term Investments	0.0	0.0	0.1	0.0	0.0
Current Assets	60.1	85.5	92.1	99.8	71.3
Net Fixed Assets	45.0	45.4	46.4	47.3	47.3
Total Assets	**105.1**	**130.9**	**138.5**	**147.1**	**118.6**
Accounts Payable	0.0	27.1	28.9	31.3	33.8
Accrued Payables	37.9	40.8	44.4	48.0	0.0
Short-term Debt	22.1	0.0	0.0	0.0	0.0
Current Liabilities	60.1	67.9	73.3	79.3	33.8
Long-term Debt	31.5	35.5	31.6	28.1	35.2
Total Liabilities	**91.6**	**103.4**	**104.9**	**107.4**	**69.1**
Equity Investment	13.5	16.6	16.6	16.7	21.4
Retained Earnings	0.0	0.0	3.3	7.4	12.1
Current Year NI	0.0	11.0	13.7	15.6	16.1
Share Repurchases	0.0	0.0	0.0	0.0	0.0
Total Liabilities & Equity	**105.1**	**130.9**	**138.5**	**147.1**	**118.7**
Check	0.0	0.0	0.0	0.1	0.1

Table 11: Projected Balance Sheet

Cash Flow Statement	**Year 0**	**Year 1**	**Year 2**	**Year 3**	**Year 4**
Loan Inflows	−53.6	−7.1	−0.0	−0.4	−11.0
Principal Payments	0.0	25.3	3.9	3.9	3.9
Interest Payments	0.0	7.0	4.5	4.0	3.4
Cash Flow to Debt	−53.6	25.2	8.3	7.5	−3.7
Net Cash Borrowed	**53.6**	−25.2	−8.3	−7.5	**3.7**
Equity Investment	−13.5	−3.1	−0.0	−0.2	−4.7
Dividends	0.0	0.0	7.7	9.6	10.9
Share Repurchases	0.0	0.0	0.0	0.0	0.0
Cash Flow to Equity	−13.5	−3.1	7.7	9.5	6.2
Net Cash from Equity	**13.5**	**3.1**	−7.7	−9.5	−6.2
Combined Cash Flow	−67.1	22.1	16.0	16.9	2.5
Tax Shield	0.0	2.5	1.6	1.4	1.2
Free Cash Flow	−67.1	**19.7**	**14.5**	**15.5**	**1.3**

Table 12: Projected Cash Flow Statement