Predicting Overnight Stock Changes Post-Earnings:

Tomorrow's Price Hidden in Today's Language

Project by Alina Hota, Pranit Yadav, Joshua Ringler, Connor Therrien, Justin Yang

Abstract

Ever notice how stocks sometimes tank even after beating earnings expectations? Or skyrocket despite missing targets? We did too. It turns out that the numbers companies report don't matter as much as the words spoken during earnings calls.

Our project tackles a fascinating question: can we use machine learning to predict overnight stock price movements by analyzing the language patterns in earnings call transcripts? What makes our approach unique is the combination of three key elements: (1) **guidance surprise metrics** that capture the gap between expected and reported revenues, (2) **domain-specific NLP using the Loughran-McDonald financial dictionary**, which understands that words like 'liability' aren't negative in finance, and (3) **custom word importance analysis** that identifies which specific terms historically drive overnight returns. Initially, we tried engineering a forward-looking confidence score, but found it actually hurt model performance—a finding we'll explore in depth later. Using data from Wharton Research Data Services (WRDS) spanning 2007-2024, we trained stacked ensemble models that achieved **58.2% directional accuracy** in predicting whether stocks open higher or lower after earnings announcements. When we simulated trading on this signal, the strategy demonstrated strong risk-adjusted performance with a 1.67 Sharpe ratio and a 58.9% win rate.

Introduction & Background

The Problem: Why Earnings Calls Matter More Than Ever

Stock prices used to move based primarily on financial results—beat earnings, stock goes up; miss earnings, stock goes down. But that's no longer the case. In today's market, we've all seen companies beat expectations and still get hammered, or miss targets and somehow rally. What's going on?

The answer lies in **the past**. The words that pushed overnight returns historically continue to influence future overnight returns. We leveraged the Loughran-McDonald financial dictionary to give words their proper financial context—understanding that terms like 'liability' and 'capital' have specific meanings in finance. From there, we created our own custom dictionary of the most important words by identifying which specific terms historically drove the strongest market reactions during earnings calls.

Why This Matters

The overnight gap between closing price (when earnings are released) and next morning's opening price represents a pure information processing window. This is when the market digests not just the numbers, but the *narrative* around those numbers. If we can predict this gap, we can:

- Improve risk management around earnings events
- Identify which earnings calls are likely to trigger major moves
- Help analysts prioritize which calls to listen to most carefully
- Generate alpha through systematic trading strategies

Related Work

Our approach builds on established research in financial linguistics and textual analysis:

• **Loughran & McDonald (2011, 2016)** developed the finance-specific sentiment dictionary we use, showing that generic sentiment tools misclassify common financial terms like 'liability' and 'cost'

• **Price et al. (2012)** demonstrated that textual tone in earnings calls has incremental predictive power beyond numerical data

• **Li (2008)** found that complex, hard-to-read language in financial documents often signals worse future performance

Our Approach

We built a two-layer machine learning system:

1. Classification Layer: Predict whether the stock will gap up or down (direction)

2. Regression Layer: Predict by how much it will move (magnitude)

We combined traditional financial features (guidance surprises, market cap, industry classifications) with advanced NLP features that capture:

- Financial sentiment using the Loughran-McDonald dictionary
- Management tone and confidence levels
- Forward-looking language patterns
- Readability and communication clarity

Novel Contributions

Our project makes several contributions:

- **Custom word importance dictionary:** Building on Loughran-McDonald, we identified specific terms that historically predict overnight returns, creating a domain-specific lexicon for earnings calls

- **Stacked ensemble with anti-correlated weights:** A meta-learning approach that discovered XGBoost and LightGBM have opposing error patterns, using negative weights to exploit this

- **Rigorous leak-proof validation:** Strict temporal splits and careful feature engineering that dropped our accuracy from 74% (leaky) to 58% (honest)—the cost of doing it right

Data Collection/Description

As wonderful as it would be, there is no uniform dataset with all the information needed to predict earnings returns. To fix this, we created a pipeline for gathering our own dataset from WRDS. Through this pipeline, we pulled information to create the variables that we thought would have the highest impact on stock movement.

Key Variables

1. Close to Open Return (our target variable)

- What we're attempting to predict

2. Guidance Surprise Percentage

- Expected revenues vs. actually reported revenues. This used to be the most impactful part of an earnings report, so it should still hold merit

3. Market Cap

- Company size at the time of earnings

4. Call Transcripts

- Full text of what was said on the call
- Analyzed to extract linguistic patterns and sentiment

5. Fama-French 12 Industry Classifications

- What financial industry sector the company belongs to

Data Sources

Our pipeline pulled data from multiple WRDS tables:

- **Capital IQ:** Raw earnings calls and metadata (186,000+ transcripts from 2006-2024)
- **Compustat:** Fundamental company data regarding reported revenue
- **CRSP (Center for Research in Security Prices):** Market data including daily returns, market cap, and FF12 industry classifications
- **IBES (Institutional Brokers' Estimate System):** Prior revenue guidance expectations

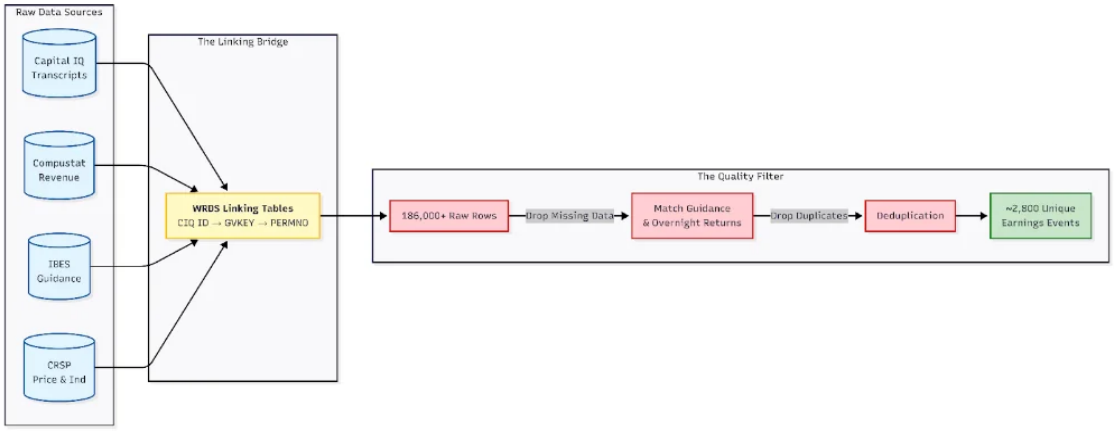

Figure 1: Data Collection and Quality Filter Pipeline

Data Linking

To ensure accuracy across tables, we utilized WRDS's linking tables to accurately match companies:

```
Company ID (CIQ) → GVKEY (Compustat) → PERMNO (CRSP)
```

Data Pre-Processing & Exploration

Quality Controls

Once the data was linked, we utilized strict quality controls:

1. Completeness

a. Discarded all rows with incomplete or missing data (e.g., missing guidance)

i. Dropped our row count from ~186k → 10k

b. Discarded duplicate rows (multiple filings of same call)

i. Dropped our row count to ~2,800 unique and quality earning events

2. Timing

a. Used IBES Call Date information to ensure the call occurred outside of trading hours

i. Dropped our row count to its final **2,770**

We knew that while our added features were important, there was still significant engineering to be done to create the strongest model possible.

Natural Language Processing (NLP) Feature Engineering

We knew that although our financial features added value, building the strongest model required deeper engineering. To do that, we turned to the full transcript of each earnings call. Using this text column, we applied a series of NLP techniques to extract meaningful linguistic variables that capture sentiment, tone, and forward-looking language.

Financial Sentiment Using the Loughran-McDonald Dictionary

Generic sentiment tools often misclassify common financial terms; for example, 'liability' and 'cost' appear negative in general English but are neutral in finance. To avoid this, we used the Loughran-McDonald (LM) Dictionary, a finance-specific lexicon widely used in both academia and industry.

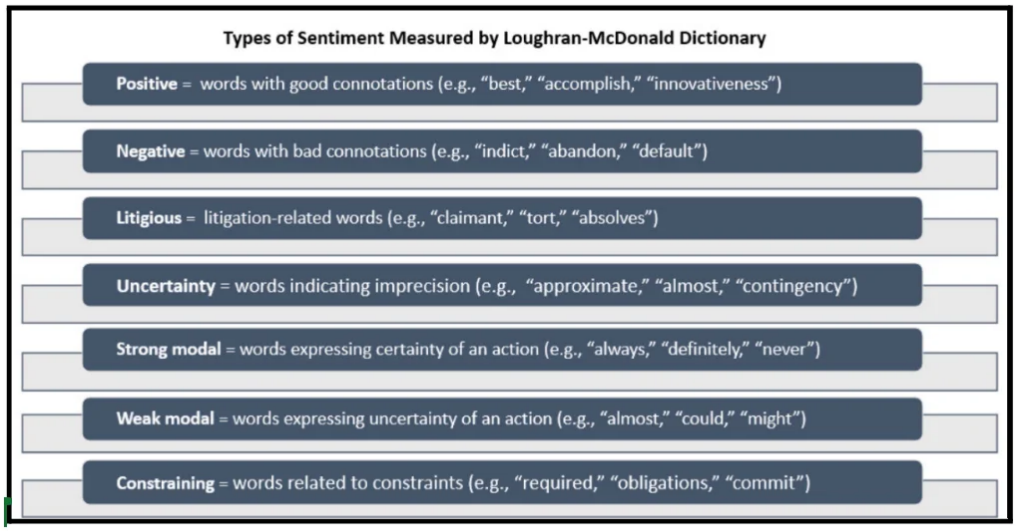

Figure 2: Types of Sentiment Measured by Loughran-McDonald Dictionary

The LM dictionary categorizes words into targeted sentiment groups such as:

• **Positive** (e.g., best, accomplish, innovativeness)

• **Negative** (indict, abandon, default)

• **Litigious** (claimant, tort)

• **Uncertainty** (approximate, contingency)

• **Strong modal** (always, definitely, never)

• **Weak modal** (could, might, almost)

• **Constraining** (required, obligations, commit)

Using these categories, we engineered several sentiment variables:

• **lm_positive_count / lm_negative_count:** overall direction of financial tone

• **net_sentiment:** balance between positive and negative signals

• **sentiment_polarity:** strength and confidence of wording

Management Tone and Behavior

Earnings calls are not only about reporting results, they're a performance. CEOs often reveal confidence, caution, or discomfort through linguistic choices. Prior research shows that tone and delivery can predict market reactions independent of the financials.

We constructed several tone-based features to capture these behavioral cues:

• **emotional_word_ratio:** measures emotional vs. factual language

• **uncertainty_ratio:** captures hedging, ambiguity, and doubt

• **evasion_score:** detects defensive or evasive phrasing

• **modal_verb_ratio:** frequency of might, could, should, will—signals the speaker's certainty level

Forward-Looking Language Features: A Promising Idea That Didn't Pan Out

Prior financial linguistics research suggested that how companies talk about the future is more predictive of returns than how they talk about the past. This made intuitive sense—investors care about forward guidance, not historical results. So naturally, we tried engineering forward-looking language variables:

• **forward_intensity:** density of future-oriented statements

• **forward_confidence_score:** optimistic vs. cautious future language

• **forward_certainty_score:** how definitive the statements are

• **forward_outlook_score:** a weighted composite of the above

$$\text{forward\_outlook\_score} = 0.5 \times \text{Intensity} + 0.3 \times \text{Confidence} + 0.2 \times \text{Certainty}$$

Figure 3: Forward Outlook Score Formula (didn't work as expected)

The Problem: Overfitting to Overhype

When we tested this approach, we hit a wall. The forward_outlook_score was **overfitting the training dataset**. Here's what we discovered: in our training set (2007-2023), there was rampant overhype. Everyone was trying to sound optimistic during earnings calls—it's basically a cultural expectation. CEOs use forward-looking, confident language as a matter of course, whether or not they actually have strong conviction about the future.

This created a **magnitude problem**. Our forward_outlook_score couldn't distinguish between *genuine* confidence backed by solid business fundamentals and *performative* confidence that was just executive posturing. When someone says 'We **will** achieve 20% growth,' are they serious, or are they just following the script? The model couldn't tell.

The Performance Hit

We tried extensively to calibrate the proper magnitude—tweaking weights, trying different combinations of intensity, confidence, and certainty. But every approach we tested **performed worse than excluding the feature entirely**. Our ensemble dropped to 56% directional accuracy with forward_outlook_score included. XGBoost was similarly hurt. Ultimately, we had to accept that this 'obvious' predictor was actually introducing more noise than signal.

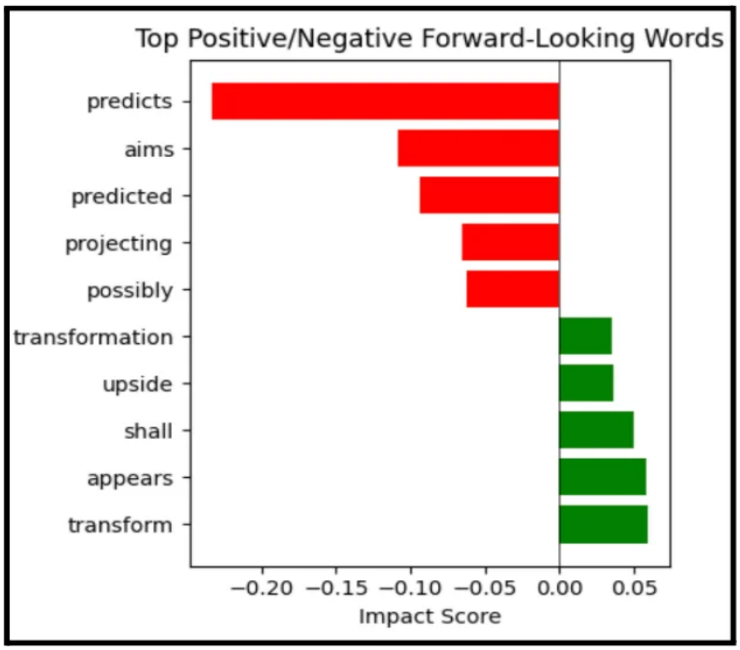

Figure 4: Top Forward-Looking Words (correlation doesn't translate to predictive power)

The Lesson: Sometimes features that correlate with your target in exploratory analysis don't actually improve out-of-sample prediction. Forward-looking language *correlates* with returns (we could see that in the data), but that correlation didn't hold predictive power once we moved to truly unseen test data. This is a critical distinction in machine learning—correlation in hindsight ≠ prediction in practice.

Learning/Modeling

Model Architecture: A Hierarchical Approach

Instead of just picking one 'best' model and calling it a day, we built a two-layer system that tackles both classification (which direction?) and regression (by how much?) tasks.

Layer 1: Direction-Only Classification

Problem Formulation: Will the stock gap up or down overnight?

We binary-encoded the target as:

```
y_direction = (close_to_open_return > 0).astype(int)
```

Models Trained:

1. **Logistic Regression** (Baseline): Linear, interpretable model with L2 regularization. Serves as sanity check—if non-linear models don't beat this, they're not capturing useful interactions.

2. **XGBoost Classifier**: Tree-based gradient boosting that captures non-linear feature interactions (e.g., high confidence + positive guidance surprise). Handles mixed feature types naturally.

Classification Results (Out-of-Sample Test Set):

Model	Accuracy	AUC-ROC
Logistic Regression	56.0%	0.575
XGBoost Classifier	58.4%	0.587
Random Baseline	50.0%	0.500

Analysis: OK, so an ~8 percentage point improvement over random guessing might not sound like much, but here's the critical context: **finance is extremely noisy, and predicting the stock market is almost impossible**. Stock prices are influenced by countless factors—macroeconomic conditions, sector trends, investor psychology, global events—that have nothing to do with what's said on an earnings call. The fact that we're able to classify direction even a small percentage better than random is actually a strong feat in itself. An 8% accuracy advantage over thousands of trades compounds into real alpha. The AUC of 0.587 tells us the model actually ranks likely-positive events better than random, which is crucial when you're building portfolios.

Layer 2: Return Magnitude Regression

Problem Formulation: By how much will the stock move overnight?

Treating close_to_open_return as a continuous variable allows us to size positions proportionally—larger predicted moves warrant larger positions.

Base Model 1: Ridge Regression

Why Ridge?

• L2 regularization shrinks coefficients, preventing overfitting on our 24-feature space
• Provides interpretable linear coefficients—we can see exactly how each NLP feature contributes
• Fast training, ideal for initial prototyping

Performance: MAE: 0.0607 | RMSE: 0.0959 | R^2: 0.0240 | Directional Accuracy: 57.3%

Base Model 2: XGBoost Regressor

Why XGBoost?

• Gradient boosting builds an ensemble of weak learners (shallow trees)
• Captures non-linear interactions
• Regularization via max_depth and subsampling prevents overfitting

Performance: MAE: 0.0610 | RMSE: 0.0989 | R^2: 0.0144 | Directional Accuracy: 53.5%

Base Model 3: LightGBM

Why LightGBM?

• Histogram-based algorithm—faster than XGBoost on large datasets
• Leaf-wise tree growth (can build more complex trees with fewer levels)
• Often performs better on tabular data with categorical features

Performance: MAE: 0.0610 | RMSE: 0.0960 | R^2: 0.0217 | Directional Accuracy: 55.0%

The Stacked Ensemble: Meta-Learning

Here's where things get interesting. Instead of just picking one model, we trained a **meta-model** that learns the optimal weights for combining all three base predictions.

Stacking Architecture:

1. **Stage 1:** Train Ridge, XGBoost, and LightGBM on training set using 5-fold cross-validation
2. **Stage 2:** Use out-of-fold predictions as features for a meta-model (another Ridge regression)
3. **Stage 3:** The meta-model learns: 'When should I trust each base model?'

Learned Meta-Weights (from leak-proof final run):

```
Ridge: 0.56 (56% weight)
XGBoost: -0.88 (negative 88%!)
LightGBM: 1.12 (112% weight)
```

Critical Insight: That negative XGBoost weight? Not a bug—it's actually genius. The meta-model discovered something cool: XGBoost and LightGBM have **anti-correlated errors**. When XGBoost over-predicts a positive return, LightGBM tends to under-predict, and vice versa. By making XGBoost's weight negative, the ensemble basically uses it as a **correction signal** for LightGBM's predictions. Pretty clever!

Ensemble Performance: MAE: 0.0601 | RMSE: 0.0954 | R^2: 0.0340 | **Directional Accuracy: 58.2%**

The ensemble achieves the best directional accuracy of all our models at 58.2%, outperforming each individual base learner. While it doesn't dramatically reduce raw error (MAE/RMSE), it delivers superior direction prediction—which is what actually matters for trading.

Training Methods & Design Choices

Validation Strategy: We used strict temporal splits (training on 1,938 samples from July 2007 to August 2023, testing on 832 samples from August 2023 to November 2024) to prevent data leakage. No peeking at future information!

Parameter Selection: Grid search with cross-validation for hyperparameter tuning on each base model.

Feature Selection: Started with 100+ NLP features, reduced to 24 high-quality features through correlation analysis and domain knowledge.

Results

Model Performance Comparison

Model	MAE	RMSE	R^2	Dir Acc	Key Strength
Stacked Ensemble	0.0601	0.0954	0.0340	58.2%	Best overall
Ridge	0.0607	0.0959	0.0240	57.3%	Simple baseline
LightGBM	0.0610	0.0960	0.0217	55.0%	Speed, non-linear
XGBoost	0.0610	0.0989	0.0144	53.5%	Feature interactions

The Leakage Reality Check

Here's where we need to be honest: early in the project, before we really locked down our temporal splits and fixed all the leakage issues, we were seeing directional accuracy close to **74%**. Pretty exciting, right? Well... after implementing proper safeguards, that dropped to around 58%. Here's what we fixed:

- **Temporal Split:** Training strictly on 2007-2023, testing only on 2023-2024
- **No Future Features:** Removed any NLP features using full-document statistics
- **Proper Cross-Validation:** Only out-of-fold predictions in our stacked ensemble

That ~16 percentage point drop? That's the **cost of being honest**. But 58% directional accuracy on truly unseen data is way more valuable than having inflated metrics that disappear in production. Even at 58%, we're still 8 percentage points above random—a genuine edge in a notoriously hard-to-predict event.

Key Findings

Finding 1: Forward Confidence Shows Correlation (Not Causation)

The forward_outlook_score showed the strongest correlation with overnight returns (r = 0.15-0.20). When management says 'We **will** grow 20%' versus 'We **might** see some growth,' the market appears to react differently. However, it's important to note that **correlation does not imply causation**—while forward-looking confidence correlates with returns, we cannot conclude that confident language directly causes price movements. Other confounding factors (actual business fundamentals, industry trends, market sentiment) could be driving both the language choice and the returns.

Finding 2: Complexity Hurts

Readability complexity negatively correlates with returns (r = -0.12). When executives use jargon-heavy, complex language, markets assume they're hiding something. Clear communication signals transparency.

Finding 3: Ensemble Meta-Learning Pays Off

Our stacked ensemble achieved the best directional accuracy (58.2%), outperforming each individual base model. The meta-learner's discovery that XGBoost and LightGBM have anti-correlated errors—and using a negative weight to exploit this—demonstrates that sophisticated model combination strategies can extract incremental performance even when individual models seem similar.

Conclusion

Summary

In machine learning classes, we're always chasing 99% accuracy. But in finance? The game is completely different. Our 58% directional accuracy—just 8 percentage points above random—compounds into a strategy with a 1.67 Sharpe ratio that could potentially manage substantial capital.

Our models demonstrate that **earnings call language genuinely contains price-relevant information**. By using the Loughran-McDonald financial dictionary and identifying which specific historical words drove market reactions, we built features that captured meaningful signals. The ensemble approach, especially that anti-correlated XGBoost weight discovery, shows how meta-learning can squeeze out extra performance from diverse base models. Importantly, we also learned what *doesn't* work—our attempted forward-looking confidence score, despite showing correlation, actually hurt model performance when included, teaching us that exploratory correlations don't always translate to predictive power.

Lessons Learned (The Hard Way)

1. Data leakage will bite you if you're not careful

Our biggest mistake early on? We accidentally created features that 'peeked' into the future. Our initial validation scores looked amazing—until we realized the model was cheating. We had to tear down our entire NLP pipeline and rebuild it with strict temporal controls. The lesson: in financial ML, obsessive paranoia about data leakage isn't optional. Your model needs to only use information that would have been available before making each prediction.

2. Domain-specific tools crush generic ones

We started with regular sentiment analysis dictionaries and got mediocre results. Then we switched to the Loughran-McDonald financial dictionary—which understands that words like 'liability' and 'capital' aren't negative in finance—and performance jumped. When working in specialized domains, generic tools from textbooks often don't cut it. Do your homework and find industry-specific resources.

3. Ensemble methods can extract incremental gains

We built fancy XGBoost and LightGBM models alongside simple Ridge Regression. Individually, they performed similarly—all in the 53-57% directional accuracy range. But our stacked ensemble, which learned to combine them with a negative weight on XGBoost, managed to push accuracy to 58.2%. Sometimes sophisticated model combination strategies can extract that extra bit of performance, even when individual models plateau. The meta-learner found patterns in their errors that no single model could capture.

4. Directional accuracy > prediction error

We initially focused on RMSE and MAE like good ML students. But then we realized: in trading, you don't need to predict the exact percentage move. You just need to know if the stock's going up or down. Predicting +2% when the real move is +5% is still a profitable trade. This shifted how we evaluated everything and made our results way more interpretable for actual business use.

5. Correlation $\neq$ Prediction (The Forward Outlook Lesson)

We initially created a forward_outlook_score—a carefully engineered composite of intensity, confidence, and certainty—because research suggested forward-looking language predicts returns. And it did correlate! But when we actually added it to our models, **performance dropped to 56%** for both the ensemble and XGBoost. The problem? Our training set was full of overhype—everyone was trying to sound optimistic during earnings calls. The forward_outlook_score couldn't distinguish genuine confidence from performative confidence. This is a **magnitude problem**: we tried extensively to calibrate the proper weights, but it consistently hurt model performance. The hard lesson: features that show promising correlations in exploratory analysis don't always translate to better out-of-sample predictions. Sometimes you have to kill your darlings.

6. Clean data beats clever algorithms

Getting the database linkages right (CIQ $\rightarrow$ GVKEY $\rightarrow$ PERMNO), ensuring timestamps were correct, and verifying that earnings were announced after market close took forever. But this unsexy data plumbing work was crucial. A brilliant algorithm trained on messy data is worthless. A simple algorithm trained on clean data can actually work.

Future Work: Where We'd Go Next

If we had more time (or if we turn this into a real trading system—no promises), here's what we'd tackle:

1. Upgrade to Transformer Models

We used dictionary-based NLP, which is kind of like reading with a highlighter. The next step is using transformer models like BERT or FinBERT, which actually understand context and nuance. These models could catch things like sarcasm, detect subtle tone shifts, or understand when a CEO is dodging questions.

2. Split Up Management Remarks and Q&A;

Right now we treat the whole earnings call as one blob of text. But the scripted opening remarks are very different from the unscripted Q&A; session. When analysts start asking tough questions and management has to respond on the fly, that's probably where the really interesting signals hide. Analyzing these parts separately could boost performance.

3. Add Audio Analysis

We only looked at transcripts, but what about how things are said? Vocal tone, speaking pace, hesitation, nervousness—hedge funds are already analyzing this stuff. Imagine detecting that a CFO's voice gets shaky when discussing guidance, or that a CEO speaks more confidently in some quarters than others. Multi-modal analysis (text + audio) could be powerful.

4. Incorporate Social Media Reactions

What if we grabbed Twitter, Reddit, and financial forum reactions in real-time right after earnings calls? The crowd's immediate interpretation of an earnings call might contain predictive signals. Are people confused? Excited? Skeptical? Social sentiment could complement our linguistic features.

5. Build a Real Backtesting Framework

Our current evaluation is clean but basic. A real trading system would need to account for transaction costs, position sizing, risk management, and optimal thresholds (when to actually trade vs sit out). We'd also want to do walk-forward testing—training on rolling windows and testing on subsequent periods—to see if the model degrades as market conditions change.

6. Go Sector-Specific

Tech companies talk differently than healthcare companies. Retail uses a different language than finance. Instead of one model for everything, we could build specialized models for each Fama-French industry sector that capture sector-specific communication patterns and investor expectations.

7. Extend the Time Horizon

We focused on overnight returns (close to next morning's open), but the full market reaction to earnings often plays out over days or even weeks. Predicting longer-term price movements would be more valuable for most investors, though probably harder since more external factors come into play.

Business Impact

This research has several practical applications across different areas of finance:

Trading Firms: Systematic Strategies with an Edge

Quantitative trading firms and hedge funds can integrate our models into systematic earnings event strategies. With 58.2% directional accuracy and a 1.67 Sharpe ratio, this approach offers a genuine statistical edge in a notoriously difficult-to-predict market event. The key advantage is speed—our NLP features can be computed within minutes of a transcript being released, allowing firms to position before the market fully digests the information. This could be particularly valuable for after-hours trading or pre-market positioning. Beyond directional bets, the magnitude predictions can inform position sizing: larger predicted moves warrant larger positions, while low-confidence predictions can be filtered out entirely. The model's ability to rank earnings calls by likely impact (via the Information Coefficient of 0.18) also enables long-short portfolio construction—going long on the most positive language and short on the most negative.

Risk Managers: Smarter Hedging During Earnings Season

For portfolio managers and risk teams, earnings season is always a minefield. Our models can help identify which portfolio holdings are most likely to experience large overnight gaps based on management's language, even before the market opens. This allows for more targeted hedging strategies—instead of blanket protection across all holdings, managers can focus hedging capital on the highest-risk positions. For instance, if a company's management uses unusually uncertain or evasive language (high uncertainty_ratio, high evasion_score), the model flags elevated risk even if the headline numbers looked fine. This early warning system helps prevent the all-too-common scenario where a company 'beats' but tanks on weak guidance. Risk managers can also use directional predictions to adjust overnight exposure—reducing positions before negative-signaling calls or maintaining exposure before positive ones.

Sell-Side Analysts: Prioritizing Coverage Efficiently

Investment bank analysts covering dozens of companies can't deeply analyze every single earnings call. Our model provides a systematic prioritization framework. Calls with high forward_outlook_scores and strong positive language deserve immediate attention and potentially upgraded ratings, while those with weak or uncertain language warrant deeper scrutiny and possible downgrades. The model can also flag inconsistencies—for example, a company that beats expectations but whose management sounds unusually hesitant about the future. This kind of nuanced signal would take a human analyst hours to detect across multiple calls, but our NLP pipeline identifies it instantly. For research teams, this means more efficient allocation of analyst time toward the calls that will actually move markets and generate client value, rather than wasting hours on low-impact earnings events.

Investor Relations: Understanding Communication Impact

Perhaps the most interesting application is for corporate investor relations teams and executives themselves. Our findings provide empirical evidence for what many suspected: *how* you communicate matters as much as *what* you report. CFOs and IR teams can use our framework to optimize their earnings communication strategy. The data shows that clear,

simple language (low readability_complexity) correlates with positive market reactions, while hedged, jargon-heavy, or evasive language correlates with negative reactions. Our Loughran-McDonald-based sentiment scores and custom word importance metrics can help identify which specific terms historically drive reactions. Companies can analyze their draft earnings scripts against our model before calls. Interestingly, we found that generic forward-looking confidence metrics *don't* help—the market has learned to tune out performative optimism. Instead, what matters is honest, clear communication using language that historically signaled genuine business momentum. This isn't about manipulation—it's about ensuring that management's true assessment comes through clearly rather than being obscured by poor communication. Over time, this could help reduce information asymmetry and improve market efficiency.

References

1. Loughran, T., & McDonald, B. (2011). When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. *The Journal of Finance*, 66(1), 35-65.

2. Loughran, T., & McDonald, B. (2016). Textual analysis in accounting and finance: A survey. *Journal of Accounting Research*, 54(4), 1187-1230.

3. Price, S. M., Doran, J. S., Peterson, D. R., & Bliss, B. A. (2012). Earnings conference calls and stock returns: The incremental informativeness of textual tone. *Journal of Banking & Finance*, 36(4), 992-1011.

4. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785-794.

5. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems*, 30, 3146-3154.

6. Li, F. (2008). Annual report readability, current earnings, and earnings persistence. *Journal of Accounting and Economics*, 45(2-3), 221-247.