Adaptive Explainable Multi-Agent Intelligence for Heterogeneous Financial Decision Fusion: A LangGraph Framework with Dynamic Confidence-Aware Coordination

Petar Canoski[1⋆], Borjan Gjorgievski[1],
Martina Toshevska[1], Slobodan Kalajdzhiski[1], and Sonja Gievska[1]

Faculty of Computer Science and Engineering (FINKI),
Ss. Cyril and Methodius University in Skopje, North Macedonia
{petar.canoski,borjan.gjorgievski}@students.finki.ukim.mk
{martina.toshevska,slobodan.kalajdzhiski,sonja.gievska}@finki.ukim.mk
⋆ Corresponding author.

Abstract. Real-world financial decisions rarely rest on one information source: price action, narrative sentiment, macroeconomic policy, and systemic risk all carry partial, heterogeneous evidence, and no single monolithic model can absorb all of it while remaining interpretable. This paper presents an adaptive, explainable multi-agent framework for heterogeneous financial decision fusion. Its contributions are primarily architectural: a model-agnostic typed output contract that lets structurally dissimilar agents be fused by a single rule; a three-tier graceful-degradation cascade so that missing data or unavailable models reduce confidence rather than halt the pipeline; and *uncertainty-driven routing*, in which each agent's contribution is weighted, and where warranted gated, by its own confidence and the risk state of the environment. We instantiate the framework on Bitcoin trading signal generation with a Technical Analysis Agent (two-stage CNN-LSTM), a Sentiment and Macro Agent (FinBERT with LLM-based reinterpretation), and a Risk and Volatility Agent (gradient-boosted model over on-chain and geopolitical signals). We further introduce an *LLM-as-a-Judge* protocol that scores each agent's reasoning at production-quality inference, and show that it surfaces concrete data-pipeline defects that aggregate backtest metrics do not reveal. An incremental ablation indicates that maximum drawdown improves only when the sentiment and risk modalities act jointly, not when either is added alone. We report the trading figures as *in-sample* case-study evidence under an explicitly stated accounting convention (Section 10); the architectural contributions are independent of them.

Keywords: Multi-agent systems · Heterogeneous decision fusion · Uncertainty-driven agent routing · Explainable AI · Financial NLP · Bitcoin trading (case study)

1 Introduction

Financial decision-making rarely rests on a single information source: equities, foreign exchange, insurance underwriting and cryptocurrency markets all present heterogeneous evidence streams differing in modality, frequency and reliability. A model ingesting all of them jointly must either force them into a shared representation, discarding modality-specific structure, or disentangle them internally with no guarantee a human can inspect how — while regulators increasingly require AI-driven financial recommendations to be explainable [2]. This paper asks: how should such a problem be decomposed into modality-specialised agents, and how should their partial, unequally reliable outputs be combined?

We use Bitcoin trading signal generation as the case study, since it offers dense structured price data, a genuine sentiment channel, explicit on-chain and geopolitical risk signals, and verifiable ground truth through realised returns. Bitcoin's price is shaped simultaneously by technical supply-and-demand dynamics, macroeconomic signals, narrative sentiment, on-chain behaviour, and geopolitical shocks [11,12] — an unusually complete instance of the general problem. Multi-agent system (MAS) design addresses it by decomposing the problem along its semantic boundaries, assigning each sub-problem to a specialised agent, and integrating partial outputs through a coordination layer producing a natural-language rationale [15].

This paper presents an adaptive, explainable multi-agent framework for heterogeneous financial decision fusion, instantiated end-to-end as *Bitcoin Multi-Agent Intelligence for Trading Signals*[1]: three modality-specialised agents feed a coordinator that performs uncertainty-driven routing — weighting, and where warranted gating, each agent's contribution by its own confidence and the prevailing risk state — implemented concretely as a LangGraph-orchestrated, typed state graph. The system is exercised on seven years of hourly BTC/USDT data (March 2019 – February 2026).

The main contributions of this paper are as follows:

1. A domain-general, modular multi-agent architecture for heterogeneous financial decision fusion, in which a *model-agnostic typed output contract* allows agents with structurally dissimilar internals — a deep sequence model, an LLM-augmented transformer classifier, and a gradient-boosted tree — to be combined by one fusion rule rather than by bespoke per-agent integration code.
2. An *uncertainty-driven agent routing* mechanism in which each agent's self-reported confidence and the coordinator's assessed risk state jointly determine how much weight that agent's signal receives, up to and including a full veto, together with a three-tier graceful-degradation cascade that converts data and model failures into reduced confidence rather than pipeline failure.
3. An *LLM-as-a-Judge* evaluation protocol for multi-agent systems that scores per-agent reasoning quality on live production inputs, and which in our de-

[1] Code: `https://github.com/petarcanoski/Bitcoin_Multi-Agent_Intelligence_for_Trading_Signals`

ployment surfaced two concrete data-pipeline defects that aggregate backtest metrics did not reveal.
4. An incremental agent-combination ablation on the Bitcoin case study, indicating that the reduction in maximum drawdown appears only when the sentiment and risk modalities are fused jointly, and not when either is added to the technical baseline alone.

We hypothesise that *agent specialisation reduces information interference between heterogeneous modalities and therefore improves robustness under market regime shifts*: a monolithic model must resolve competing gradients from price, sentiment, and risk signals internally, with no guarantee that a regime shift in one modality does not corrupt its response to an unrelated one, whereas decomposing the problem confines each modality's noise to its own agent and routes cross-modality influence only through an explicit, inspectable fusion step. Section 8.4 reports the evidence we have for this hypothesis, and Section 10 states plainly what that evidence can and cannot establish.

2 Related Work

LSTM networks have been widely applied to OHLCV (Open, High, Low, Close, Volume) time series [7] and remain competitive baselines; Temporal Convolutional Networks (TCN) [3] and Temporal Fusion Transformers (TFT) [9], the latter building on the transformer [14], have since been proposed for financial sequence modelling. All optimise a single model over a purely price-derived feature set and are silent on sentiment, macroeconomic context and on-chain risk; when such a model is wrong, an operator has no decomposable explanation of which source misled it.

FinBERT [1], a BERT (Bidirectional Encoder Representations from Transformers)-based [6] model further trained on a financial corpus and fine-tuned on the Financial PhraseBank [10], substantially outperforms general-purpose sentiment classifiers on financial text. Large language models (LLMs) such as GPT-3 [4] and LLaMA [13] extend this capability with contextual reasoning over compound, ironic, or forward-guidance language — a gap raw classification scores cannot fill. This line of work solves the complementary half of the problem — extracting a reliable sentiment signal from text — but is rarely wired into a decision pipeline that also reasons about price action or systemic risk.

Multi-agent approaches to algorithmic trading have a long history in market simulation and strategy design [15], and the availability of LLM-powered agents and orchestration frameworks such as LangChain [5] and LangGraph [8] has produced a rapidly growing body of LLM-agent trading systems. FinMem [17] introduces layered memory with explicit character design; FinAgent [18] adds multimodal inputs and dual-level reflection; and TradingAgents [16] organises LLM agents into the specialised roles of a trading firm — analysts, bull and bear researchers, a risk-management team, and a trader — coordinated through structured debate. This is the closest prior work to ours, and it establishes role specialisation and risk oversight as effective design patterns. Our framework

differs in two complementary respects. First, these systems realise specialisation through prompting and role assignment over a shared LLM backbone; ours composes *structurally heterogeneous* agents, each with a different learned model class chosen on the merits of its modality — which is what makes an explicit model-agnostic output contract necessary. Second, their coordination is realised through debate, reflection or memory retrieval, whereas ours applies an explicit, inspectable routing rule keyed on per-agent confidence and assessed risk. We do not claim superior trading performance relative to these systems: we have not run a controlled comparison, and Section 10 explains why our figures would not support such a claim.

Explainability work in finance remains dominated by *post-hoc* attribution such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations), applied after a black-box model has produced a prediction, rather than *intrinsic* explanations generated as part of the decision process. Ours takes the intrinsic approach: every agent emits a natural-language justification alongside its signal, which the Coordinator synthesises into a human-readable rationale.

3 System Architecture and Methodology

The pipeline consists of three independent specialist agents and one coordinator. Figure 1 shows the data flow.

Design Principles. The architecture follows four principles. (i) *Specialisation* — each agent handles one modality, so its model can be chosen on merit (a sequence model for price, a language model for text, a gradient-boosted tree for tabular risk) rather than forced into one architecture; this confines each modality's noise to its own agent, so cross-modality *information interference* (Section 1) can enter only through the Coordinator's explicit fusion step. (ii) *Explainability* — every output carries a natural-language justification as a first-class field, so the final rationale traces back to specific agent evidence. (iii) *Resilience* — every external dependency has a fallback, so partial failures degrade confidence rather than halt the pipeline. (iv) *Modularity* — any agent can be retrained or replaced without modifying the others, since agents communicate only through a shared, versioned contract.

The overall shape — specialist workers writing into shared state that a coordinator reads — is a standard multi-agent pattern [15], not a novel topology, and the role decomposition parallels LLM-agent trading systems [16]. Our contribution is not the topology but what travels along it: the typed contract (Section 3.1), the degradation cascade (Section 3.2), and the routing rule (Section 3.3).

3.1 Data Backbone, Pipeline Flow, and Output Contract

All three specialists are independently runnable Python modules exposing a `run()` entry point returning a structured, JSON-serialisable signal object. The

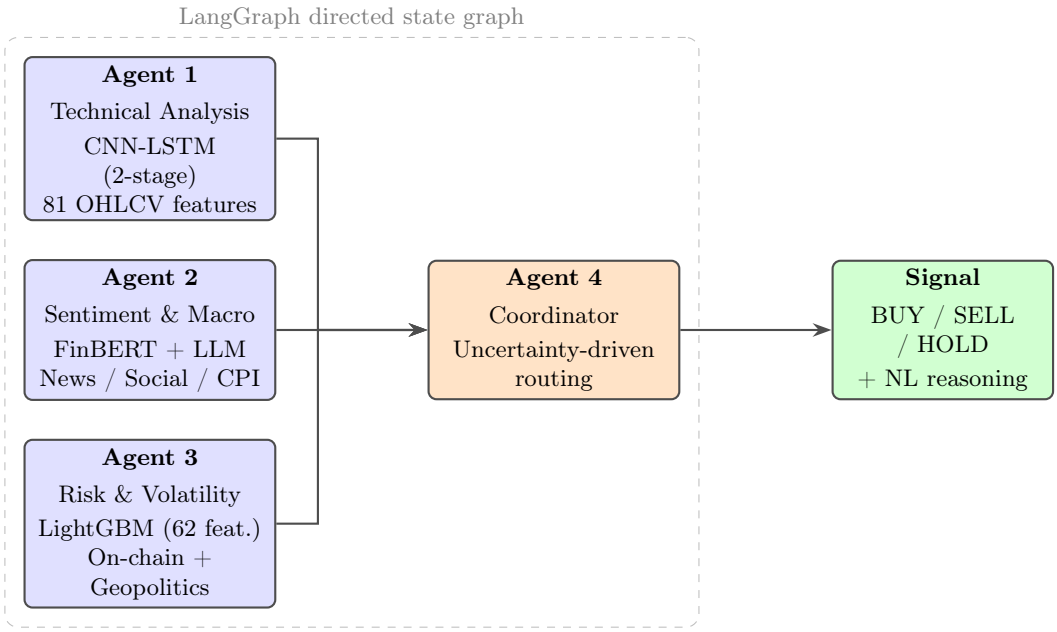

Fig. 1. System architecture. Agents 1–3 operate independently and forward typed signal objects to Agent 4 (Coordinator), which fuses them and emits an explainable recommendation.

Technical and Risk agents share one feature backbone: an hourly OHLCV table and a permutation-importance ranking (Section 4.2) computed once and re-used to select input columns for both the CNN-LSTM models and the Risk Agent's LightGBM classifier. The Coordinator invokes the Technical Agent's PyTorch inference in-process and the other two as subprocesses emitting JSON on `stdout`, letting each depend on its own library stack without version conflicts.

Every agent, whatever its internals, returns a typed schema carrying a discrete `signal` $\in$ {`BUY`, `SELL`, `HOLD`} (or {`LOW`, `MEDIUM`, `HIGH`} for the Risk Agent), a continuous `score`/`confidence`, `key_factors`, a `reasoning` string and `data_sources`. This schema is our own design, not an adopted standard: interchange standards exist for market data and trade messaging, but we are aware of none for inter-agent decision messages carrying confidence and free-text justification. Its practical value is that the Coordinator can fuse heterogeneous agents with one model-agnostic rule (Section 7), and any agent can be replaced without touching the coordinator.

3.2 Common Execution Pattern and Resilience

Each specialist follows the same five-stage pattern — fetch, analyse, aggregate sub-scores into one weighted score, threshold, explain — with confidence

grounded differently per agent (softmax margin, mean component confidence, retrieved-data volume) but an identical shape. Each implements the same three-tier fallback: the Risk Agent tries its LightGBM classifier, falls back to a heuristic recombining the same weights on directly fetched metrics, and returns a safe `MEDIUM_RISK` default at confidence 0 if that also fails, rather than raising; the Sentiment Agent mirrors this. The Coordinator therefore always receives a well-formed output, so the fusion logic can apply unconditionally.

3.3 Uncertainty-Driven Agent Routing

The scientific contribution of the coordination layer is the *routing rule* it applies, not the orchestration tool used to implement it: instead of always combining the three agents' signals with the same context-blind formula, the Coordinator treats each agent's confidence and the Risk Agent's assessed risk level as first-class inputs that determine how much weight — including zero — a given signal receives. This works in two layers: confidence enters the weighted-fusion score directly (Section 7), so a low-confidence output automatically contributes less; and an explicit *risk veto* overrides the fused decision to `HOLD` whenever the Risk Agent reports high uncertainty and the fused signal is not strong enough to act despite it.

We implement this as a LangGraph `StateGraph`: each node is one agent invocation, and shared typed state accumulates outputs so the Coordinator has the full history when it runs. The fixed node order (Technical, Sentiment, Risk, Coordinator) is chosen for deterministic tracing, not any data dependency, preserving the conceptual independence of Figure 1; a closed-form *legacy* mode evaluates the same rule without the LangGraph runtime. LangGraph is a convenient state-management library here, not a research contribution. The fusion weights are fixed, hand-calibrated constants — transparent and auditable, but not adaptive if an agent's relative reliability drifts.

4 Technical Analysis Agent

This section covers the price-action specialist: data, features, model, and training conditions.

4.1 Dataset and Feature Engineering

We use hourly BTC/USDT candlestick data from Binance spanning March 2019 to February 2026 — approximately 61,000 bars over seven years. The data are split temporally into training (2019-03-02 to 2024-01-19; 42,700 bars), validation (2024-01-19 to 2025-02-04; 9,103 bars), and test (2025-02-05 to 2026-02-20; 9,133 bars) sets with no overlap.

From the raw OHLCV series we engineer 128 features spanning five categories: *price action* (log returns, candle body/wick ratios, distance from rolling

highs/lows), *momentum* (RSI [Relative Strength Index], MACD [Moving Average Convergence Divergence], Stochastics, Williams %R, EMA [Exponential Moving Average] crossovers), *volatility* (ATR [Average True Range], Bollinger Bands, historical volatility), *volume* (OBV [On-Balance Volume], CMF [Chaikin Money Flow], MFI [Money Flow Index], VWAP [Volume-Weighted Average Price] ratio), and *higher-timeframe context* (4h and 1d EMA/ATR/RSI features). These 128 features are not the product of expert consultation or automated search: they are the standard indicator families of the technical-analysis literature, each instantiated at several lookback lengths (e.g. ATR at 7, 14 and 21 bars) across three timeframes. The intent was to over-generate a superset and prune it empirically (Section 4.2) rather than commit to a hand-picked subset in advance. All features are rolling z-score normalised over a 200-bar window.

4.2 Permutation Importance Feature Selection

To prune the feature set we apply permutation importance: each feature column is independently shuffled and the change in cumulative backtest P&L recorded, treating features whose removal *improves* P&L as spurious. We drop the 47 features with negative importance, retaining **81**. *This selection was performed against the test window rather than the validation window*, which is why the reported figures are in-sample with respect to feature selection (Section 10).

The five highest-scoring features are, in descending order, the 21-, 14- and 7-bar *Average True Range* — realised volatility over roughly one day, half a day and a third of a day — followed by two *higher-timeframe trend-position* features: the percentage distance of price from its 9- and 50-period exponential moving averages computed on the daily rather than hourly series. The model therefore relies most on how volatile the market currently is, and on where price sits relative to its short- and medium-term daily trend. The same 81 features are re-used by the Risk Agent's LightGBM (Light Gradient-Boosting Machine) model, so both learned components share one definition of signal versus noise.

4.3 Two-Stage CNN-LSTM Architecture and Training

The agent uses two independently trained models:

Stage 1 — Trade Detection. A binary classifier (trade / no-trade) determines whether the current market conditions are strong enough to justify entering a position. A prediction above threshold $\theta = 0.45$ triggers the direction model. This is an *operating point*, not a probability calibration: 0.5 would be the natural cut only if the classes were equally frequent and equally costly, which they are not. Lowering θ makes the filter more permissive; its value was selected by search (Section 8.3).

Stage 2 — Direction Prediction. A binary classifier (long / short) maps the same feature window to trade direction. The probability of the long class $p_{\text{long}} \geq 0.5$ yields a BUY signal; otherwise SELL.

Both models share the same CNN (Convolutional Neural Network)-LSTM (Long Short-Term Memory) backbone: three Conv1D blocks ($[64, 128, 256]$ channels, kernel size 3, BatchNorm, ReLU, Dropout) followed by a two-layer LSTM ($h = 256$) and a fully-connected classification head. The Stage 2 model additionally includes a regression branch for exit-magnitude estimation. Input windows of **30 bars** were chosen by the search in Section 8.3. The remaining structural hyperparameters — channel progression, kernel size, hidden width, dropout and optimiser settings — were not individually tuned for this task; they follow conventional defaults for one-dimensional convolutional-recurrent sequence models, and we make no claim that they are optimal. Both models are trained with AdamW (lr $= 10^{-3}$, $\lambda = 10^{-5}$), cross-entropy loss with inverse-frequency class weighting, ReduceLROnPlateau scheduling (patience 7), and early stopping (patience 15); Stage 2 adds a weighted MSE (Mean Squared Error) regression loss ($\lambda_{\text{reg}} = 0.3$).

4.4 Results

Table 1 reports Stage 1 and Stage 2 classification metrics. The trade detection model achieves 64.2% accuracy; when market conditions pass the filter, the direction model achieves 66.5%. Note the asymmetry in Stage 1: recall on the `trade` class is essentially 1.0 while recall on `no-trade` is 0.128, i.e. the filter admits almost every bar and rejects few. The stage is therefore better understood as a weak permissive gate than as a selective detector.

Table 1. Technical Agent classification metrics (test set). S1 accuracy 0.642 (8,653 trades); S2 accuracy 0.665 (5,385).

Stage	Class	Prec.	Rec.	F1
S1 (trade)	no-trade	0.996	0.128	0.226
	trade	0.622	1.000	0.767
S2 (direction)	short	0.702	0.621	0.659
	long	0.633	0.712	0.670

5 Sentiment and Macro Agent

This section covers the narrative-sentiment specialist: information sources and scoring, and the FinBERT-plus-LLM classification pipeline.

5.1 Information Sources

The agent aggregates three channels with configurable weights: *news sentiment* ($w = 0.40$; NewsAPI, crypto-focused queries), *social sentiment* ($w = 0.30$; monitored Twitter/X accounts), and *macro indicators* ($w = 0.30$; CPI [Consumer

Price Index] releases, Fed rate decisions, tariff announcements via AlphaVantage).

The combined score is:

$$S = 0.40\, s_{\text{news}} + 0.30\, s_{\text{social}} + 0.30\, s_{\text{macro}}, \quad s_i \in [-1,\, 1]. \tag{1}$$

Recency is enforced via time decay: items older than 24 hours are down-weighted proportionally. The final signal is BUY if $S > 0.3$, SELL if $S < -0.3$, and HOLD otherwise.

5.2 FinBERT Classification and LLM-Based Reasoning

Each news item and tweet is tokenised and passed through FinBERT [1] (ProsusAI/finbert), yielding a probability distribution over {positive, negative, neutral}. The per-item sentiment score is $s = p_{\text{pos}} - p_{\text{neg}}$. When the pre-trained model is unavailable, a keyword-based fallback is used. FinBERT is used entirely off-the-shelf and deliberately so: a publicly available domain-adapted model with established provenance is preferable to one we would have to train and validate ourselves. Only the two CNN-LSTM models and the Risk Agent's LightGBM classifier were trained by us, and only because no off-the-shelf model exists for those tasks on this feature backbone.

Raw FinBERT scores can misrepresent sentiment when articles use indirect language or forward-guidance hedging. To address this, the agent invokes the Groq API (LLaMA-3-8b-instant) with a structured prompt supplying the aggregate FinBERT scores and the top-k article snippets, requesting a contextual reinterpretation and a natural language rationale. This two-stage pipeline — statistical classification followed by LLM reasoning — combines the efficiency of fine-tuned classifiers with the semantic depth of generative models [4].

5.3 Sanity Check on Financial PhraseBank

We verified our FinBERT integration on *financial_phrasebank* (sentences_75agree) [10], obtaining 94.73% accuracy (macro F1 = 0.9365), with mean sentiment scores of -0.915, $+0.064$ and $+0.859$ for negative, neutral and positive sentences. This is an *integration sanity check, not a benchmark contribution*: the Financial PhraseBank is the corpus on which FinBERT itself was fine-tuned [1], so this is close to training-set evaluation and is not evidence of generalisation. We report it only to establish that the model loads correctly, that our mapping $s = p_{\text{pos}} - p_{\text{neg}}$ separates the classes, and that polarity is not inverted — failures that would otherwise propagate silently into the fusion rule.

6 Risk and Volatility Agent

This section covers the systemic-risk specialist: its input signals, production model, and standalone evaluation.

6.1 On-Chain and Geopolitical Signals, and the LightGBM Model

The agent combines two streams with configurable weights: *on-chain metrics* ($w = 0.60$; transaction volume, hash rate, mempool size, active addresses, miner outflows) and *geopolitical events* ($w = 0.40$; news tagged as war, sanctions, economic crisis or trade conflict, scored by a keyword-weighted relevance model).

The composite risk score is:

$$R = 0.60\, R_{\text{on-chain}} + 0.40\, R_{\text{geo}}, \quad R \in [0,\, 1]. \tag{2}$$

Thresholds $R < 0.3 \Rightarrow$ `LOW`, $R > 0.7 \Rightarrow$ `HIGH`, and `MEDIUM` otherwise. The production risk model is a LightGBM classifier trained by us on 62 volatility-oriented features from `features_1h.parquet`, predicting a three-class label derived from 48-bar forward drawdown (DD, the peak-to-trough decline in cumulative return): `low_risk` ($< 1\%$), `medium_risk` (1–3%), or `high_risk` ($> 3\%$).

6.2 Results

Over the test period (9,115 bars) the severe drawdown rate is 25.7%. The agent discriminates risk regimes in the expected direction: `low_risk` bars (5,747; 63.0%) show a mean 48h DD of 1.78% with a 20.0% severe ($> 3\%$) rate; `medium_risk` (2,630; 28.9%) 2.40% and 30.5%; and `high_risk` (738; 8.1%) 4.02% and 52.4%. Binary classification (high-risk $\rightarrow$ severe drawdown) achieves macro F1 = 0.550 — the separation is real but modest.

7 Coordinator Agent

7.1 Uncertainty-Aware Signal Fusion and Risk Veto

The routing rule of Section 3.3 executes as one LangGraph node, after the three specialists' `AgentOutput` objects have accumulated in shared state. Confidence-weighted voting combines the three directional scores:

$$D = 0.45\, c_1 v_1 + 0.35\, c_2 v_2 + 0.20\, c_3 v_3, \tag{3}$$

where $v_i \in \{+1, 0, -1\}$ is each agent's directional vote and $c_i \in [0, 1]$ its own reported confidence — an uncertain agent is automatically down-weighted with no explicit routing decision needed. These coefficients are fixed, hand-calibrated constants (Section 3.3), not learned weights. A discrete *risk veto* adds a second, coarser layer of routing on top: it forces `HOLD` when Agent 3 signals `HIGH` risk and the directional score is borderline ($|D| < 0.55$) — precisely when the system is not confident enough to act despite elevated risk. The final signal is:

$$\hat{y} = \begin{cases} \texttt{BUY} & D \geq +0.20 \\ \texttt{SELL} & D \leq -0.20 \\ \texttt{HOLD} & \text{otherwise or risk veto.} \end{cases} \tag{4}$$

Finally, the Coordinator invokes the Groq API to synthesise the three agents' explanations into one human-readable rationale giving the final signal, the supporting evidence, and the primary risk caveat.

8 Experimental Evaluation

8.1 Accounting Convention

The convention materially affects how the figures below should be read, so we state it explicitly. Each bar is assigned a *triple-barrier* return: a position opened at that bar's close exits at an ATR-scaled take-profit or stop-loss barrier, or at a time barrier, whichever comes first. Reported cumulative P&L is the *arithmetic sum* of these per-bar returns, minus a 0.1% fee per side, over all active bars. Three consequences follow, none matching a standard equity curve. The trades *overlap* — a new position opens at almost every bar while earlier ones remain open — so the figures describe a portfolio of concurrent unit-sized positions, not a compounding account. Returns are *summed, not compounded*, so totals are not bounded by −100%. The same convention applied to Buy-and-Hold is why it reads −127.7%: impossible for an actual long-only holding, and an artefact of the convention rather than of market behaviour. Sharpe is computed on the same overlapping series and annualised by $\sqrt{8760}$; overlapping windows are strongly autocorrelated, so variance is understated and these values inflated relative to a non-overlapping Sharpe. The magnitudes are internally comparable across rows but not against externally reported returns.

8.2 Architecture Comparison

To justify the CNN-LSTM choice, we train TCN [3] (6 causal dilated residual blocks, hidden = 128) and TFT [9] (VSN [Variable Selection Network] + LSTM + MHA [Multi-Head Attention], $d = 128$) on the same 81 features and split, against LightGBM (GBDT [Gradient-Boosted Decision Trees]) as a non-temporal baseline. CNN-LSTM ranks highest and the non-temporal baseline is unprofitable, consistent with temporal context mattering here. All four share the same selected feature set, so the comparison is internally consistent but inherits the selection caveat of Section 10.

Table 2. Architecture comparison on the BTC/USDT test set (81 selected features, same split, convention of Section 8.1).

Model	Cum. P&L	Sharpe	Win Rate	Trades
CNN-LSTM	**+1,261%**	**12.97**	**63.6%**	**8,441**
TCN	−900%	−9.35	49.5%	7,350
TFT	−1,099%	−10.94	50.3%	9,110
LightGBM	−419%	−5.74	51.1%	7,176

8.3 Hyperparameter Selection

Two hyperparameters were selected post-training. A sequence-length search over $\{6, 12, 18, 24, 30, 48, 60, 90, 120\}$ bars selected **30**, and a threshold search selected

0.45 over the 0.55 default. *Both were scored on the test window*, which — with the feature selection of Section 4.2 — is why the figures here are in-sample (Section 10).

8.4 Incremental Agent-Combination Ablation

To examine the specialisation hypothesis (Section 1) we re-run the backtest incrementally, adding one agent at a time to the Technical baseline. Each combination re-normalises the Coordinator's weights over only the agents present (e.g. *TA+Sentiment* uses 0.45/0.80 and 0.35/0.80), thresholds unchanged; *Full MAS* additionally applies the risk veto, isolating discrete uncertainty-triggered routing from continuous weighting. Table 3 reports the results. As a consistency check, splitting the test period into six equal consecutive windows and evaluating the same model on each without retraining yields positive P&L in all six (+138% to +372%); since selection used this same window, this establishes only that performance is not concentrated in one sub-period, not out-of-sample validity.

Table 3. Incremental agent-combination ablation, test period (2025-02-05 to 2026-02-20; 9,133 bars; convention of Section 8.1).

Strategy / Combination	Cum. P&L	Sharpe	Win Rate	Max DD
Buy & Hold	−127.7%	—	—	—
TA-only	+1,799%	18.87	65.7%	−71.9%
TA+Sentiment	+1,612%	17.91	65.3%	−71.2%
TA+Risk	+1,726%	18.20	65.1%	−71.9%
TA+Sentiment+Risk	**+1,269%**	**15.40**	**65.0%**	**−68.8%**
Full MAS (+risk veto)	+1,259%	15.36	65.0%	−68.8%

Two observations follow. First, cumulative P&L and Sharpe decline monotonically as agents are added — the expected cost of gating marginal trades through additional, sometimes-disagreeing evidence. Second, maximum drawdown does *not* improve when either modality is added alone (Sentiment: −71.2%; Risk: −71.9%, unchanged from TA-only) — it improves only when the two act *together* (−68.8%). Since the single-agent rows bracket the joint row, the pattern is not explained by either signal individually, which is the form of evidence the hypothesis predicts: the reduction arises at the coordinator's joint fusion step. The effect is modest (3.1 percentage points) and measured on one asset over one window, so we treat it as suggestive rather than established. The discrete risk veto adds little beyond continuous weighting (423 of 9,133 bars, 4.6%, vetoed to `HOLD`).

9 LLM-as-a-Judge Evaluation

Aggregate backtest metrics do not capture whether each agent reasons correctly at the level of a single inference. We therefore introduce an *LLM-as-a-Judge*

protocol: live agent input–output pairs are packaged as JSON and submitted to Gemini 2.5 Flash with a structured prompt requesting a 1–10 quality score, reasoning, and three actionable recommendations, across four phases of increasingly strict prompting. Table 4 summarises the final evaluation. Technical, Sentiment and Coordinator agents are rated **9/10**; the Risk Agent receives **6/10** for two issues: (i) the geopolitical news pipeline admitted irrelevant articles (entertainment, local sports) tagged as political instability, diluting risk scores; and (ii) `volatility_metrics` was empty despite a non-zero `volatility_score`, indicating a data-pipeline gap.

Table 4. LLM-as-a-Judge evaluation (Gemini 2.5 Flash, final prompt revision, live agent inputs).

Agent	Score	Primary findings
Technical	9/10	Coherent signal/confidence; threshold logic correct.
Sentiment	9/10	Strong FinBERT integration; LLM reasoning aligned.
Risk	6/10	Noisy geopolitical events; empty volatility metrics.
Coordinator	9/10	Transparent fusion; each sub-agent contribution visible.

We regard the numeric scores as soft evidence — one model's judgement, not a validated instrument. The durable finding is different, and in our experience the most useful result in this paper: both Risk Agent issues are *silent* defects. Neither raises an exception, changes any output's shape, nor shows in aggregate P&L, since a diluted risk score still yields a well-formed signal the fusion rule consumes without complaint. Unit tests miss them because the code behaves as written; backtests miss them because the metrics stay plausible. Inspecting per-inference reasoning traces caught them. We recommend this protocol as a complement to backtesting for any production multi-agent system, independently of whether the scores are trusted.

10 Limitations

The trading figures in Section 8 are subject to three limitations that we state explicitly, since they bound what can legitimately be concluded from them.

Selection was not nested. Permutation-importance feature selection (Section 4.2) and both hyperparameter searches (Section 8.3) were scored on the test window rather than the validation window. All reported test figures are therefore *in-sample with respect to model selection*, and the walk-forward slices subdivide that same window rather than providing independent confirmation. A properly nested protocol — selecting on validation and touching test once — would be

required to establish out-of-sample performance, and we would expect the resulting figures to be materially lower. Rebuilding the results under such a protocol is the single most important item of future work.

The accounting convention is non-standard (Section 8.1): summed rather than compounded returns over overlapping positions, without capital constraints or slippage, with correspondingly inflated Sharpe values.

The sentiment benchmark is not independent: Financial PhraseBank is FinBERT's own fine-tuning corpus, so that accuracy is an integration check, not evidence of generalisation.

Two scope limitations apply to the architecture itself: the fusion weights are hand-calibrated rather than learned, so the system does not adapt if an agent's reliability drifts; and the evaluation covers one asset over one period, so the domain-generality we claim is an argument from construction — no agent-specific logic sits in the coordinator — not an empirical result.

11 Conclusion

We presented an adaptive, explainable multi-agent framework for heterogeneous financial decision fusion, instantiated on Bitcoin trading signal generation. Its durable contributions are architectural: a model-agnostic typed output contract that lets structurally dissimilar agents be combined by a single fusion rule; a three-tier degradation cascade converting data and model failures into reduced confidence rather than pipeline failure; and an uncertainty-driven routing rule in which each agent's confidence and the assessed risk state determine its weight, up to a full veto. None of these depends on the asset or model classes used here.

Our central hypothesis — that specialisation reduces cross-modality information interference and improves robustness — receives suggestive but not conclusive support: drawdown improves only when the sentiment and risk modalities are fused jointly, not when either is added alone, the pattern the hypothesis predicts. The effect is small, measured on one asset over one window, and the accompanying figures are in-sample (Section 10).

The result we consider most transferable is methodological: the LLM-as-a-Judge protocol surfaced two silent data-pipeline defects that neither unit tests nor aggregate backtest metrics revealed, because both produced well-formed outputs and plausible metrics. Reasoning traces are a distinct and complementary observability surface for multi-agent systems, and we expect that finding to hold well beyond financial applications.

Future Work. In priority order: (i) rebuilding the results under a nested selection protocol with a compounded, non-overlapping return convention; (ii) learning the fusion weights from realised outcomes; (iii) instantiating the framework on a second domain to test the domain-generality claimed above; and (iv) a controlled comparison against frameworks such as TradingAgents [16] and FinMem [17] under a shared protocol.

References

1. Araci, D.: FinBERT: Financial sentiment analysis with pre-trained language models (2019)
2. Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al.: Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion **58**, 82–115 (2020)
3. Bai, S., Kolter, J.Z., Koltun, V.: An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 (2018)
4. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners (2020)
5. Chase, H.: LangChain. `https://github.com/langchain-ai/langchain` (2022)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT 2019. pp. 4171–4186. Association for Computational Linguistics (2019)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997)
8. LangChain AI: LangGraph: Building stateful multi-actor applications with LLMs. `https://github.com/langchain-ai/langgraph` (2024)
9. Lim, B., Arık, S.Ö., Loeff, N., Pfister, T.: Temporal fusion transformers for interpretable multi-horizon time series forecasting. International Journal of Forecasting **37**(4), 1748–1764 (2021)
10. Malo, P., Sinha, A., Korhonen, P., Wallenius, J., Takala, P.: Good debt or bad debt: Detecting semantic orientations in economic texts. Journal of the Association for Information Science and Technology **65**, 782–796 (2014)
11. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. `https://bitcoin.org/bitcoin.pdf` (2008)
12. Picasso, A., Merello, S., Ma, Y., Oneto, L., Cambria, E.: Technical analysis and sentiment embeddings for market trend prediction. Expert Systems with Applications **135**, 60–70 (2019)
13. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30. pp. 5998–6008 (2017)
15. Wooldridge, M.: An Introduction to Multiagent Systems. John Wiley & Sons, 2nd edn. (2009)
16. Xiao, Y., Sun, E., Luo, D., Wang, W.: Tradingagents: Multi-agents llm financial trading framework. arXiv preprint arXiv:2412.20138 (2024)
17. Yu, Y., Li, H., Chen, Z., et al.: Finmem: A performance-enhanced llm trading agent with layered memory and character design. arXiv preprint arXiv:2311.13743 (2023)
18. Zhang, W., Zhao, L., Xia, H., et al.: A multimodal foundation agent for financial trading: Tool-augmented, diversified, and generalist. arXiv preprint arXiv:2402.18485 (2024)