The Gaining Paths to Investment Success: Information-Driven LLM Graph Reasoning for Venture Capital Prediction

Anonymous ACL submission

Abstract

Most venture capital (VC) investments fail, while a few deliver outsized returns. Accurately predicting startup success requires synthesizing complex relational evidence—company disclosures, investor track records, and investment network structures—through *explicit reasoning* to form coherent, interpretable investment theses. Traditional machine learning and graph neural networks both lack this reasoning capability. LLMs offer strong reasoning but face a modality mismatch with graphs. Recent graph-LLM methods target *in-graph* tasks where answers lie within the graph, whereas VC prediction is *off-graph*: the target exists outside the network. The core challenge becomes selecting graph paths that maximize predictor performance on an external objective while enabling step-by-step reasoning. We present MIRAGE-VC, addressing two obstacles: *path explosion* (thousands of candidate paths overwhelm LLM context) and *heterogeneous evidence fusion* (different startups need different analytical emphasis). Our information-gain-driven path retriever iteratively selects high-value neighbors, distilling investment networks into compact chains for explicit reasoning. A multi-agent architecture integrates three evidence streams via learnable gating based on company attributes. Under strict anti-leakage controls, MIRAGE-VC achieves +5.0% F1 and +16.6% Precision@5, and sheds light on other off-graph prediction tasks such as recommendation and risk assessment. Our code is available.[1]

1 Introduction

Venture capital (VC) investment is characterized by extreme asymmetry: from 1985 to 2009, roughly 60% of VC-backed firms lost money, while only 10% returned over five times the initial investment (Kerr et al., 2014). This high-risk, high-reward profile makes accurate startup success prediction crucial for portfolio optimization and capital allocation.

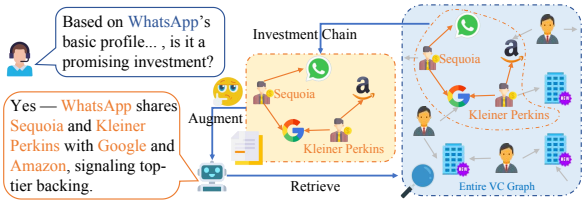

Figure 1: Impact of the selected chain on prediction.

The challenge lies in synthesizing complex relational evidence across multiple sources (Gompers et al., 2020). A typical investment decision requires integrating (i) the target company's peer-relative market positioning (Kim and Ritter, 1999), (ii) the track record and reputation of its lead investors (Hsu, 2004), and (iii) structural signals embedded in the investment network—such as co-investment patterns, investor coalitions, and proximity to successful exits (Hochberg et al., 2007). For instance, when evaluating a messaging startup like WhatsApp, the investment chain "WhatsApp ← Sequoia → Google ← Kleiner Perkins → Amazon" (Figure 1) indicates: top-tier screening by Sequoia increases follow-on financing probability, proximity to tech giants enhances scalability potential, and coalition strength among premier VCs facilitates strategic M&A opportunities. Integrating such heterogeneous evidence requires not only pattern recognition but also *explicit reasoning*—connecting the dots across comparables, investor profiles, and graph structures to form coherent, interpretable investment theses (Kaplan and Strömberg, 2004).

Existing predictive approaches fall short on these requirements. Traditional machine learning methods (Arroyo et al., 2019; Bento,

[1] https://anonymous.4open.science/r/MIRAGE-VC-323F

2017) rely on isolated firm features and ignore relational context. Graph neural networks (GNNs) (Lyu et al., 2025; Zhang et al., 2021) capture high-order investor–company dependencies but operate as a black box without exposing underlying reasoning. Consequently, they lack interpretability—providing no human-readable explanation for why a startup is promising—and cannot incorporate external knowledge like recent funding trends outside the training graph.

Large language models (LLMs) offer a natural remedy through their strong reasoning and broad world knowledge (Liu et al., 2023; Ko and Lee, 2024). However, LLMs face a fundamental *modality mismatch* when processing graph-structured data (Wang et al., 2023): their architectures are optimized for sequential text, not relational topologies. While recent work (Sun et al., 2023; Luo et al., 2023) integrates LLMs with knowledge graphs (KGs) for *in-graph question answering*—where the answer is a node within the graph (e.g., "Which company did Sequoia invest in after Google?")—VC prediction is an *off-graph* task: the prediction target (startup success) exists outside the network, and the graph serves solely as retrievable evidence. This distinction applies to other domains such as recommendation systems (predicting user–item affinity from interaction graphs) and credit risk assessment (predicting default from transaction networks). The core challenge thus becomes: ***how to select graph-derived paths that maximize a predictor's performance on an external objective***, rather than finding paths that terminate at an in-graph answer.

Even if we restrict our focus to path-based retrieval, the primary obstacle is **exponential path explosion**: shallow 1-hop neighborhoods around a target company often lack sufficient signal (Yu et al., 2021), yet extending to 3 or 4 hops generates thousands of candidate paths, many of which are redundant or weakly informative (Zhang et al., 2025). Indiscriminately feeding all paths to an LLM overwhelms its context window and dilutes attention. Additionally, VC prediction requires integrating semantically heterogeneous sources—company disclosures, investor profiles, and graph paths—whose relative importance varies by startup type, demanding adaptive fusion mechanisms to avoid over-reliance on secondary signals.

To address these challenges, we propose MIRAGE-VC, a multi-perspective retrieval-augmented generation (RAG) framework for VC prediction. To tackle path explosion, we introduce an ***information-gain-driven path retriever*** that iteratively expands from the target company, at each hop selecting the neighbor whose inclusion maximally improves an LLM predictor's accuracy. This selector is trained offline using task-specific information gain signals computed by a frozen LLM, then deployed at inference to extract a compact set of high-value investment chains—enabling explicit, chain-of-thought reasoning without overwhelming the LLM's context window. To integrate heterogeneous evidence, we employ a ***learnable gating network*** that dynamically weights verdicts from three information streams—company disclosures, lead investor profiles, and graph-based paths—each analyzed by a dedicated LLM agent. A manager agent then synthesizes these weighted signals into a calibrated prediction with an interpretable rationale.

Our contributions are as follows:

- By distilling complex VC networks into high-gain paths that synthesize evidence across multiple investors and deals, we allow LLMs to reason over relational patterns, exposing actionable investment signals valuable to VC practitioners.
- Our gating mechanism dynamically weights heterogeneous evidence streams based on startup characteristics, mirroring how different venture types require different analytical emphasis.
- Under strict anti-leakage controls, we achieve state-of-the-art performance on real-world VC data (+5.0% F1, +16.6% Precision@5). Our paradigm of selecting graph paths by marginal utility generalizes to other off-graph prediction tasks, such as recommendation and risk assessment.

2 Related Work

2.1 Graph-based VC Prediction

Traditional machine learning predictors rely on independent firm-level features and ignore relational context (Arroyo et al., 2019; Bento,

2017), whereas GNNs model investor–company graphs to capture high-order relational signals. SHGMNN (Zhang et al., 2021) combines predefined meta-paths, lightweight GNNs and Markov random field inference to integrate heterogeneous topologies and propagate labels for large-scale early-stage startup identification. GST (Lyu et al., 2025) applies unsupervised graph self-attention to update a dynamic startup-investor bipartite graph, improving node embeddings via link prediction and node classification to capture rich investor–company relations. These studies demonstrate that the structural properties of VC investment networks can significantly improve predictive accuracy. However, they remain limited by narrow knowledge scopes, weak reasoning capabilities, and a lack of interpretability.

2.2 Graph-augmented LLMs

Recent systems couple LLMs with knowledge graphs for *in-graph* QA—answers are entities in graph, reached or verified via triple retrieval and path reasoning (Sun et al., 2023; Luo et al., 2023). While effective at fact verification, they optimize in-graph objectives (entity/relation correctness) rather than selecting multi-hop evidence by its marginal utility to an external predictor. In our off-graph VC setting, the graph serves as evidence for startup-success prediction; what is needed is utility-aware selection of a few high-value investment chains as explicit evidence for prediction.

Proposed to remedy text-only RAG's inability to model structure and multi-hop dependencies, GNN-RAG frameworks retrieve relevant nodes via embedding similarity and inject local structural cues before handing context to an LLM (Mavromatis and Karypis, 2024). Yet they typically surface no explicit reasoning paths, limiting LLMs' strength in stepwise, interpretable chain-of-thought reasoning (Wei et al., 2022) and constraining multi-hop inference over heterogeneous investment networks.

3 Preliminary

3.1 Problem definition

This study aims to predict the success of early-stage start-ups, defined as companies that have completed their first formal financing round (seed or angel) but have not yet raised Series A funding (Zhang et al., 2021). While success is often measured by the attainment of Series A financing, prior studies use varying observation windows, which can introduce temporal bias. To mitigate this, we adopt a consistent one-year observation window following the seed round. This approach aligns with stage-based evaluation practices and helps control for external environmental factors (Boocock and Woods, 1997). The core task is to predict whether an early stage startup will secure subsequent financing within one year of its initial funding.

3.2 Data overview

We use the PitchBook[2] Global VC dataset, which spans investment activities from 2005 to November 2023. The dataset includes detailed investment records specifying the invested company, investor identity, funding amount, and financing stage. It also contains demographic information on both entrepreneurs and investors, including background, location, education, and professional biographies. Additionally, startup-level attributes are provided, such as team composition, industry classification, keyword tags, and geographic location. In total, the dataset encompasses 263,729 startups and 1,014,157 individuals. See Appendix A.6 for more details.

3.3 VC investment network

We model the VC ecosystem as a time-stamped heterogeneous information network $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{V}_{\text{cmp}} \cup \mathcal{V}_{\text{inv}}$ contains company and investor nodes. Each directed edge $e = (v_{\text{inv}}, v_{\text{cmp}}, t)$ represents an investment event from investor to company at time t, annotated with attributes such as the financing round and investment amount. For each company c^* that completes an angel or seed round at time t, we assign a binary label $y^* = 1$ if it secures Series A funding within the following 12 months, and $y^* = 0$ otherwise.

4 Methodology

4.1 Overview of our method

As shown in Figure 2, our proposed framework follows four sequential stages: (1) Graph retrieval: To supply the investment chain agent

[2]PitchBook is a financial data platform providing comprehensive information on private and public capital markets, including venture capital, private equity, and M&A transactions.

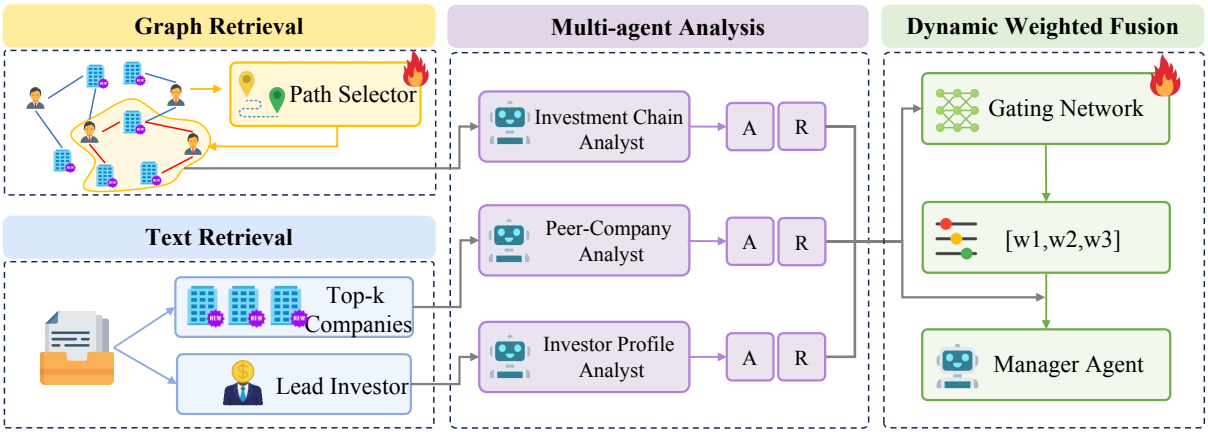

Figure 2: Overall Framework of MIRAGE-VC. It contains four key components: *Graph Retrieval* for path selecting, *Text Retrieval* for companies and investor, *Multi-agent Analysis* for multi-perspective information, *Dynamic Weighted Fusion* for adaptive information fusion.

with structured evidence, a learnable graph retriever extracts a high-value company–investor path from the investment graph. (2) Text retrieval: To provide the company and investor agents with textual context, semantic matching over public filings yields two textual views: (i) a similar company context and (ii) a lead-investor profile composed of demographics, career history, and labeled deal records. (3) Multi-agent analysis: The three prompts are processed by frozen LLM agents, each independently returning a binary decision and supporting rationale. (4) Perspective fusion: A lightweight gating network embeds and weighs the agent outputs. These are passed to a frozen manager agent, which produces a calibrated success probability and interpretable final decision.

4.2 Graph Retrieval

4.2.1 From classic IG to graph paths

As shown in Figure 3, path selection is framed as a sequential node selection problem. Starting from the target node c^*, at each hop, we choose the neighbor whose inclusion maximally improves the model's prediction accuracy. Although this heuristic does not guarantee a globally optimal path, it provides an efficient approximation—analogous to decision-tree splits via information gain—well suited to our multi-hop retrieval setting (Quinlan, 1986):

$$\mathrm{IG}(A) = H(Y) - H\big(Y \mid A\big) \tag{1}$$

We extend this principle to graphs by treating each candidate node v as an "attribute" A and estimating label uncertainty using the cross-entropy of a frozen LLM predictor. The rest of this section describes how these LLM-based IG signals are annotated offline, and how a lightweight selector model is trained to approximate them during inference.

4.2.2 LLM-generated gain labels

To obtain oracle supervision for path selector, we use a frozen LLM to quantify task-specific information gain per candidate expansion. For each target company c^*, we build a breadth-first expansion tree of depth at most three, retaining up to three previously unseen neighbors per node. At hop $h \in \{0, 1, 2\}$ let $S^{(h)} = \langle c^*, \ldots, u\rangle$ denote a current path and $\{v_1, v_2, v_3\} \subseteq N(u) \setminus S^{(h)}$ the corresponding candidate set.

Prompt construction To measure the incremental value of each candidate node v_i, we generate two prompts per expansion: (i) a *baseline* prompt P_{base} that verbalizes nodes in $S^{(h)}$, and (ii) a *candidate* prompt P_{v_i} that verbalizes the extended path $S^{(h)}_{v_i} = \langle c^*, \ldots, u, v_i\rangle$. A frozen LLAMA-3.1-8B classifier returns the success probabilities p_{base} and p_{v_i}. The procedure for converting causal-LM logits into binary probabilities p is detailed in Appendix.

Task-specific information gain Given the gold label $y \in \{0, 1\}$ ($1 = \textit{Success}$, $0 = \textit{Failure}$), we define the marginal gain from including v_i:

$$\Delta_{v_i} = \underbrace{\mathrm{CE}(y,\, p_{\text{base}}) - \mathrm{CE}(y,\, p_{v_i})}_{\text{cross-entropy reduction}} + \lambda_{\text{conf}}\big(|p_{v_i} - 0.5| - |p_{\text{base}} - 0.5|\big) \tag{2}$$

where the first term rewards the reduction in the prediction error (irrespective of $y = 0$ or 1); the second encourages confidence once the correctness is taken into account. CE denotes binary cross-entropy and $\lambda_{\text{conf}} \in [0, 1]$ balances correctness against confidence shift.

Training tuples Each training instance consists of: $\left(S^{(h)},\ S_{v_i}^{(h)},\ \Delta_{v_i}\right)$ The selector later receives the baseline path $S^{(h)}$, the extended path $S_{v_i}^{(h)}$, and the scalar gain Δ_{v_i} it should learn to predict. Because gains are computed for both successful and failed companies, the selector is explicitly trained to prefer extensions that push the LLMs towards the correct class with higher confidence.

4.2.3 Selector training objective

Each hop h of a target company contributes one *ranking group* $G^{(h)} = \{v_1, v_2, v_3\}$ with associated gains $\Delta_{v_1}, \Delta_{v_2}, \Delta_{v_3}$ annotated as in Eq. (2). For each candidate $v \in G^{(h)}$, we compute a difference feature:

$$x_v = \left[e_{\text{base}} \,\|\, e_v \,\|\, (e_v - e_{\text{base}})\right] \in \mathbb{R}^{2304} \quad (3)$$

where e_{base} and e_v are 768-dimensional sentence embeddings extracted once by a frozen encoder. A lightweight two-layer MLP $s_\theta : \mathbb{R}^{2304} \to \mathbb{R}$ assigns a score to each expansion.

Listwise objective To match the full gain pattern within each group we optimize a listwise objective. We first apply a within-group shift $r_i = \Delta_{v_i} - \min_j \Delta_{v_j}$, which preserves ordering while ensuring non-negativity ($r_i \geq 0$ and $\arg\max_i r_i = \arg\max_i \Delta_{v_i}$). We then form temperature-smoothed targets

$$q_i = \frac{\exp(r_i/\tau)}{\sum_{j=1}^{k} \exp(r_j/\tau)} \quad (4)$$

$$p_i = \frac{\exp(s_\theta(x_{v_i})/\tau)}{\sum_{j=1}^{k} \exp(s_\theta(x_{v_j})/\tau)} \quad (5)$$

The selector aligns its scores to the oracle distribution via

$$\mathcal{L}_{\text{list}}(G^{(h)}) = \text{KL}(q \,\|\, p) = \sum_{i=1}^{k} q_i \big(\log q_i - \log p_i\big) \quad (6)$$

where $\tau > 0$ controls target smoothness. Groups with $\sum_i r_i = 0$ carry no loss and are skipped. The final objective sums the listwise loss over all groups:

$$\mathcal{L}(\theta) = \sum_h \mathcal{L}_{\text{list}}(G^{(h)}) \quad (7)$$

This listwise training reproduces the within-group gain ranking and concentrates probability mass on high-gain candidates, enabling top-1 expansion at inference without re-invoking the LLM. See Appendix A.7.1 for more details.

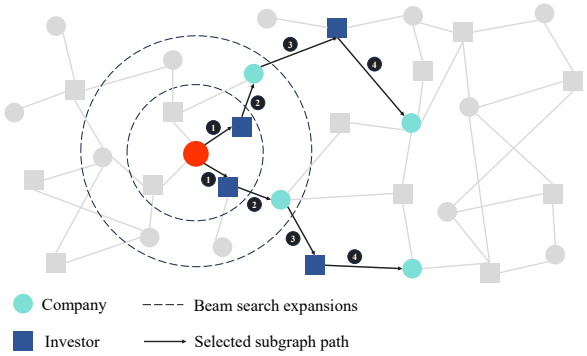

Figure 3: Illustration of how the path selector retrieves the best path from graph

4.3 Text Retrieval

4.3.1 Company Retrieval

To place the target company with historically comparable cases, we retrieve companies whose public descriptions are semantically similar to that of the target. The intuition is that companies sharing industry focus, product form, or market stage provide informative priors on likely financing outcomes. We use a frozen sentence encoder to embed each description and rank candidates by cosine similarity (Mikolov et al., 2013)—retaining the top-k peers $\mathcal{N}_k$. To avoid temporal leakage, we only consider firms founded before the target.

4.3.2 Investor Retrieval

Early capital often comes with intensive screening; hence the background of the lead investor provides strong priors about a startup future trajectory. We therefore identify, for the target company, the lead investor v^* as the one who committed the largest amount in its first disclosed financing round at time t_0.

From PitchBook entries for v^*, we extract two types of time-stamped records and discard any entry with timestamp $t \geq t_0$ to avoid leakage: empirical records $H^{\text{emp}} = \{(r_k^{\text{emp}}, t_k^{\text{emp}})\}$ and investment records $H^{\text{inv}} = \{(c_k^{\text{inv}}, t_k^{\text{inv}})\}$, where r_k^{emp} denotes a role or title held by v^*

and c_k^{inv} denotes a company previously backed by v^*. Both lists are ranked by recency and truncated to the top n items.

For every invested company c_k^{inv} we attach its brief profile and historical outcome label $\text{label}(c_k^{\text{inv}}) \in \{\textsc{Success}, \textsc{Failure}\}$. In addition, we collect static demographic attributes of v^* (education, age, gender), denoted A^*. The resulting structured summary $\{A^*, H^{\text{emp}}, H^{\text{inv}}\}$ is verbalised into a investor-analysis prompt.

4.4 Multi-agent Analysis

We instantiate three specialist LLM agents, each mimicking a typical VC due-diligence role, to elicit complementary evidence. The **Peer-Company Analyst (PC)** agent examines the similar-company prompt built from peer-company documents. The **Investor Profile Analyst (IP)** agent reads an investor-analysis prompt that summaries the lead investor's biography and historical portfolio. And the **Investment Chain Analyst (IC)** agent reasons over a graph-path prompt that presents information-gain–optimized chains from the investment network. Each prompt is processed by the same frozen GPT-3.5 Turbo, which outputs a binary verdict and accompanying free-form rationale. Because the LLM backbone is shared and frozen, any differences in output reflect differences in evidence alone.

4.5 Gating Network

We formalize rationale fusion as a supervised weighting problem: each agent's textual rationale is jointly embedded with the target company's structured profile, and a lightweight gating network learns instance-specific weights to aggregate evidence for binary classification. This approach preserves richer evidence than scalar scores, yields interpretable weights linked to their supporting sentences, and maintains a frozen backbone LLM.

For each target company, we dispose of three rationales R_i ($i \in \{\mathbf{PC}, \mathbf{IP}, \mathbf{IC}\}$) produced by the specialist agents. Each rationale is embedded using a frozen sentence encoder: $r_i = f_{\text{enc}}(R_i) \in \mathbb{R}^{d_r}$. The company's structured attributes (industry, stage, region) are represented by a fixed vector $a^* \in \mathbb{R}^{d_a}$. A two-layer MLP g_ϕ scores every view conditioned on the instance.

$$s_i = g_\phi\big([\, r_i \,\|\, a^*]\big) \tag{8}$$

$$w_i = \frac{\exp(s_i)}{\sum_j \exp(s_j)} \tag{9}$$

The weights w_i vary from case to case. The gated representation is the convex combination.

$$r_{\text{f}} = \sum_i w_i \, r_i \;\in\; \mathbb{R}^{d_r} \tag{10}$$

which is concatenated with the attributes, is passed to another two-layer MLP h_θ to obtain a task-aligned score for success prediction.

$$p = \sigma\big(h_\theta([\, r_{\text{f}} \| a^*])\big) \;\in\; (0,1) \tag{11}$$

With ground-truth label $y \in \{0,1\}$ the gating parameters $\{\phi, \theta\}$ are learned by binary cross-entropy.

$$\mathcal{L}(\phi,\theta) = -\,y \log p - (1-y)\log(1-p) \tag{12}$$

The softmax coefficients $\{w_i\}$ therefore offer an explicit, per-instance attribution of how much the company text, investor text, and graph path perspectives contribute to the final verdict. See Appendix A.8 for more details.

4.6 Manager Agent

To obtain a comprehensive and human-readable final decision, the following artifacts are collated into a meta-prompt and forwarded to an additional frozen GPT-3.5 Turbo instance (the decision/manager agent): the target company's profile, the four agent predictions $\hat{y}_i$, the four rationales R_i, and the learned importance weights w_i. Conditioned on this structured input, the decision agent produces: (i) a final binary decision $\hat{y}_{\text{final}} \in \{True, False\}$ and (ii) a natural language explanation grounded in the individual rationales and their respective weights. This architecture provides interpretable, multi-source decision-making aligned with human VC analysis practices.

5 Experiments & Results

5.1 Datasets

1. Dataset Sampling and Splitting We train both the Path Selector and Gate Network on subsets drawn from the original investment graph which contains a large pool of candidate

companies. To reduce computational overhead and avoid redundancy, we randomly sample 2,000 and 11,000 companies—respecting the overall success-to-failure ratio—for the Path Selector and Gate Network, respectively. Each subset is split into training, validation, and test sets in a 70 : 15 : 15 ratio, with class balance maintained across splits.

2. Final evaluation We select 2,510 startups that successfully secured their first round of financing between October 2021 to November 2023—entirely after the LLM's pretraining cutoff—to prevent any test instances from appearing in the pretraining corpus and avoid data leakage. This set includes 1,977 negative samples and 533 positive samples.

5.2 Baselines

We compare our model with state-of-the-art baselines across four categories: GNN-based methods (SHGMNN, GST), embedding-based methods (BERT Fusion), RAG-based LLM methods (RAG, GNN-RAG), and recent LLM-driven VC predictors (SSFF).

SHGMNN (Zhang et al., 2021) aggregates the heterogeneous network by meta-paths into a graph and applies a diffusion GNN with convex MAP inference in a variational EM loop to model label dependencies. **GST** (Lyu et al., 2025) models the evolving graph of startups and investors with unsupervised graph self attention, refines embeddings via link prediction and node classification losses, and feeds monthly graph snapshots into an LSTM to predict success over five years. **BERT Fusion** (Maarouf et al., 2025) concatenates BERT embeddings of each startup's Crunchbase[3] self-description with structured fundamentals and trains a lightweight neural classifier to predict success. **Standard RAG** (Lewis et al., 2020) uses a frozen dense retriever to embed queries and passages into a shared semantic space, retrieves the top k similar company profiles and investor summaries, and conditions a single LLM on them via RAG Token and RAG Sequence. **SSFF** (Wang et al., 2025) unites a divide-and-conquer multi-agent analyst block, an LLM-enhanced random-forest predictor with a founder-idea-fit network, and a RAG external-knowledge module to score startup prospects. **GNN-RAG** (Mavromatis and Karypis, 2024) couples a deep KGQA GNN that ranks candidate nodes and extracts shortest-path reasoning traces with an LLM that consumes those verbalized paths, yielding graph-aware RAG for KG question answering.

[3]Crunchbase is a public platform providing comprehensive data on companies, funding rounds, investors, and market trends.

5.3 Evaluation Metrics

In evaluating binary classification tasks, standard metrics such as precision, recall, and F1 score are commonly used. However, to better reflect the practical needs of investors selecting high-potential startups, we adopt the Precision at K ($P@K$) metric. $P@K$ measures the proportion of successful companies among the top K model recommendations, where candidates are ranked by the model's predicted confidence. This metric is well-established in VC prediction research (Sharchilev et al., 2018; Zhang et al., 2021; Lyu et al., 2021) for its ability to highlight top-performing investments. Further, to assess model performance over time, we compute the Average Precision at K ($AP@K$) across monthly cohorts. A higher $AP@K$ indicates that the model consistently prioritizes successful companies, thereby offering greater practical value to investors.

5.4 Parameter Settings

To prevent leakage of evaluation data into the LLM's pretraining, we use OpenAI's GPT-3.5 Turbo (knowledge cutoff: September 2021) to ensure deterministic outputs during experimentation. All training is conducted on a single NVIDIA RTX 4090 GPU with 24 GB of memory. For text embedding, we use Sentence-BERT to encode all textual inputs. Additional implementation details and parameter settings are provided in the Appendix.

5.5 Model Performance

All reported metrics are averaged over five independent runs of the LLM. Table 1 shows that our model improves AP@5 by 16.6%, AP@10 by 16.7%, and AP@20 by 8.0%—measures that directly capture retrieval quality when only a handful of top candidates can be pursued in practice. Notably, our model's AP@K gains increase as K decreases, demonstrating that higher confidence corresponds to greater accuracy in identifying promising startups—a

Table 1: Performance comparison with baselines. All values are percentages (the "%" sign is omitted). $AP@K$ indicates the monthly-averaged Precision@k.

Methods	**AP@5**	**AP@10**	**AP@20**	**Precision**	**Recall**	**F1**
SHGMNN	25.41	24.56	26.22	20.65	82.37	32.97
GST	26.71	25.71	27.14	21.75	**83.54**	34.51
BERT Fusion	24.67	26.67	25.33	23.63	24.95	24.27
Standard RAG	24.43	24.12	25.23	23.12	60.34	33.43
SSFF	28.23	30.02	28.42	23.23	69.41	34.81
GNN-RAG	29.42	27.53	27.04	22.81	71.10	34.54
Ours	**34.29**	**32.14**	**29.21**	**24.32**	73.44	**36.54**

property that is particularly valuable in real-world decision-making. It also achieves relative gains of 5.0% in F1 and 2.9% in Precision over the strongest baselines, indicating a more balanced and accurate classification of success outcomes. Compared to GNN-based methods (GST and SHGMNN), which boost recall through broad structural coverage but suffer precision drops from noisy neighbors; SSFF, a recent LLM-driven VC predictor that combines multi-agent analysis and RAG yet often surfaces redundant evidence; and RAG-based approaches (Standard RAG and GNN-RAG), which ground predictions in text but ignore explicit multi-hop relational chains—MIRAGE-VC's information-gain path retriever filters out low-value graph paths, and its multi-view gating adaptively weighs heterogeneous evidence, yielding balanced recall, higher precision, and a stronger ability to surface top-performing investment candidates. See experiments on different backbone models and train-free model in Appendix A.3 and A.4.

We then examine the results from a reasoning perspective. Analysis of 2,510 test samples shows that correctly classified startups had longer retrieved paths (4.44 vs. 3.31), suggesting richer network context aids the reasoning process (Appendix A.5). Appendix A.11 showcases the interpretability of reasoning outputs.

6 Ablation Study

Table 2 reports the ablation results. Removing the graph retrieval component significantly cuts Precision by 1.3% and F1 by 2.5%, highlighting the essential role of structural evidence. Both naively concatenating all 3-hop neighbors and picking paths at random degrade performance, demonstrating the superior noise-filtering capability of our path selector. Eliminating either similar company documents or the investor profiles also results in a notable performance drop, indicating their complementary value. In the fusion stage, aggregating all evidence within a single agent costs 1.4% F1, and replacing the learnable gating mechanism with fixed weights further costs 0.6% F1, highlighting the necessity of multi-agent fusion and adaptive gating.

Table 2: Results of ablation studies.

Removed Sub-module	**Precision**	**F1**
w/o Graph Retrieval	23.01	34.06
w/o Path Selector (all)	22.72	33.29
w/o Path Selector (random)	23.24	34.76
w/o Similar Company	23.45	35.54
w/o Investor Analysis	23.32	35.43
w/o Multi-agent Fusion	22.97	35.13
w/o Gating Network	24.05	35.94
FULL	24.32	36.54

7 Conclusion

We present MIRAGE-VC, a multi-perspective retrieval-augmented generation framework that addresses the lack of explicit reasoning and interpretability in existing VC prediction methods. Our information-gain-driven path retriever distills massive investment networks into compact chains, enabling LLMs to perform step-by-step reasoning over investor coalitions and network positions. A learnable gating network dynamically weights different aspects of startup characteristics. Experiments show substantial improvements. Unlike prevailing graph-LLM methods that target in-graph question answering, our approach addresses off-graph prediction where graphs serve as evidence for external objectives. This paradigm extends naturally to other off-graph tasks such as recommendation and risk assessment.

Limitations

While MIRAGE-VC pioneers the integration of graph-path retrieval with RAG in VC prediction, it exhibits the following limitations:

- **Single Proprietary Dataset.** We trained and evaluated MIRAGE-VC solely on PitchBook, a proprietary dataset that cannot be publicly released. This choice reflects common practice in VC research—prior work (e.g., Venture Capital Contracts (Ewens et al., 2022), SHGMNN (Zhang et al., 2021)) similarly relies on PitchBook due to its comprehensive coverage and data quality. The primary public alternative, Crunchbase, offers an open snapshot ending in 2013[4], which is too outdated for modern LLM-based prediction and would risk data leakage (many post-2013 companies likely appear in LLM pretraining corpora). Future work could enhance both generalizability and reproducibility by constructing a public benchmark: supplementing the 2013 Crunchbase snapshot with freshly crawled recent data, or collaborating with data providers to release an anonymized or time-lagged subset suitable for research.

- **Myopic Supervision Objective.** The current supervision scheme is local: Δ measures only the one-hop impact of adding a single node, and our beam search optimizes these immediate gains. This greedy strategy may overlook globally optimal subgraphs or beneficial interactions among multiple nodes. To address this, we plan to explore look-ahead scoring mechanisms and sequence-level optimization techniques—such as reinforcement learning over path sequences—to capture long-term dependencies and holistic graph structures.

Ethical Considerations

Data provenance and consent. We rely exclusively on publicly available and licensed (PitchBook) company and investor level records, using identifiers only for identity resolution. No human-subject data are collected, and we do not disclose raw documents and proprietary records.

Privacy, licensing, and compliance. All inputs come from public or licensed fields, with strict temporal ordering to prevent future-event leakage. Investor demographics (e.g., education, gender) are used solely in aggregated summaries. Users must honor the original data licenses and must not attempt re-identification.

[4] https://github.com/mtwilliams/crunchbase-in-2013

References

Javier Arroyo, Francesco Corea, Guillermo Jimenez-Diaz, and Juan A Recio-Garcia. 2019. Assessment of machine learning performance for decision support in venture capital investments. *Ieee Access*, 7:124233–124243.

Francisco Ramadas da Silva Ribeiro Bento. 2017. Predicting start-up success with machine learning. Master's thesis, Universidade NOVA de Lisboa (Portugal).

Grahame Boocock and Margaret Woods. 1997. The evaluation criteria used by venture capitalists: evidence from a uk venture fund. *International Small Business Journal*, 16(1):36–57.

Michael Ewens, Alexander Gorbenko, and Arthur Korteweg. 2022. Venture capital contracts. *Journal of Financial Economics*, 143(1):131–158.

Paul A Gompers, Will Gornall, Steven N Kaplan, and Ilya A Strebulaev. 2020. How do venture capitalists make decisions? *Journal of Financial Economics*, 135(1):169–190.

Yael V Hochberg, Alexander Ljungqvist, and Yang Lu. 2007. Whom you know matters: Venture capital networks and investment performance. *The journal of finance*, 62(1):251–301.

David H Hsu. 2004. What do entrepreneurs pay for venture capital affiliation? *The journal of finance*, 59(4):1805–1844.

Steven N Kaplan and Per ER Strömberg. 2004. Characteristics, contracts, and actions: Evidence from venture capitalist analyses. *The journal of finance*, 59(5):2177–2210.

William R Kerr, Ramana Nanda, and Matthew Rhodes-Kropf. 2014. Entrepreneurship as experimentation. *Journal of Economic Perspectives*, 28(3):25–48.

Moonchul Kim and Jay R Ritter. 1999. Valuing ipos. *Journal of financial economics*, 53(3):409–437.

Hyungjin Ko and Jaewook Lee. 2024. Can chatgpt improve investment decisions? from a portfolio management perspective. *Finance Research Letters*, 64:105433.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Xiao-Yang Liu, Guoxuan Wang, Hongyang Yang, and Daochen Zha. 2023. Fingpt: Democratizing internet-scale data for financial large language models. *arXiv preprint arXiv:2307.10485*.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

Shiwei Lyu, Xiaofeng Li, Suting Hong, Qing Ke, Jinjie Gu, Kunpeng Zhang, and Haipeng Zhang. 2025. Help me screen: Analyzing and predicting the success of start-ups in dynamic venture capital networks. *ACM Transactions on Intelligent Systems and Technology*.

Shiwei Lyu, Shuai Ling, Kaihao Guo, Haipeng Zhang, Kunpeng Zhang, Suting Hong, Qing Ke, and Jinjie Gu. 2021. Graph neural network based vc investment success prediction. *arXiv preprint arXiv:2105.11537*.

Abdurahman Maarouf, Stefan Feuerriegel, and Nicolas Pröllochs. 2025. A fused large language model for predicting startup success. *European Journal of Operational Research*, 322(1):198–214.

Dat Mai. 2024. Stockgpt: A genai model for stock prediction and trading. *arXiv preprint arXiv:2404.05101*.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Junlang Qian, Zixiao Zhu, Hanzhang Zhou, Zijian Feng, Zepeng Zhai, and Kezhi Mao. 2025. Beyond the next token: Towards prompt-robust zero-shot classification via efficient multi-token prediction. *arXiv preprint arXiv:2504.03159*.

J. Ross Quinlan. 1986. Induction of decision trees. *Machine learning*, 1(1):81–106.

Boris Sharchilev, Michael Roizner, Andrey Rumyantsev, Denis Ozornin, Pavel Serdyukov, and Maarten De Rijke. 2018. Web-based startup success prediction. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 2283–2291.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M Ni, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. *arXiv preprint arXiv:2307.07697*.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36:30840–30861.

Xisen Wang, Yigit Ihlamur, and Fuat Alican. 2025. Ssff: Investigating llm predictive capabilities for startup success through a multi-agent framework with enhanced explainability and performance. *Preprint*, arXiv:2405.19456.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Junchi Yu, Tingyang Xu, Yu Rong, Yatao Bian, Junzhou Huang, and Ran He. 2021. Recognizing predictive substructures with subgraph information bottleneck. *IEEE transactions on pattern analysis and machine intelligence*, 46(3):1650–1663.

Yangyang Yu, Zhiyuan Yao, Haohang Li, Zhiyang Deng, Yuechen Jiang, Yupeng Cao, Zhi Chen, Jordan Suchow, Zhenyu Cui, Rong Liu, and 1 others. 2024. Fincon: A synthesized llm multi-agent system with conceptual verbal reinforcement for enhanced financial decision making. *Advances in Neural Information Processing Systems*, 37:137010–137045.

Liangliang Zhang, Zhuorui Jiang, Hongliang Chi, Haoyang Chen, Mohammed Elkoumy, Fali Wang, Qiong Wu, Zhengyi Zhou, Shirui Pan, Suhang Wang, and 1 others. 2025. Diagnosing and addressing pitfalls in kg-rag datasets: Toward more reliable benchmarking. *arXiv preprint arXiv:2505.23495*.

Shengming Zhang, Hao Zhong, Zixuan Yuan, and Hui Xiong. 2021. Scalable heterogeneous graph neural networks for predicting high-potential early-stage startups. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2202–2211.

Wentao Zhang, Lingxuan Zhao, Haochong Xia, Shuo Sun, Jiaze Sun, Molei Qin, Xinyi Li, Yuqing Zhao, Yilei Zhao, Xinyu Cai, and 1 others. 2024. A multimodal foundation agent for financial trading: Tool-augmented, diversified, and generalist. In *Proceedings of the 30th acm sigkdd conference on knowledge discovery and data mining*, pages 4314–4325.

Han Zhou, Xingchen Wan, Lev Proleev, Diana Mincu, Jilin Chen, Katherine Heller, and Subhrajit Roy. 2023. Batch calibration: Rethinking calibration for in-context learning and prompt engineering. *arXiv preprint arXiv:2309.17249*.

A Appendix

A.1 More Related Work

With the rise of LLMs, a variety of LLM-based financial and VC decision-support systems have been proposed (Liu et al., 2023; Zhang et al., 2024; Ko and Lee, 2024). These systems typically rely on textual and numerical features or on simulation of real-world scenarios to improve prediction accuracy. For example, SSFF (Wang et al., 2025) and FinCon (Yu et al., 2024) systems establish a hierarchical manager–analyst collaboration mechanism, outperforming expert teams across multiple tasks; and StockGPT (Mai, 2024) is pretrained on extensive quantitative stock-market data to autonomously learn price-movement patterns, yielding substantial excess returns and demonstrating the promise of generative AI in complex financial decision making. However, none of these approaches directly integrate graph-structured knowledge—such as investor–startup relationship networks—and thus they are unable to fully capture path dependencies.

A.2 Impact of Parameters on Performance

We randomly sampled 1,000 companies from the non-test portion of our PitchBook data and, for each hyperparameter setting, ran each perspective's retrieval-and-analysis component five times to estimate mean F_1 scores and standard errors (Figure 4). In the text retrieval study (Figure 4 (a)), we varied the number of similar-company shots K and lead-investor résumé entries N from 0 to 6: performance rose sharply above the zero-shot baseline, peaked at $(K, N) = (4, 5)$, and then plateaued, motivating our choice of $K = 4$, $N = 5$. In the graph retrieval grid (Figure 4 (b)), we swept maximum search depths $d_{\max} \in \{1, \ldots, 6\}$ and path counts $P \in \{1, \ldots, 4\}$; the highest F_1 was observed at $(P, d_{\max}) = (2, 4)$, which we adopt in all main experiments.

A.3 Backbone Generalization and Few-Shot Comparison

A.3.1 Backbone LLM Experiments

Given the limitations of GPT-3.5-turbo, as shown in Table 3, we re-ran the inference pipeline with stronger backbone LLMs, and the

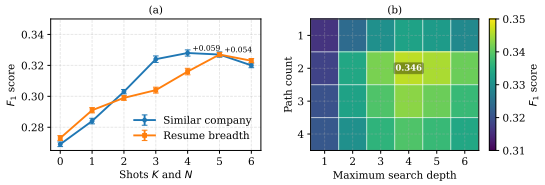

Figure 4: (a) Text retrieval: effect of the number of similar companies K and resume breadth N on the agents' F_1. (b) Graph retrieval: effect of search depth $d_{\max}$ and path count P on F_1.

results show consistent improvements: With GPT-4o-mini, MIRAGE achieves a +40.1% relative improvement in AP@5 and +11.4% in F1. With Qwen-3 4B, MIRAGE delivers a +34.6% improvement in AP@5 and +14.5% in F1. However, we note a potential data-leakage risk with newer LLMs, as their pretraining data may overlap with our test set. This is why we report results with GPT-3.5-turbo in the main paper to ensure a fair evaluation.

A.3.2 Train-Free Few-Shot Variant

As shown in Table 3, we also implemented a train-free few-shot prompting variant as a baseline. Its performance was over 40% lower than MIRAGE, indicating that while train-free pipelines reduce overhead, they are not yet viable when performance is a priority.

A.4 Comprehensive Performance Metrics and Sensitivity Analyses

A.4.1 Classification Metrics

As detailed in Table 4, we recalculated additional classification metrics: AUC-PR: Our method achieves a 46.9% improvement over the random baseline and a 3.8% improvement over the strongest baseline (SSFF). AUC-ROC: Our method achieves an 18.2% improvement over the random baseline and a 3.0% improvement over the strongest baseline (GNN-RAG). These results confirm that our approach consistently outperforms baselines across all key metrics.

A.4.2 Explanation of low recall

4.2As outlined in the paper and supported by prior VC research, real-world investment scenarios prioritize identifying the best few companies under limited resources, making AP@k the most relevant metric for evaluation. In this regard, our method consistently achieves significantly higher AP@k scores than all baselines, which reflects its effectiveness for practical ap-

Table 3: Backbone generalization and comparison with train-free few-shot baselines. All values are percentages (the "%" sign is omitted). $AP@K$ denotes the monthly-averaged Precision@K.

Backbone	**Setting**	**AP@5**	**AP@10**	**AP@20**	**Precision**	**Recall**	**F1**
GPT-3.5-Turbo	few-shot	22.84	22.07	23.14	21.53	57.44	31.24
	+ MIRAGE	34.29	32.14	29.21	24.32	73.44	36.54
GPT-4o-mini	few-shot	25.80	24.73	24.04	22.81	66.54	33.53
	+ MIRAGE	36.35	33.75	30.38	25.18	72.21	37.34
Qwen-3 4B (2025)	few-shot	21.94	21.14	21.01	20.04	63.02	30.41
	+ MIRAGE	29.53	28.43	28.14	23.33	68.55	34.83

Table 4: Discrimination metrics across baselines. All values are in $[0, 1]$.

Method	AUC-PR	AUC-ROC
Random	0.241	0.500
SHGMNN	0.324	0.541
GST	0.335	0.556
BERT Fusion	0.293	0.543
Standard RAG	0.312	0.562
SSFF	0.341	0.571
GNN-RAG	0.334	0.574
Ours	**0.354**	**0.591**

plication.

Regarding recall, it is important to note that the two methods with the highest recall are early GNN-based approaches. These methods achieve higher recall primarily due to:

- **Over-reliance on neighborhood aggregation:** These models aggregate broad neighborhood information, which increases recall but also introduces noise.
- **Use of broad positive thresholds:** This further inflates recall but comes at the cost of generating many false positives, leading to much lower precision compared to other methods.

In contrast, our proposed method is designed to optimize the precision–recall trade-off and achieves the best overall performance across composite metrics. This aligns with the practical requirements of scenarios like VC prediction, where precision in selecting high-potential targets is far more critical than simply achieving high recall.

A.4.3 Sensitivity Analysis

As detailed in Appendix A.2 (Figure 4), we evaluate sensitivity to the number of retrieved paths and the maximum reasoning depth. Performance is locally stable around the tuned hyperparameters: within path counts of 1–3 and depths of 3–5, the F_1 score deviates from the optimum by at most 0.003–0.010. The heatmap spans $\sim$ 0.31–0.35 and exhibits a clear ridge near 2–3 paths and depth 3–5, indicating low sensitivity in this region. At the extremes, shallow depth (≤ 2) or single-path configurations provide insufficient evidence and reduce F_1; conversely, overly deep settings (≥ 6) with many paths introduce noise and increase reasoning cost, slightly degrading performance. The results show that our method is robust to variations in hyperparameters within a broad optimal region, and achieves consistent improvements across metrics.

A.5 Error Analysis

We analyze the structural characteristics that differentiate correct from incorrect predictions. Figure 5 shows that correctly predicted companies have significantly deeper accessible paths (average maximum depth 4.44 hops) compared to misclassified cases (3.31 hops). The distribution exhibits a bimodal pattern: peaks at 2 hops and 5 hops, with fewer cases at intermediate depths (3–4 hops). Two-hop cases are abundant because every company connects to at least one investor (hop 1) who has co-invested with others (hop 2), forming the minimal network structure. Five-hop cases represent our retrieval cap—networks that could extend further but are truncated at the search limit. Intermediate depths are less common because expansion depends on continuous co-investment chains: many investors' ego-networks hit dead ends after the first layer due to sparse portfolios or isolated investment patterns, causing retrieval

to either stay shallow (2 hops) or jump directly to the cap (5 hops) when connections do exist. Errors concentrate in the shallow regime where limited structural evidence increases prediction ambiguity, while correct predictions benefit from the richer multi-hop context available in deeper networks.

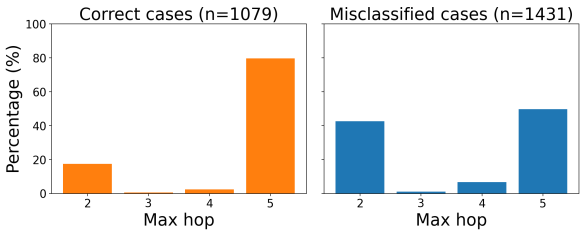

Figure 5: Distribution of path retriever maximum hop length by prediction outcome

A.6 More Details about Data

A.6.1 Data Sources and Time Filtering of data

Our entity-level documents are built by concatenating descriptive tabular fields (e.g., team members, educational background, sector, stage, region) and explicitly exclude outcome/status columns tied to the prediction targets (acquisition, IPO, financing); combined with time filtering that restricts text to information available before the prediction cutoff, this design minimizes label-leakage risk. As an empirical check, we keyword-searched a random sample of 200 company profiles and 300 investor profiles for `acquisition`, `IPO`, and `financing` and found no evidence of direct leakage. We will include implementation details of the document construction and time-filtering procedures in the revised paper.

A.6.2 Interaction Between Graph and Text Retrieval

As mentioned above, our raw data originate from relational tables. For each entity, we build a textual profile by concatenating salient columns, followed by normalization and deduplication. This yields a corpus of entity-level documents used across modules.

Text retrieval. Text retrieval operates directly over these constructed textual profiles, selecting semantically relevant entities from the corpus (e.g., via dense or keyword retrieval).

Graph retrieval. From the relational tables we induce an investment network and retrieve key paths based on relations (e.g., company–investor–portfolio links). The nodes along the selected paths are then *mapped back* to their textual profiles, which are consumed by downstream modules.

A.7 Implementation Details

A.7.1 Path Selector

Binary probability of LLMs We follow the mainstream likelihood-based scoring practice that normalizes the token-level log-likelihoods of verbalized labels (e.g., True/False) to obtain a Bernoulli probability, as adopted and analyzed in recent work on zero-shot classification, calibration, and probability-based prompt selection (Zhou et al., 2023; Qian et al., 2025). Given a prompt P, we verbalize labels as the strings "True" (Success) and "False" (Failure). For a string $w = (t_1, \ldots, t_m)$, we use the string log-likelihood

$$\log P(w \mid P) = \sum_{j=1}^{m} \log P(t_j \mid P, t_{<j}) \,.$$

Let

$$L_T = \log P(\text{“True”} \mid P), L_F = \log P(\text{“False”} \mid P)$$

The success probability is obtained by two-way normalization:

$$p = \frac{e^{L_T}}{e^{L_T} + e^{L_F}} = \sigma\big(L_T - L_F\big), \quad \sigma(x) = \tfrac{1}{1+e^{-x}}$$

We only query log-probabilities for the target strings.

Training Settings The trade-off weight in Eq. (2), $\lambda_{\text{conf}} \in [0, 1]$, is selected on the validation split by a small grid search; Table 5 shows that performance is stable in the range $0.1 - 0.3$, and we therefore fix $\lambda_{\text{conf}} = 0.2$ in all experiments. Hyperparameter is listed in Table 8.

Table 5: Validation *NDCG@1* (%) under different confidence–trade-off weights λ_{conf}.

λ_{conf}	0	0.10	0.20	0.30	0.40	0.50
NDCG@1 (%)	62.7	63.1	**63.4**	63.2	62.9	62.5

Evaluation and Results We evaluate the Path Selector with two ranking metrics. Hit@1 measures whether the selector's top-ranked candidate matches an oracle best-gain candidate in

the group. It directly reflects the success rate of our top-1 expansion policy at inference, thus aligning tightly with how the selector is actually used; a higher Hit@1 means we more often choose the maximal-gain extension. NDCG@1 normalizes the gain of the selected candidate by the maximum attainable gain in the group, yielding a score in $[0, 1]$. Unlike the binary Hit@1, NDCG@1 gives partial credit when the chosen candidate is near-optimal, making it more stable under noisy or close-valued oracle gains and better for hyperparameter tuning. A random baseline is obtained by drawing an i.i.d. score from $\mathcal{U}(0, 1)$ for every candidate and applying the same evaluation procedure. Compared with random scoring, the Path Selector improves Hit@1 by $+0.0879$ and NDCG@1 by $+0.1851$, confirming its ability to consistently priorities expansions of higher task-specific information gain.

Table 6: Selector performance on the held-out TEST split.

Method	NDCG@1 (%)	Hit@1 (%)
Random $\mathcal{U}(0, 1)$	44.92	33.33
PATH SELECTOR	**63.43**	**42.12**

A.8 Gating Network

Data Preparation Each training instance consists of the three agent rationales, encoded by a frozen all-MiniLM-L6-v2 sentence encoder into 384-dimensional vectors $\{\mathbf{h}_i\}_{i=1}^{3}$. Static company descriptors (industry, region, funding round) are one-hot-encoded into a 14-dimensional vector $\mathbf{c}$ and concatenated with the agent embeddings.

Training Settings Query–key projections $\mathbf{W}_Q, \mathbf{W}_K$ are applied to each $\mathbf{h}_i$ to obtain view-level attention scores. The attended view representations, together with $\mathbf{c}$, are fed to a two-layer MLP that outputs instance-specific weights $\mathbf{w} \in \mathbb{R}^3$ ($\sum_i w_i = 1$). The weighted sum of the three views is finally mapped by a second two-layer MLP to the task-aligned score for success prediction. All remaining hyperparameters are summarized in Table 8.

Training Objective and Inference Usage We use an MLP only to generate an auxiliary score and train the gate under the success objective, so that the learned weights align with how much each agent (path, similar-company, lead-investor) should contribute for a given company. At inference, the gating module outputs only these weights; the final decision is made by a downstream decision agent that takes the agents' analyses together with the learned weights.

Table 7: Performance of the gating network versus a random-weight baseline on the held-out test set.

Method	P (%)	F1 (%)
Random	19.98	19.49
Ours	**23.15**	**35.13**

Evaluation and Results Following the main task, the gating network is assessed on Precision (P) and F1. As a sanity check we replace the learned weights by a uniform random choice (Random); its performance marks the chance level of selecting the most informative view. Table 7 shows that the learned gate substantially outperforms this baseline, confirming that the network has indeed captured non-trivial view–attribute interactions.

Table 8: Hyper-parameters for the Gate Network (GN) and Path Selector (PS)

Parameter	GN	PS
Text vector dimension	384	384
Company key dimension	53	—
Batch size	256	256
Training epochs	50	30
Hidden width	256	256
Optimiser	AdamW	AdamW
Learning rate	5×10^{-4}	3×10^{-4}
Temperature τ	—	0.5

A.9 Text Embedding Model Analysis

Two pipeline components rely on a frozen sentence-encoder to obtain text representations: (i) the *graph retriever*, where the encoder embeds path descriptions, and (ii) the *gate network*, where it embeds each agent's generated answer. Here we analyse whether swapping the encoder markedly affects intermediate metrics. Table 9 shows that across both modules the performance gap between alternative encoders is marginal, confirming that our main results are not sensitive to the specific choice of text-embedding model.

Table 9: Impact of text encoders on the graph retriever and the gate network.

Text encoder	Dim.	Graph retriever		Gate network	
		$NDCG$ (@1,%)	Hit (@1,%)	P (%)	F_1 (%)
all-MiniLM-L6-v2	384	63.3	42.2	23.45	35.13
jina-embeddings-v2-base	768	63.4	42.1	23.12	35.31
e5-large-v2	1024	63.1	41.9	23.25	35.25

A.10 Resource Utilization and Latency

Our end-to-end pipeline comprises three phases: (i) retrieval via GPT-3.5 API, (ii) oracle scoring with Llama-3.1-8B and selector training, and (iii) gating network training and ablation studies. We issued approximately 40,000 GPT-3.5 requests (under 12,000 tokens each), processing $\approx$ 480 million tokens. Locally, we generated 16,857 gain labels with Llama-3.1-8B (5,619 group expansions, all within its 8 000-token window). Training the listwise selector on an NVIDIA RTX 4090 (24 GB VRAM) for 50 epochs required about 5 minutes, and the training of gating network completed in under 10 minutes. Including hyperparameter sweeps, total GPU time was 10 GPU h.

During inference, we noticed that LLM's usage is no higher than other well-known LLM-based forecasting systems, such as SSFF and GNN-RAG. This difference in cost is particularly insignificant when performing VC investment forecasting, a non-immediate and low-frequency forecasting task (weekly or monthly) that differs significantly from stock price forecasting. In our process, each prediction proceeds in two stages: the first stage issues three API calls, and the second stage issues a fourth call. On average, a single prediction consumes $18,831$ input tokens and takes 7.8 s wall-clock. This per-instance LLM usage is operationally acceptable. In a practical setting with 1000 new target companies per month, the theoretical inference budget is $188,310$ input tokens in total, costing about \$0.0192 with GPT-4o-mini, and requiring roughly 2 h end-to-end runtime, which is operationally acceptable.

A.11 Interpretability of Multi-Agent Reasoning Outputs

Using a canonical case (WhatsApp), we show how three evidence views are processed by the LLM and fused into a single binary decision (success means second round, acquisition, or IPO). The **Peer-Company**, **Investor Profile**, and **Investment Chain** analysts provide, respectively, time-consistent peer signals, lead-investor track record and alliances, and multi-hop path cues("WhatsApp ← Sequoia → Google ← Kleiner Perkins → Amazon"); a **Manager Agent** combines them to produce the verdict with a concise, evidence-grounded rationale.

A.11.1 Peer-Company Analyst

```
### Peer-Company Analyst ###
Prediction : Success
Analysis   : Among earlier peers in global
    mobile IM/OTT, the dominant exit path was
    acquisition after multiple rounds, not
    standalone IPO: ICQ (1996) → acquired by
    AOL in 1998; Skype (2003) → acquired by
    eBay in 2005, then by Microsoft in 2011;
    fring (2006) → acquired by GENBAND in
    2013. Standalone IPO cases generally
    paired with strong local monetization
    engines and high ARPU. By contrast,
    WhatsApps product and commercial
    stanceminimalist, anti-advertising,
    penetration-firstaligns more closely with
    ICQ/Skype/fring, i.e., large user scale
    with restrained monetization. We therefore
     expect that, after its initial financing,
     WhatsApp is significantly more likely to
    obtain follow-on funding or be acquired by
     a platform incumbent than to pursue an
    independent IPO; under the binary rule (
    success = second round, acquisition, or
    IPO), this view yields Success.
```

A.11.2 Investor Profile Analyst

```
### Investor Profile Analyst ###
Prediction : Success
Analysis   : WhatsApps first external round
    was sole-led by Sequoia (Partner Jim Goetz
    ). Pre-WhatsApp, Goetz had already
    established a successful pattern in mobile
    /platform assets: AdMob (mobile ads)
    acquired by Google for $750M in 2009; long
    -term board role at Palo Alto Networks,
    which IPOd in 2012; and Clearwell Systems,
     acquired by Symantec in 2011. This
    evidences a concentrated, long-haul style
    and the ability to route high-penetration,
     low-short-term-monetization entry assets
    toward large follow-on rounds or strategic
     M&A via superior resource orchestration.
    Combined with WhatsApps product profile
    and user growth, this view implies a high
    likelihood of at least a second round and
```

```
    a credible path to strategic acquisition;
    under the binary rule, Success.
```

A.11.3 Investment Chain Analyst

```
### Investment Chain Analyst ###
Prediction : Success
Analysis   : First, Sequoias sustained bets
    and track record in early consumer-
    internet projects imply a mature, partner-
    style methodology across product
    refinement, growth-model validation, key-
    hire placement, and follow-on round
    syndicationcreating strong signaling and
    improving access to subsequent financing.
    Second, graph linkages to established
    platforms such as Google/Amazon indicate
    structural adjacency in scaled
    capabilitiesdistribution, cloud, and
    advertising monetizationmaking it easier
    to obtain technical guidance, pilot
    opportunities, and ecosystem compliance
    playbooks. For a communications
    application that prizes reliability and
    low latency, such adjacency reduces
    engineering and compliance uncertainty
    during expansion and thus strengthens
    outside investors confidence in
    sustainable growth. Third, the Sequoia
    Kleiner Perkins collaboration typically
    confers more institutionalized capacity in
     follow-on introductions, acquisition-
    window identification, strategic-investor
    dialogues, and executive talent
    searchaccelerating the opening of the next
     financing or M&A window. Taken
    togethertop-tier screening, platform
    adjacency, and a coordinated investor
    alliancethese signals reasonably raise the
     prior that WhatsApp will secure a next
    round or achieve a major liquidity event.
```

A.11.4 Manager Agent

```
### Manager Agent ###
Prediction : Success
Analysis   : Synthesizing the three evidence
    sets, I materially raise the prior for a
    successful outcome for WhatsApp. Along the
     WhatsApp  Sequoia → Google  Kleiner
    Perkins → Amazon lineage, Sequoias
    partner-style playbook plus adjacency to
    Google/Amazon in distribution/cloud/
    compliance creates reusable organizational
     capacity for scale-up, key hiring, and
    follow-on syndicationreducing engineering
    and regulatory uncertainty and directly
    improving the odds of securing subsequent
    rounds. The earlier-cohort baseline shows
    that this sectors modal exit is
    acquisition after multiple financings;
    standalone IPOs skew toward heavy
    monetization narratives, which contrasts
    with WhatsApps minimalist, anti-ads,
    penetration-first posturemaking strategic
    M&A the more efficient medium-/long-term
    route. Finally, from the lead-investor
    vantage, Sequoia/Jim Goetz has repeatedly
    steered high-penetration entry assets to
    large follow-ons or M&A, providing the
    executional lever to realize that path.
    The logic chain is mutually reinforcing:
    platform adjacency lowers uncertainty (
    enabling follow-ons), the sectoral
    precedent favors M&A (enabling exit), and
    the lead-investor playbook supplies the
    mechanism (enabling execution). Hence the
    final binary decision: Success (1), with
    pathway ordering Follow-on financing
    Strategic acquisition  Standalone IPO.
```

A.12 Prompt Templates and Examples

This section provides the exact prompt templates used by each module and one illustrative example per template.

A.12.1 Company and Investor Basic Info Case

```
### Company Profile ###
Company name    : ACME Robotics
Founded year    : 2023
Headquarters    : San Francisco, USA
Industry        : Service Robotics
Employees       : 35 (as of 2025)
Key prototype   : Compact autonomous cleaning
    robot for boutique hotels
Revenue status  : Pre-revenue; paid pilots
    scheduled Q4-2025
Funding to date : USD 3.5 M (Seed round, Jun
    -2024)
Lead investors  : FutureFund (Jane Doe),
    SeedSpark Ventures
Company overview: ACME Robotics develops AI-
    driven service robots that automate
    routine cleaning tasks in hospitality and
    small retail environments. The platform
    combines low-cost modular hardware with on
    -device perception and a subscription
    software stack, aiming to deliver pay-as-
    you-go automation for venues that cannot
    afford traditional industrial solutions.
```

```
### Lead-Investor Profile ###
Investor name: Jane Doe  Partner @ FutureFund
Tenure       : 2016 - present

Previous positions
• Senior Engineer, ABB Robotics (2008 - 2012)
    global industrial-robotics leader.{
    COMPANY_PROFILE} (success)
• Investment Associate, TechEdge Capital (2012
     - 2016)early-stage deep-tech VC.\{COMPANY
    \_PROFILE\} (success)

Focus sectors    : Robotics • Edge AI • IoT
Assets under mgmt: USD 1.4 B

Investment record
• RoboVacacquired by Dyson (2021). {
    COMPANY_PROFILE} (success)
• MechArmIPO (2022). {COMPANY_PROFILE} (
    success)
• NanoGripacquired by Bosch (2020). {
    COMPANY_PROFILE} (success)
```

```
• ServoLinkceased operations (2019). {
    COMPANY_PROFILE} (failure)

Board seats : MechArm • FlexDroid • SensorX
Awards      : Forbes 30 Under 40 in VC (2023)
```

A.12.2 Path Analyst Prompt

```
Role: You are a senior venture-capital analyst
     who excels at step-by-step
      reasoning over investment paths to judge
     whether a seed / angel-stage
      start-up is likely to secure Series-A
    funding within the next year.

You are given three blocks of information:

(1) High-value investment path retrieved for {
    COMPANY_NAME}:{PATH_TEXT}
(2) Company profiles appearing in the path (
    each with outcome labels; True = raised
    Series A
    within 12 months after seed/angel, False =
     did not):{COMPANY_PROFILES} Success/
    Failure:{LABELS}
(3) Investor profiles appearing in the path:{
    INVESTOR_PROFILES}
(4) Target company profile:{
    TARGET_COMPANY_PROFILE}
Task:
  • Analyse the evidence and predict whether {
    COMPANY_NAME} will
    raise a Series-A round within 12 months.
  • Output **exactly** in the format:

      Prediction: True/False
      Analysis: <your step-by-step reasoning>

  • If evidence is insufficient, reason
    cautiously but still decide.
```

A.12.3 Company Analyst Prompt

```
Role:
  You are a senior venture-capital analyst who
     excels at using information from
  industry peers (similar companies) to judge
    whether a seed/angel-stage target
  will secure Series-A funding within the next
     year.

You are given:

(1) Target company profile:{
    TARGET_COMPANY_PROFILE}

(2) Comparable companies (each with outcome
    labels; True = raised Series A
    within 12 months after seed/angel, False =
     did not):{COMPANY_PROFILES} Success/
    Failure:{LABELS}

Task:
  • Analyse the evidence and predict whether {
    COMPANY_NAME} will
    raise a Series-A round within 12 months.
  • Output **exactly** in the format:

      Prediction: True/False
      Analysis: <your step-by-step reasoning>

  • If evidence is insufficient, reason
    cautiously but still decide.
```

A.12.4 Investor Analyst Prompt

```
Role:
  You are a senior venture-capital analyst who
     specialises in evaluating a
  start-ups lead seed/angel investor record to
     judge whether the target can
  secure Series-A funding within the next year.


You are given:
(1) Target company profile:{
    TARGET_COMPANY_PROFILE}

(2) Lead-investor rsum (prior operating roles
    and portfolio companies,
    each annotated as success or
    failuresuccess = the company raised Series
     A
    within 12 months of its seed/angel round;
    failure = it did not):{INVESTOR_PROFILE}

Task:
  • Analyse how the investors past successes
    and failures relate to the target
    companys sector, stage, and needs.
  • Predict whether the target will raise a
    Series-A round within 12 months.
  • Output **exactly** in the format:

      Prediction: True/False
      Analysis: <your step-by-step reasoning>

  • If evidence is insufficient, reason
    cautiously but still decide.
```

A.12.5 Manager Analyst Prompt

```
Role:
  You are a senior venture-capital analyst who
     excels at synthesizing other
  experts viewpoints to decide whether a seed/
    angel-stage start-up will secure
  Series-A funding within the next year.

You are given:

(1) Path-analyst verdict
    • Prediction: {PATH_PREDICTION}
    • Analysis  : {PATH_ANALYSIS}

(2) Similar-company analyst verdict
    • Prediction: {SIM_PREDICTION}
    • Analysis  : {SIM_ANALYSIS}

(3) Lead-investor analyst verdict
    • Prediction: {INV_PREDICTION}
    • Analysis  : {INV_ANALYSIS}

(4) Aggregate-weight advice
    The historical importance of the three
    perspectives is
```

```
    {WEIGHTS_VECTOR}

(5) Target company profile
    {TARGET_COMPANY_PROFILE}

Task:
  • Produce a single, final prediction on
    whether the target will raise a
    Series-A round within 12 months.
  • Output **exactly** in the format:

      Prediction: True/False
      Analysis: <your step-by-step reasoning>

  • If evidence is insufficient, reason
    cautiously but still decide.
```