Deep Research Partner Search

A 5-Phase Intelligent Search Architecture for Strategic Partner Discovery

---

End-to-End Workflow

Search is one component of the complete PartnerScope pipeline:

Stage	What Happens
1. Discovery Chat	AI-guided conversation extracts structured requirements: partner type, must-haves, success criteria, red flags
2. Search	5-phase deep research finds 50-60 candidates from databases and web *(this document)*
3. Evaluation	Dynamic framework scores candidates on strategic fit, adapting criteria to startup context
4. Refinement	Adjust weights, exclude candidates, re-rank instantly —no re-searching needed

Users can also **compare results** against external tools (Gemini Deep Research, OpenAI Deep Research) to validate quality.

Search Overview

PartnerScope employs a multi-phase deep research methodology to identify high-quality strategic partners for startups. Unlike simple keyword searches, our approach combines iterative query refinement, need decomposition, and an innovative batch-scoring system inspired by Recursive Language Models (arXiv:2512.24601).

Key Metrics: - Top-8 Average Score: **88.2/100** - Cost per search: **$0.80-1.50** - Candidates evaluated: **50-60 per search** - Final output: **20 ranked partners**

Data Sources

PartnerScope supports two complementary data sources that can be used independently or combined:

1. Database/CSV Source (Simulated Crunchbase)

Purpose: Leverage structured company databases for high-quality, pre-vetted candidates.

How it works: - Pre-curated CSV files exported from Crunchbase or similar databases - Keyword-based matching maps user queries to relevant CSV files - Provides structured data: company name, description, industry, location, CB Rank

Use case: When you have access to premium databases (Crunchbase, PitchBook, CB Insights) and want to search their curated company data.

Configuration:

```
# Keyword mappings connect queries to CSV files
mappings = {
    'pilot': {
        'keywords': ['housing', 'university', 'wellness',
        'student'],
        'csv_path': 'test_data/csv/pilot-partners.csv'
    },
    'validation': {
        'keywords': ['research', 'clinical', 'hospital',
        'academic'],
        'csv_path': 'test_data/csv/validation-partners.csv'
    }
}
```

Advantages: - Fast lookups (no API calls) - High data quality from curated sources - No per-query costs - Consistent, structured output

Limitations: - Requires manual CSV preparation/export - Data freshness depends on export frequency - Limited to companies in your database

---

2. AI Web Search Source

Purpose: Real-time discovery of partners across the entire web using LLM-powered search.

How it works: - OpenAI's web search tool performs live internet searches - LLM interprets results and extracts structured company data - 5-phase deep research methodology (detailed below)

Use case: When you need comprehensive discovery beyond your database, or don't have access to premium data sources.

Advantages: - Real-time, up-to-date information - Discovers companies not in databases - No data preparation required - Finds non-obvious partners through reflection

Limitations: - Per-query API costs ($0.80-1.50) - Slower than database lookups - May require enrichment for missing fields

---

Hybrid Mode (Recommended)

Best practice: Enable both sources for maximum coverage.

Source	Finds	Example
CSV/Database	Known players, established companies	Fortune 500 healthcare distributors
Web Search	Emerging players, niche specialists	New hospital networks, regional GPOs

Combined Output: - Database candidates provide reliable baseline - Web search adds discovery of non-obvious partners - Phase 4 (RLM Filtering) ranks ALL candidates together - Best partners surface regardless of source

---

The 5-Phase Architecture

Phase	Name	Process	Output
INPUT	User Need	*"Distribution partners with hospital networks"*	Search request
1	**Initial Discovery**	Generate 4 search queries from different angles. Execute web searches (6 companies per query). Capture obvious/direct matches.	~24 candidates
2	**Strategic Reflection**	Analyze gaps in Phase 1 results. Identify overlooked industries & partner types. Generate 3 "creative" queries for non-obvious fits.	+18 candidates
3	**Need Decomposition**	Decompose partner need into 3-4 specific sub-needs. Run targeted search for EACH sub-need. Ensure comprehensive coverage.	+16 candidates
4	**Batch Filtering**	Process candidates in batches of 8 (avoids context rot). Score each 1-10 on partnership fit.	Top 20 ranked

Phase	Name	Process	Output
		Aggregate scores programmatically.	
5	**Enrichment**	Identify candidates with missing data. Web search to fill gaps. Ensure clean, complete output.	20 final partners

Pipeline Flow:

> **User Input** → Phase 1 (Discovery) → Phase 2 (Reflection) → Phase 3 (Decomposition) → Phase 4 (Filtering) → Phase 5 (Enrichment) → **20 Ranked Partners**

Phase Details

Phase 1: Initial Discovery

Objective: Cast a wide net to capture obvious partnership candidates.

Process: 1. Generate 4 distinct search queries based on: - Startup profile (name, industry, stage) - Stated partnership needs - Keywords and context

1. Execute each query via OpenAI's web search tool

2. For each company found, extract:

    - Company name & website
    - Industry/sector
    - Location & size
    - Description
    - Needs satisfied (tags)
    - Partnership fit rationale

Query Generation Rules: - Keep queries short (5-10 words) - Use natural language (no boolean operators) - Each query takes a different angle

Example:

```
Input: "Distribution partners with hospital networks"

Generated Queries:
1. "hospital distribution partners healthcare startups"
2. "medical device distributors US market"
3. "healthcare supply chain companies partnerships"
4. "hospital network vendor programs"
```

Phase 2: Strategic Reflection

Objective: Identify gaps and find non-obvious partners.

Process: 1. Analyze companies found in Phase 1 2. Identify patterns and gaps: - Which industries are over-represented? - What partner types are missing? - What adjacent markets weren't explored?

1. Generate 3 "creative" queries targeting gaps

Reflection Framework: - What's the REAL underlying problem? - What types of partners did we NOT search for? - What unconventional partners could help?

Example Output:

```
Gap Analysis:
"Companies found are mostly large national distributors.
Missing: Regional hospital networks, GPO buying groups,
specialty clinic chains, healthcare IT integrators"

Creative Queries:
1. "group purchasing organizations healthcare startups"
2. "regional hospital systems vendor partnerships"
3. "healthcare IT companies distribution partnerships"
```

Phase 3: Need Decomposition & Targeted Search

Objective: Ensure comprehensive coverage by addressing each need individually.

Process: 1. Decompose the partner need into 3-4 specific, non-overlapping sub-needs

1. For each sub-need:
    - Generate targeted search query
    - Find companies that SPECIALIZE in that specific area
    - Tag results with the sub-need they address

Example:

```
Original Need: "Distribution partners with hospital networks"

Decomposed Sub-Needs:
1. Medical device distribution and logistics
2. Hospital group purchasing organization relationships
3. Healthcare supply chain management
4. Clinical trial site network access

Each sub-need gets its own targeted search.
```

Phase 4: Batch Filtering & Ranking

Objective: Intelligently filter and rank candidates without quality degradation.

The Problem: Evaluating 50+ candidates in a single LLM context causes “context rot” - the model’s attention degrades for items in the middle of long lists, leading to inconsistent scoring.

The Solution: Inspired by insights from Recursive Language Models (arXiv:2512.24601), we use **batch processing with external state management** to evaluate candidates in small groups, avoiding quality degradation from long contexts.

Note: This is not true RLM (which involves recursive self-calls). We apply the core insight—“don’t stuff everything in one context”—through iterative batch processing.

Process: 1. Store all candidates in external state (not in LLM context) 2. Process in batches of 8 candidates 3. For each batch: - Present candidates to LLM - Score each 1-10 on partnership fit - Store scores externally (outside LLM) 4. Aggregate all scores programmatically 5. Return top-K by score

Scoring Criteria: | Score | Meaning | |———-|———–| | 9-10 | DIRECTLY addresses multiple stated needs | | 7-8 | Addresses at least one stated need well | | 5-6 | Tangentially related but not clear fit | | 1-4 | Does NOT address the specific needs |

Why Batch Size 8? - Small enough to avoid context rot (~2K tokens) - Large enough for efficient processing - Empirically validated for consistent scoring

---

Phase 5: Selective Enrichment

Objective: Ensure all output data is complete and accurate.

Process: 1. Identify candidates missing: - Valid website URL - Adequate description (>30 chars)

1. For each incomplete candidate (up to 10):
    - Execute targeted web search
    - Extract missing fields
    - Update candidate record
2. Mark candidates as enriched

Technical Implementation

Model Configuration

Operation	Model	Cost/1M tokens
Query Generation	gpt-4.1	$2.00 / $8.00
Web Search	gpt-4.1 + web	$2.00 / $8.00 + $0.01/call

Operation	Model	Cost/1M tokens
Reflection	gpt-4.1	$2.00 / $8.00
Batch Scoring	gpt-4.1	$2.00 / $8.00
Enrichment	gpt-4.1 + web	$2.00 / $8.00 + $0.01/call

Cost Breakdown (Typical Search)

Phase	API Calls	Est. Cost
Phase 1	5 (1 query gen + 4 search)	$0.25
Phase 2	4 (1 reflection + 3 search)	$0.20
Phase 3	5 (1 decompose + 4 search)	$0.25
Phase 4	7 batches	$0.15
Phase 5	Up to 10 enrichments	$0.15
Total	**~25 calls**	**$0.80-1.50**

Output Format

Each candidate includes:

```json
{
  "name": "Company Name",
  "website": "https://company.com",
  "industry": "Healthcare Technology",
  "location": "San Francisco, CA",
  "size": "500-1000 employees",
  "description": "Brief description of the company...",
  "needs_satisfied": ["distribution", "hospital networks"],
  "how_it_helps": "Why this company is a good partnership fit",
  "discovery_phase": "phase1",
  "validation_score": 8,
  "validation_reasoning": "Strong hospital network
        relationships..."
}
```

Performance Benchmarks

Partners are evaluated using PartnerScope's dynamic evaluation framework, informed by the PartnerMAS methodology. The framework adapts criteria and weights based on startup context and partnership needs.

Metric	PartnerScope	Gemini Deep Research	OpenAI Deep Research
	88.2	82.6	79.4

Metric	PartnerScope	Gemini Deep Research	OpenAI Deep Research
Top-8 Average			
Overall Average	78.5	—	—
Score Range	62–96	—	—
Method	5-phase + batch filtering	Single-pass	Single-pass

Refinement: Iterate as You Learn

As results come in, specifications can be narrowed down and gaps filled through additional research.

What you can do:

Action	How It Works
Adjust weights	Change importance of criteria as patterns emerge—rankings update instantly
Exclude candidates	Remove companies that don't fit—remaining candidates re-rank automatically
Change criteria	Add or remove evaluation dimensions as requirements become clearer
Re-research	Run additional targeted searches to fill gaps or explore new directions
Compare externally	Validate results against Gemini or OpenAI Deep Research

Traditional research is linear: search once, deliver, start over if priorities change. PartnerScope is iterative: refine specifications, re-search specific areas, and build on what you've learned.

Key Innovations

1. Multi-Phase Query Refinement

Unlike single-query searches, we iterate through discovery → reflection → targeted phases to find both obvious and non-obvious partners.

2. Need Decomposition

Breaking complex needs into specific sub-needs ensures comprehensive coverage and prevents blind spots.

3. Batch Processing with External State

Inspired by RLM insights (arXiv:2512.24601), we process candidates in small batches with external state management. This preserves evaluation quality across large candidate sets, avoiding the “context rot” problem in long-context LLM evaluations. (Note: This is iterative batch processing, not true recursive LLM calls.)

4. Programmatic Score Aggregation

Scores are stored externally and aggregated programmatically, enabling consistent ranking across any number of candidates.

References

1. “Recursive Language Models.” arXiv:2512.24601, 2025. https://arxiv.org/abs/2512.24601
2. “PartnerMAS: An LLM Hierarchical Multi-Agent Framework for Business Partner Selection on High-Dimensional Features.” arXiv:2509.24046v1, 2025. https://arxiv.org/abs/2509.24046
3. OpenAI. “GPT-4.1 Technical Report.” 2025.

Document Version: 2.0 Last Updated: January 2026 PartnerScope — From months of research to minutes of conversation. | UTokyo Research, 2026