DocName:pipeline_walkthrough.md

What we learned from this pipeline walkthrough

1. Why naïve RAG breaks on regulatory docs

- **Structure > text**: Tax docs encode meaning via hierarchy, cross-references, and scope—not just sentences.
- **Lexical similarity is misleading**: “Box 2e” vs “Box 2f” look close but mean different things.
- **Negation flips truth**: Without polarity, embeddings create false positives.
  👉 **Conclusion**: You must model structure and semantics explicitly.

---

2. Knowledge graphs are the right abstraction

- Anchors (boxes/sections) act as **ownership boundaries**.
- Typed edges encode **how** concepts relate (includes, excludes, applies_if), not just that they co-occur.
- Polarity (positive vs negative) is essential to prevent incorrect associations.
  👉 **Conclusion**: A graph captures what embeddings alone cannot.

---

3. Physical layout understanding is non-negotiable

- Font size, bolding, columns, and page flow determine meaning.
- Column-aware ordering prevents misassignment across columns.
- Body-font inference anchors all downstream classification.
  👉 **Conclusion**: Semantics depend on accurate physical reconstruction.

---

4. Regex-first + heuristics beats LLM-first here

- Deterministic regex + position rules give **high precision** for roles and edges.
- Explicit uncertainty (`role=NULL`) is better than hallucinated labels.
- LLMs are better as **judges**, not primary extractors, at this stage.
  👉 **Conclusion**: Use LLMs to validate, not to guess structure.

---

5. Sentence-level gating is critical

- Extracting edges per sentence avoids cross-sentence bleed.
- Subset cues (“any part of”, “portion of”) must be explicit.
- Narration filters prevent descriptive text from becoming rules.
  👉 **Conclusion**: Precision requires aggressive gating.

---

6. Negation must dominate precedence

- `excludes` edges override all others.
- Negative knowledge creates **hard negatives** for training.
- Without this, models learn the wrong associations.
  👉 **Conclusion**: Negation is first-class knowledge.

---

7. Validation is as important as extraction

- Deterministic checks catch structural and statistical failures early.
- Edge distribution (A8) revealed a real weakness despite “working” output.
- Provenance and DAG integrity are mandatory for trust and traversal.
  👉 **Conclusion**: If it isn’t validated, it isn’t usable.

---

8. Training data quality comes from graphs, not chunks

- Graph edges naturally generate **balanced, labeled contrastive pairs**.
- `excludes` edges enable high-quality hard negatives.
- Stratified sampling prevents reference spam from dominating.
  👉 **Conclusion**: Better graphs → better training → better RAG.

---

9. Current bottleneck is semantic recall, not structure

- Structure and anchors are solid (22/22 boxes, clean DAG).
- Failure is due to **under-extraction of semantic edges**, not noise.
- Improving typed-edge patterns will likely flip the system to PASS.
  👉 **Conclusion**: The system is directionally correct; extraction depth is the next lever.

---

10. Big picture takeaway

This pipeline shows that **high-stakes documents require symbolic structure + semantic polarity**, not just vector similarity.
You're effectively building a **document compiler**—turning PDFs into executable knowledge.

If you want, next we can:

- Design new patterns to boost semantic edge recall
- Add lightweight ML to assist typed-edge detection
- Define retrieval strategies that exploit the graph directly (not just embeddings)

tax_embedding_technical_overview.md- Technical Overview: Graph-Based Contrastive Learning for Tax Document Retrieval

What We Learned

1. Tax documents are graphs, not text

IRS forms and instructions are inherently **relational**:

- Meaning lives in **edges** ("includes", "exception to", "see also"), not just nodes.
- Any system that treats them as independent chunks will *necessarily* miss meaning.

👉 **Insight:** Retrieval quality is capped unless document relationships are modeled explicitly.

---

2. Naive RAG fails because it destroys semantic dependencies

Chunking by box, section, or token count:

- Breaks parent–child relationships
- Loses cross-box logic (e.g., *Box 1a includes 1b and 2e*)
- Cannot answer comparative or concept-level questions

👉 **Insight:** The failure is structural, not just model quality.

---

3. Embedding models only learn what training pairs teach

Out-of-the-box embeddings:

- Don’t know that “Box 1a” and “Box 1b” are related
- Treat similar-looking but unrelated boxes as close
- Miss negation and exception semantics

👉 **Insight:** Retrieval improves only if we **explicitly teach relationships** during training.

---

4. Graphs are the right abstraction for both understanding and training

By modeling chunks as nodes and references as edges, we unlock:

- High-quality **positive pairs** (graph neighbors)
- **Hard negatives** (same doc, similar surface form, no edge)
- Hierarchical learning (section → subsection)
- Cross-document alignment (filer ↔ recipient ↔ form)

👉 **Insight:** The graph becomes a *supervisor* for contrastive learning.

---

5. Contrastive learning works best when pairs encode meaning, not proximity

Effective pairs come from:

- `includes`, `exception_to`, `same_field`, `defines`
- Cross-document equivalence
- Concept ↔ atomic grounding

Ineffective pairs come from:

- Page proximity
- Random same-section sampling

👉 **Insight:** Pair *quality* matters more than pair *volume*.

---

6. Conceptual chunks are necessary—but must be constrained

Atomic chunks are faithful but fragmented.
Conceptual chunks:

- Enable concept-level queries ("What are qualified dividends?")
- Improve recall and ranking dramatically

But only if:

- They are **source-grounded**
- Fully traceable (`derived_from`)
- Strictly validated for faithfulness

👉 **Insight:** Conceptual synthesis is powerful **only when auditable**.

---

7. Graph distance is a better negative signal than randomness

Hard negatives are most effective when they are:

- Same form
- Same structural level
- Numerically or lexically similar
- But **graph-distant**

Example:

- Box 1b vs Box 2a (both dividend amounts, different meaning)

👉 **Insight:** Graph topology gives us principled hard negatives for free.

---

8. Training must balance multiple relationship types

No single pair type is sufficient:

- Query–passage teaches retrieval
- Graph similarity teaches semantic structure
- Cross-doc pairs teach alignment
- Hierarchical pairs teach abstraction

- Concept↔atomic pairs teach grounding

👉 **Insight:** Embedding quality is a function of **relationship diversity**, not just data size.

---

9. Retrieval should be concept-aware, not chunk-only

Best-performing strategies:

- Retrieve **conceptual chunks first**, then expand to atomics
- Or retrieve both and re-rank jointly

Worst-performing:

- Atomic-only retrieval with large k

👉 **Insight:** Concepts should guide retrieval, not just embeddings.

---

10. Success is measurable—and looks different

Improvement isn't just higher similarity scores:

- Concept queries retrieve **complete semantic coverage**
- Comparative queries retrieve **both sides + relationship**
- Edge cases retrieve **rules + exceptions**

👉 **Insight:** Metrics like **concept coverage** matter as much as MRR.

The Core Takeaway (One Sentence)

> **To make tax documents retrievable, we must train embedding models on the same relational structure that humans use to understand them—and graphs are the missing supervision layer.**

If you want, next we can:

- Pressure-test this against another IRS form (e.g., 1099-INT or W-2)
- Translate this into a reusable "Graph-RAG" pattern

- Or define what *not* to model in the graph to keep it scalable

Here's a **clean synthesis of what we learned** from this Living Strategy — not a restatement, but the distilled insights that now guide decisions.

---

1. The Core Insight: Tax Retrieval Is a *Structure* Problem First, Not an Embedding Problem

- Tax documents are **not flat text**. They are:
  - Hierarchical (form → section → box → paragraph)
  - Cross-referential ("Box 1a includes 1b")
  - Precision-sensitive ("2e ≠ 2f")
- Any system that ignores structure will plateau quickly, regardless of model quality.

Learning:

> Retrieval quality is bottlenecked by **structural fidelity**, not model capacity.

This justifies:

- Knowledge graphs
- Registries
- Deterministic extraction
- Provenance-first design

---

2. Fine-Tuning Is a *Conditional Bet*, Not a Given

You explicitly framed fine-tuning as **falsifiable**, which is a major learning.

- Strong baselines today (contextual prefixes + BM25 + rerank) may already solve 70–80%
- Fine-tuning is only justified if it **materially beats** that baseline (+10–15% Recall@5)

Learning:

> We are not "building embeddings"; we are running an experiment with a clear kill switch.

This prevents sunk-cost bias and keeps the architecture honest.

---

3. The Knowledge Graph's Real Value Is *Training Signal Control*, Not Just Retrieval

A subtle but critical realization:

- The KG is not primarily for GraphRAG-style summarization
- Its highest leverage use is:
  - Generating **trusted contrastive pairs**
  - Filtering false negatives
  - Weighting training loss by edge confidence

Learning:

> The KG is a *data quality machine* more than a retrieval feature.

Retrieval benefits are incremental; training benefits are foundational.

---

4. Granularity Must Be Multi-Level — There Is No Single "Right Chunk Size"

You resolved a classic tension cleanly:

- Training wants **bigger, semantically rich units**
- Retrieval wants **small, precise units**

Learning:

> One graph, multiple granularities, different query patterns.

Hierarchy is the unifying abstraction:

- Anchors for training
- Paragraphs for retrieval
- Sentences only when density demands it

This avoids building parallel systems.

---

5. Confidence + Provenance Is the Only Scalable Way to Use LLMs Safely

The document makes this explicit and operational:

- LLMs are **proposers, not authorities**
- Every decision must:
    - Cite evidence
    - Emit confidence
    - Be rejectable

Learning:

> Debuggability beats raw capability.

This is why:

- Edge confidence tiers matter
- LLM-as-judge is gated
- “Unsupported claims” fail hard

This turns LLM usage from magic into engineering.

---

6. Most “Hard” Problems Are Actually *Routing* Problems

A key emerging realization in the Open Questions:

- Many queries don’t need retrieval at all:
    - “Explain Box 2a”
    - “What goes in Box 1b”
- These are deterministic alias → canonical ID lookups

Learning:

> If we don’t build a router, we’ll overuse retrieval and underperform.

This reframes part of the system from “search” to “structured navigation.”

---

7. Precision Failures Are Inevitable With Single-Vector Models — So We Hedge

You didn’t commit blindly to embeddings:

- You acknowledged known failure modes (2e vs 2f)
- You built in:
  - BM25
  - Cross-encoder reranking
  - ColBERT / SPLADE benchmarks

Learning:

> Architectural redundancy is intentional, not indecision.

This keeps options open and prevents lock-in.

---

8. Evaluation Is a First-Class System, Not an Afterthought

Another strong takeaway:

- Evaluation is:
  - Stratified
  - Precision-aware
  - Multi-hop aware
- Near-miss negatives prevent false confidence

Learning:

> If the eval can’t fail the system, the system isn’t real.

This ensures progress is measurable and meaningful.

---

9. Phasing Is About *Risk Burn-Down*, Not Features

Phase A isn't about shipping features — it's about validating assumptions:

- Can we extract reliably?
- Can we build a clean graph?
- What is the real baseline ceiling?

Learning:

> Each phase answers a "should we continue?" question.

That's a mature systems mindset.

---

10. The Meta-Learning: This Is an Anti-Overengineering Strategy

Despite the complexity, the document is deeply **anti-overbuild**:

- Multiple explicit pivot points
- Cheap baselines compared
- Alternatives (ColBERT, routing, cheap extraction) kept alive

Learning:

> Complexity is earned only if numbers demand it.

One-Line Summary

> **We learned that winning tax document intelligence is about controlling structure, trust, and evaluation—not about betting on smarter models—and that every expensive component must justify itself against a strong, simpler baseline.**

If you want, next we can:

- Turn this into a **decision tree** ("If metric X < Y, do Z")
- Extract a **1-page exec version**

- Or map this to a **reference architecture diagram**

Tax_rag_living_strategy.md:

Here's a clean synthesis of what we learned from this Living Strategy — not a restatement, but the distilled insights that now guide decisions.

---

1. The Core Insight: Tax Retrieval Is a *Structure* Problem First, Not an Embedding Problem

- **Tax documents are not flat text. They are:**
  - **Hierarchical (form → section → box → paragraph)**
  - **Cross-referential ("Box 1a includes 1b")**
  - **Precision-sensitive ("2e ≠ 2f")**
- **Any system that ignores structure will plateau quickly, regardless of model quality.**

Learning:

> **Retrieval quality is bottlenecked by structural fidelity, not model capacity.**

This justifies:

- **Knowledge graphs**
- **Registries**
- **Deterministic extraction**
- **Provenance-first design**

---

2. Fine-Tuning Is a *Conditional Bet*, Not a Given

You explicitly framed fine-tuning as falsifiable, which is a major learning.

- **Strong baselines today (contextual prefixes + BM25 + rerank) may already solve 70–80%**
- **Fine-tuning is only justified if it materially beats that baseline (+10–15% Recall@5)**

Learning:

> **We are not “building embeddings”; we are running an experiment with a clear kill switch.**

This prevents sunk-cost bias and keeps the architecture honest.

---

3. The Knowledge Graph’s Real Value Is *Training Signal Control*, Not Just Retrieval

A subtle but critical realization:

- **The KG is not primarily for GraphRAG-style summarization**
- **Its highest leverage use is:**
  - **Generating trusted contrastive pairs**
  - **Filtering false negatives**
  - **Weighting training loss by edge confidence**

Learning:

> **The KG is a *data quality machine* more than a retrieval feature.**

Retrieval benefits are incremental; training benefits are foundational.

---

4. Granularity Must Be Multi-Level — There Is No Single “Right Chunk Size”

You resolved a classic tension cleanly:

- **Training wants bigger, semantically rich units**
- **Retrieval wants small, precise units**

Learning:

> **One graph, multiple granularities, different query patterns.**

Hierarchy is the unifying abstraction:

- **Anchors for training**

- **Paragraphs for retrieval**
- **Sentences only when density demands it**

This avoids building parallel systems.

---

5. Confidence + Provenance Is the Only Scalable Way to Use LLMs Safely

The document makes this explicit and operational:

- **LLMs are proposers, not authorities**
- **Every decision must:**
    - **Cite evidence**
    - **Emit confidence**
    - **Be rejectable**

Learning:

> **Debuggability beats raw capability.**

This is why:

- **Edge confidence tiers matter**
- **LLM-as-judge is gated**
- **“Unsupported claims” fail hard**

This turns LLM usage from magic into engineering.

---

6. Most “Hard” Problems Are Actually *Routing* Problems

A key emerging realization in the Open Questions:

- **Many queries don’t need retrieval at all:**
    - **“Explain Box 2a”**
    - **“What goes in Box 1b”**
- **These are deterministic alias → canonical ID lookups**

Learning:

> **If we don't build a router, we'll overuse retrieval and underperform.**

This reframes part of the system from "search" to "structured navigation."

---

7. Precision Failures Are Inevitable With Single-Vector Models — So We Hedge

You didn't commit blindly to embeddings:

- **You acknowledged known failure modes (2e vs 2f)**
- **You built in:**
  - **BM25**
  - **Cross-encoder reranking**
  - **ColBERT / SPLADE benchmarks**

Learning:

> **Architectural redundancy is intentional, not indecision.**

This keeps options open and prevents lock-in.

---

8. Evaluation Is a First-Class System, Not an Afterthought

Another strong takeaway:

- **Evaluation is:**
  - **Stratified**
  - **Precision-aware**
  - **Multi-hop aware**
- **Near-miss negatives prevent false confidence**

Learning:

> **If the eval can't fail the system, the system isn't real.**

This ensures progress is measurable and meaningful.

---

9. Phasing Is About *Risk Burn-Down*, Not Features

Phase A isn't about shipping features — it's about validating assumptions:

- **Can we extract reliably?**
- **Can we build a clean graph?**
- **What is the real baseline ceiling?**

Learning:

> **Each phase answers a "should we continue?" question.**

That's a mature systems mindset.

---

10. The Meta-Learning: This Is an Anti-Overengineering Strategy

Despite the complexity, the document is deeply anti-overbuild:

- **Multiple explicit pivot points**
- **Cheap baselines compared**
- **Alternatives (ColBERT, routing, cheap extraction) kept alive**

Learning:

> **Complexity is earned only if numbers demand it.**

One-Line Summary

> **We learned that winning tax document intelligence is about controlling structure, trust, and evaluation—not about betting on smarter models—and that every expensive component must justify itself against a strong, simpler baseline**