Here's a structured mind map summarizing the key insights from the paper *"Towards Scalability and Extensibility of Query Reformulation Modeling in E-commerce Search"* by Ziqi Zhang et al.

🧠 Mind Map: Enhancing Query Reformulation in E-commerce Search

1. Motivation

- **Challenge with Tail Queries**:Less common or "tail" queries often lack sufficient behavioral data (e.g., clicks, purchases), making it difficult for search systems to deliver relevant results
- **Query Reformulation (QR)**:Proposes that tail queries can leverage the behavioral patterns of semantically similar, more popular queries to improve search relevance
- **Scalability & Extensibility Issues**:Implementing QR in regions with low traffic and complex multilingual environments poses challenges in terms of scalability and adaptability

2. Research Objectives

-Develop a QR solution that is both scalable and extensible, capable of functioning effectively even with limited and noisy training data -Ensure the solution can handle complex linguistic characteristics inherent in multilingual settings

3. Methodology

- **Data Mining Refinement**:Enhanced processes to extract meaningful patterns from sparse and noisy behavioral data
- **Model Training Objectives Redefinition**:Adjusted training goals to better capture semantic similarities between queries
- **Training Strategy Reshaping**:Implemented new strategies to train models effectively with limited high-quality data

4. Implementation

- **Integration into Amazon's Infrastructure**:The proposed QR solution was implemented within Amazon's product search system
- **Multilingual Focus**:Special attention was given to Japanese, Hindi, and English queries to address linguistic diversity

5. Evaluation & Results

- **Online A/B Testing**:Conducted to assess the effectiveness of the QR solution
- **Revenue Impact**: -Japanese Search Ranking: 0.14% increase in overall revenue -Hindi Search Ranking: 0.29% increase in overall revenue -English Search Ranking: 0.08% incremental gain compared to the

legacy system -Japanese Ads Matching: 0.36% increase in Ads revenue

6. Conclusions & Future Work

- **Effectiveness**:The QR solution demonstrated measurable improvements in search relevance and revenue across multiple languages
- **Scalability & Extensibility Achieved**:The approach proved effective even with limited training data and in complex linguistic settings
- **Future Directions**:Further research may explore broader applications of the QR solution and its integration into other components of e-commerce platforms

---

This mind map encapsulates the paper's comprehensive approach to enhancing query reformulation in e-commerce search, addressing challenges related to data sparsity and linguistic diversity

Sure! Let's break down **Figure 3** from the paper *"Towards Robustness Analysis of E-Commerce Ranking System"* in detail. This figure illustrates the **training and mining pipeline** for improving ranking model robustness using the **ANCE (Approximate Nearest Neighbor Negative Contrastive Estimation)** strategy.

🧩 Figure 3: ANCE Mining and Continuous Fine-tuning Process

➢ 1. ANCE Self-Retrieval Process (Bi-Encoder Phase)

- **Goal**: Generate useful training pairs for the bi-encoder by mining hard negatives.
- **Steps**:
    1. A **bi-encoder** model is initialized and used to **retrieve top-K candidates** from the product catalog for each query.
    2. These retrieved (query, product) pairs are treated as **pseudo-positive training pairs**.
    3. To remove potential **false positives** (i.e., irrelevant results that happen to be top-ranked), any pair that **already exists as a co-purchase** is **subtracted out** from the dataset.
        - This avoids training on potentially biased or overly easy samples.
    4. This forms the **ANCE dataset**, which includes hard negatives and high-quality pseudo-positives.

➢ 2. Bi-Encoder Fine-tuning

- The bi-encoder model is **iteratively fine-tuned** on the ANCE data generated above.
- After each iteration of fine-tuning, the self-retrieval step is **repeated**, using the improved model to generate new and better pseudo-labeled pairs.
- This loop continues, gradually improving the model's understanding of relevance.

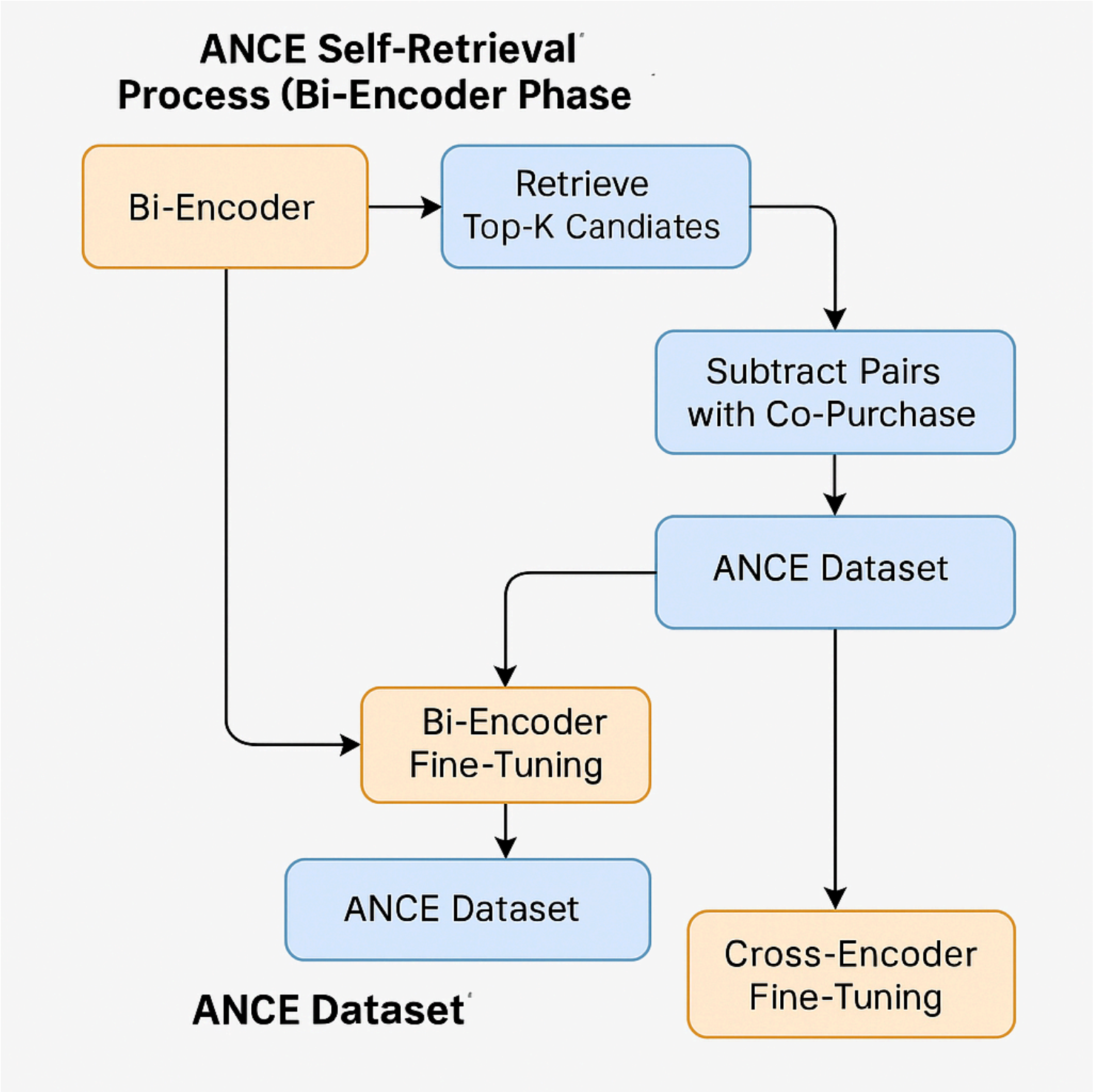
➤ **3. Cross-Encoder Fine-tuning**

- Once the **bi-encoder** reaches sufficient quality, the **cross-encoder** is introduced.
- The **cross-encoder** is trained using the **same ANCE dataset** that was generated from the **bi-encoder** phase.
- The cross-encoder can **better model interactions** between query and product text (since it considers both together during encoding), leading to **higher final ranking quality**.

💡 Summary of Key Advantages

- **ANCE-based mining** provides **diverse and challenging training data** without requiring manual labeling.
- The **bi-encoder** is used for efficient retrieval, while the **cross-encoder** offers higher accuracy for reranking.
- The **pipeline is iterative**, allowing the model to continuously improve by learning from its own top results.
- This approach supports **robust ranking** by emphasizing **semantic consistency** and avoiding reliance on fragile, co-purchase-based signals.

Let me know if you'd like a visual version of this pipeline explained with arrows and boxes!



Great! Let's contextualize the **ANCE mining and continuous fine-tuning process** using examples from the **Samsung e-commerce search ecosystem** and understand how it improves both **bi-encoder** and **cross-encoder** performance.

🛒 Use Case: Samsung E-commerce Search

Let's say users are searching for products like:

- "**S23 Ultra smartphone**"

- “**Galaxy S23 Ultra phone**”
- “**Samsung flagship phone 2023**”

These are **semantically similar queries** but may return **different ranked results** unless the ranking model understands the underlying **intent and product similarity** well.

🔁 Applying the ANCE Process in Samsung Ecospace

🔸 Step 1: Bi-Encoder Retrieval (Self-Supervised Mining)

Example Flow:

1. **Query:** “Samsung flagship phone 2023”
2. The **bi-encoder** retrieves top-K products from the catalog:
    - Product A: Galaxy S23 Ultra 256GB
    - Product B: Galaxy S22 Ultra
    - Product C: Galaxy S23+
3. These form the **self-retrieved pseudo-labeled data**.

👉 But we want to make sure we’re **not just memorizing co-purchased products** (e.g., S23 Ultra + Samsung Buds).

🔸 Step 2: Filter Out Co-Purchases

- Remove pairs like:
  (Query: "Samsung flagship phone 2023", Product: Samsung Galaxy Buds 2)
  Because it’s likely a **co-purchase**, not semantically a perfect product match.

📦 Result: A curated **ANCE dataset** of (query, product) pairs that reflect **semantic relevance** rather than behavioral bias.

🔸 Step 3: Fine-Tuning Bi-Encoder

Train the bi-encoder with this curated ANCE data.

Impact:

- Learns to embed queries like:
    - “Galaxy S23 Ultra phone”

    - “Samsung top phone 2023”
- Closer to products like “Galaxy S23 Ultra” in embedding space.
- Generalizes better to **tail queries** (e.g., "S-series best phone 2023")

🔸 Step 4: Fine-Tuning Cross-Encoder

Now pass the same ANCE data to a **cross-encoder**, which looks at query and product jointly.

Example input:

- Query: “Galaxy S23 Ultra phone”
- Product Title: “Samsung Galaxy S23 Ultra 256GB Phantom Black”

🔍 The cross-encoder uses full token interaction to determine:

- Word overlaps
- Contextual relationships
- Better scoring → higher accuracy for final reranking

✅ Benefits for Samsung E-commerce

Component	Value
Bi-Encoder	Faster retrieval, great for large-scale product catalog searches
Cross-Encoder	Accurate reranking, improved semantic matching
ANCE Mining	Provides better training data without manual labels
Continuous Loop	Models get stronger over time by learning from top retrievals
Tail Queries	More robust handling of low-frequency, long-tail search terms

---

📈 Real-world Example

User Query	Default Behavior (No ANCE)	With ANCE Models
“galaxy fold 5 pro”	Returns Fold 4, unrelated accessories	Prioritizes Fold 5 models

“best Samsung phone for gaming”	Returns generic S-series and A-series	Boosts S23 Ultra, Galaxy Z Fold with Snapdragon
“Samsung Tab with keyboard”	Shows standalone tablets or random keyboards	Boosts Galaxy Tab S9 bundles with keyboards

---

Would you like a visual for how this ANCE loop could work inside Samsung’s ecosystem?

Great — let’s explain this ANCE-based process using a **Samsung e-commerce search** example, focusing on **improving bi-encoder and cross-encoder models** for **robust and relevant product ranking**, especially for long-tail or ambiguous queries.

🔍 Real-World Scenario: Samsung E-Commerce Search

Let’s say we have the following **user queries** on the Samsung store:

1. **Query A:** “purple dress for women”
2. **Query B:** “women’s purple outfit”
3. **Query C:** “Galaxy phone with pen”
4. **Query D:** “Galaxy S24 Ultra with stylus”

🎯 Goal

Build a system that understands these are **semantically similar** queries and ranks **similar relevant products** (like Galaxy S24 Ultra, Galaxy Note series, or specific smartwatches and clothing).

🔁 Step-by-Step with Samsung Context

1. Bi-Encoder Self-Retrieval (ANCE Mining Phase)

a. Retrieve Top-K Candidates

- The **bi-encoder** takes a query like **“Galaxy phone with pen”** and retrieves the **top 100 products** using vector similarity.
- Products returned might include:
    - Galaxy S24 Ultra

- Galaxy Note20
- Galaxy Tab S8 with S Pen
- Some unrelated accessories like pens, or pencil cases (false positives)

b. Subtract Co-Purchase Pairs

- If a user also searched “Galaxy Tab S8 with Pen” and bought it, the system recognizes this as a **co-purchase** and **excludes it** from the pseudo-positive training set.
- Why? These pairs may reinforce accidental biases — we want **semantic similarity, not behavioral shortcutting**.

c. ANCE Dataset Creation

- Remaining (query, product) pairs (e.g., “Galaxy phone with pen” → Galaxy S24 Ultra) are used as **pseudo-positive** training examples.
- Hard negatives: other top retrieved products that look similar but aren’t relevant.

---

2. Bi-Encoder Fine-Tuning

- Fine-tune the bi-encoder using this **ANCE dataset**, improving its ability to **retrieve** more relevant products based on semantics (not just co-purchase trends).
- Over time, the model becomes better at knowing that:
  - “Galaxy phone with pen” ≈ “S24 Ultra stylus phone”
  - “purple dress” ≈ “women’s purple outfit”

---

3. Cross-Encoder Fine-Tuning

- Once the bi-encoder is good, pass the same **(query, product)** ANCE pairs to a **cross-encoder**.
- Cross-encoder examines both the query and product **together** in detail.
  - For example:
    - Query: “purple dress for women”
    - Product Title: “Samsung BESPOKE Purple Dress Shirt – Women's Collection”
    - Product Description: “Crafted for smartwear comfort with a lavender hue...”
  - It now focuses on **word-level interactions**, like:

- "Purple" ↔ "lavender"
- "dress" ↔ "shirt"
- "women" ↔ "women's collection"

This fine-tuning allows the cross-encoder to be very precise in **ranking** the best products from the bi-encoder shortlist.

✅ Benefits for Samsung Search

Area	Before ANCE	After ANCE
Tail Queries	Poorly served (few/no results)	Better understanding from similar popular queries
Product Ranking	Biased by co-purchase trends	Semantic relevance dominates
Synonym Handling	Misses semantic variants	Learns that "lavender" ≈ "purple", "phone with pen" ≈ "S Pen support"
Model Quality	Bi-encoder retrieves imprecise results	Retrieves focused list; cross-encoder ranks accurately
Business Impact	Missed opportunities, low click-through	Higher relevance, improved CTR and sales

---

🚀 Final Thoughts

By using **ANCE + continuous finetuning**, Samsung's search can:

- Deliver consistent results for **semantically identical queries**
- Improve **coverage for tail and misspelled queries**
- Enable **robust, multilingual support** (especially in international stores)
- Reduce dependency on fragile signals like click logs

Would you like help designing a custom ANCE training loop or query pair mining strategy tailored to Samsung data?

Awesome — let's walk through a **custom ANCE training pipeline** for Samsung's e-commerce platform. I'll outline the system in 4 stages, keeping in mind your data (queries, product metadata, behavior logs, etc.), and then propose how to build it practically.

⚙️ Custom ANCE Training Loop for Samsung E-Commerce Search

🗒️ Stage 1: Query-Product Mining

1A. Behavioral Signals Collection

- Collect anonymized data:
    - Queries with high/low impressions
    - Clicks, cart-adds, purchases
    - Co-purchase logs (query-product → purchase)

1B. Semantic Duplicate/Variant Query Generation

Use:

- Rule-based normalization (e.g., “tv stand black” ↔ “black tv cabinet”)
- LLM/embedding similarity
- Manual seed lists for tail query variants

Goal: Group these queries into **equivalence classes** (semantically similar).

🔍 Stage 2: Bi-Encoder Self-Retrieval

2A. Retrieve Top-K Products

- For each query variant in a class (e.g., “galaxy tablet with pen”):
    - Use current bi-encoder to retrieve top 100 products
    - Filter to **category-restricted results** (e.g., only tablets)

2B. Hard Negative Mining

- Identify products retrieved but not clicked/purchased
- Exclude **co-purchase items** (i.e., query → product purchased)
- Keep hard negatives and likely pseudo-positives

🧠 Stage 3: Bi-Encoder Fine-Tuning

- Create triplets:

    - (Query, Positive Product, Hard Negative Product)

- Fine-tune your **bi-encoder** using contrastive loss (e.g., in-batch negatives + hard negatives)
- Iterate:
    - After each epoch or few epochs, regenerate retrievals and mine again (i.e., **continuous ANCE loop**)

Output: A high-quality retrieval model trained to surface semantically relevant results.

🤝 Stage 4: Cross-Encoder Fine-Tuning

- Use best ANCE pairs (query, product) from above
- Train a cross-encoder model (e.g., BERT, RoBERTa) to **classify relevance** (binary or graded)

You can use structured inputs:

[QUERY] purple dress for women
[TITLE] Samsung Lavender Smart Dress
[DESC] Stylish lavender casual wear, tailored for women...

-
- Loss: Binary cross-entropy or regression (score from 0–1)

🧪 Optional Enhancements

Idea	Description
Multilingual	Use language-specific fine-tuned encoders (Hindi, Japanese, etc.)
LLM Paraphrasing	Generate synthetic variants of tail queries for training
Cross-Store Training	Use more frequent queries from other regions to help tail regions (e.g., low Hindi traffic)
Category-Aware Encoding	Inject product taxonomy (e.g., Mobile > Smartphones > Galaxy) into model input
Fusion Training	Train a **bi-encoder + cross-encoder cascade** with fusion layers at scoring time

---

💬 Want to Build It?

I can help you:

- Write the pipeline in PyTorch/TensorFlow or HuggingFace
- Design the input formats and sampling strategies
- Simulate ANCE mining on a sample dataset
- Visualize embedding spaces before and after training
- Integrate into your search stack (retriever + reranker)

Would you like the base code or a notebook for the ANCE loop to get started?