ML System Design: Omni Search

Author	@Satheeshkumar Karuppusamy (Deactivated)
IC Levels	*CL3+ (IC2+)*
Time Length	*45m*
Labels	• ml-system-design • data-engineering • system-design
Date Added	Feb 22, 2021

Problem Statement

At Roblox, we have multiple entities in our platform, for example - games, players, avatars and catalog items. We have a landing page for each entity and the users have to use entity specific search boxes to discover interested items of that entity type

Let us assume we are going to build a single landing page where users can search and discover all the entities they are interested in. Could you design an end to end system (especially the intelligence component) to solve this problem? To be specific we have to solve following sub problems:

1. A solution to rank the avatar search (we have log data from naive ranker) results.
2. A solution to blend all these entities (do not have log data for blended results

Solution to the question

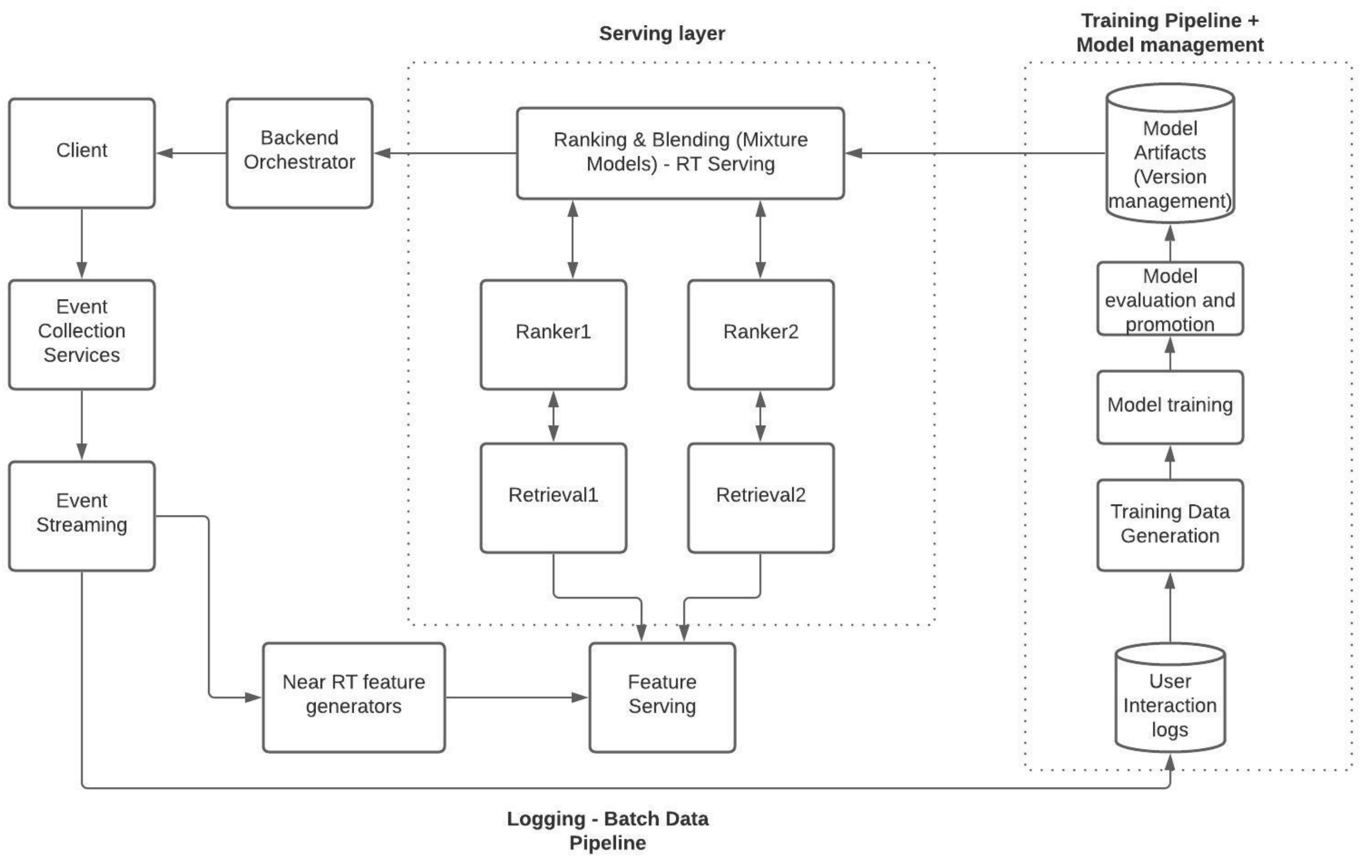
Part1: Problem clarification and understanding

Since this is an open ended problem, the candidate are expected to ask some of the basic followup questions to model it correctly:

- User experience: whether results per entity should be rendered together in a single unified widget or as individual widgets?
- Expected number of entities?
- Number of results per page?
- Product metrics to measure overall success and for each entity?
  - Overall - CTR
  - Game Search - Play time
  - Player Search - Player Engagement
  - Avatar Search - Purchases or downloads
- Any initial data analysis to understand the data from different verticals?

Part2: Higher level design

NOTE: The design given below is just an example. There could be different choices


Serving layer
Training Pipeline + Model management
Client
Backend Orchestrator
Ranking & Blending (Mixture Models) - RT Serving
Model Artifacts (Version management)
Ranker1
Ranker2
Model evaluation and promotion
Event Collection Services
Model training
Retrieval1
Retrieval2
Event Streaming
Training Data Generation
Near RT feature generators
Feature Serving
User Interaction logs
Logging - Batch Data Pipeline

Part3: Deep dive - Designing core components

Since ML is a very broad area and the knowledge level (depth and breadth) varies quite a lot among candidates, please use the following as some potential areas to test the candidate. It will be very hard to see any candidate touching everything given here

3a: Modeling choices

1. Baseline approach:

- Q: Could you explain a very naive solution for building a vertical level ranker?
  - Example: exact/fuzzy string matching

2. Traditional approach (Optional - for candidates with IR background):

- Q: How would you formulate Search as an IR problem
  - Create TF-IDF vector with a game being represented as a document by its title and description & apply LSA.
  - Compute Vector similarity based ranking (cosine similarity, BM25) between the documents and the query vectors to get a bigger pool of highly relevant candidates
- Q: How can you make **IR approaches faster**?
  - Create inverted indices of tokens → documents like lucene index
- Q: What are the pros and cons with very simple IR approaches?
  - Pros: Faster and simple, completely unsupervised
  - Cons: Does not capture the distributional properties of the natural languages better - ends up having low recall

3. Retrieval step (Optional - for candidates with Search or recommendations background):

- Q: Could you explain the **scope** of Retrieval step?
  - Generating larger number of candidates efficiently - low latency with high recall
- Q: Could you come up with some **modeling choices** for retrieval step? some possible choices are as given below:
  - The approaches described above - traditional string similarity approaches
  - Embedding based approach: Two tower model, Matrix Factorization
- Q: What **metrics** would you use to measure the performance of Retrieval layer?
  - Primary metrics: Recall, Recall @ k

4. Ranking step (Mandatory - Avatar Search Ranker)

- Q: Could you explain the **scope** of Ranking step?
  - Given the generated candidates, rank them based on user's preference - **ordering** is more important
- Q: How would you **model** the ranking problem?
  - Learning to rank approaches (similar to the ones we have for next game recommendation question)
  - Pointwise: Playtime per impression (regression)
  - Pairwise: Given two documents, which document is better. Objective is to reduce the number of inversions (reduce incorrect ordering in the result pair). Example: RankNet
- Q: Can you explain the **metrics** for the Ranking layer?
  - Offline Model metrics to compare different model choices and/or parameter settings:
    - General metrics: Precision @ k, Mean Average Precision
    - Metrics to measure positional effects: Discounted Cumulative Gain, Mean Reciprocal Rank
  - Online metrics: Purchases for avatar search

5. Mixing/Blending step (Mandatory)

- Q: What is the **purpose** of this step?
  - Results from different entities/verticals should be blended before sent to the user
  - Based on the product requirements, we could do one of the following
    - Rank widgets in a particular order (Widget level ranking)
    - Blend and rerank the vertical level top results to a single widget (Single widget ranking)
- Q: **Metrics** to measure for mixing/blending step?
  - Offline metrics: MAP, DCG
  - Online metrics: w1 * Play time + w2 * Player Engagement + w3 * purchases

Open ended questions to understand system choices

3b: Training/Inference pipeline

- **NOTE: Focus only for blending step**
- Q: How would you prepare training data for blending step?
  - Widget Level ranking:
    - Compute user-entity affinity by converting user-item impressions from the existing system
    - Build a simple ranker and use it for collecting training data to train a better model
  - Single Widget ranking:
    - Use user-item impressions from the existing log data to bootstrap training data preparation
    - Build a simple model and then use feedback data to prepare training data for a better model
- Q: How do you make sure test data is representative of the actual population?
  - Sampling techniques
  - Importance of random samples - sampling distribution
  - Stratified sampling by query frequency
- Q: Training efficiency: How to speed up the training process - different ways to speed up model training?
  - Distributed training
- Q: Training cadence: How to decide the cadence for the model re-training?
  - If the distributional characteristics of the data changes too often, frequent retraining is necessary or it can be little liberal

3c: Experiments and Evaluation

- Q: Can you explain your Model selection strategy - Given n model choices how would you choose one or set of potential candidates for online experiments?
  - Through offline experiments and compare model's performance through comparable metrics
- Q: Can you explain about the online experimentation strategy?
  - Talk about A/B testing and the higher level related details

3d: Model management

- See if the candidate touches upon the following topics:
  - Measure and monitor change in top line metric over time
  - Model versioning
  - Q: Model promotion: Assume you have an automated pipeline to train a model, how would you decide your new model is ready to be promoted?
  - Q: Model monitoring: How would you monitor if the model we have in production is behaving as expected?

3e: Additional Followups (when necessary - for modeling focused candidates with less exposure to ML systems side of things)

- Q: How to prepare training data if you are going to be building the V1 or V0 of your vertical level ranking model without any log data?
  - heuristic based approaches, semi supervised approaches - start with something + bootstrapping, for NLP and CV - contrastive learning and some unsupervised approaches like LSH, crowd sourced - week labels
- Q: If the network architecture is too big with a large number of parameters (for example BERT), how would you use them in the production inference pipeline?
  - Distillation, quantization and teacher-student models
- Q: Setting up a feedback loop: What kind of additional feedback would you get from the client to improve the performance of the system?
  - Other additional events the model can use
- Q: Importance of online experiments over offline experiments? How would you quantify those outcomes?

Part4: ML coding [Only if necessary]

If we have more time left, we can use last 10 mins to ask them to implement one of the metric that we have discussed in the interview: either P-R curve, ROC curve or other similar metrics to gauge candidate's depth of understanding.

NOTE: In this round, the purpose of this question is not to check exact implementation of these metrics. Just to understand if they have core understanding of the metrics and little bit of hands on skills - so its enough to write the pseudo code to convert understanding into acceptable solution

ROC coordinates computation

```
def compute_fpr(y_gold, y_pred):
    """
    Computes false positive rate = FP/(FP + TN)
    """
    fpr = FP/(FP + TN)
    return fpr

def compute_tpr(y_gold, y_pred):
    """
    Compute true positive rate = TP/(TP + FN)
    """
    tpr = TP/(TP + FN)
    return tpr

def roc(y_gold, y_pred_probs, bucket_size=10):
    """
    Args:
        y_gold: ground truth
        y_pred_probs: predicted prob label for the data points
        bucket_size: bucket size to compute roc thresholds
    """
    thresholds = [current_index/100 for current_index in range(0, 100, bucket_size)]
    roc_coordinates = []

    for current_threshold in thresholds:
        y_pred = [
            1 if current_prob > current_threshold else 0 for current_prob in range(y_pred_probs)]
        roc_coordinates.append((compute_fpr(y_gold, y_pred), compute_tpr(y_gold, y_pred)))

    return roc_coordinates
```

Hints to the Candidate

Please look at the indented suggestions for each question and give hints accordingly

Test Cases

N/A

Notes to the Interviewer

Given the breadth and depth of the Machine learning field, it will be very hard to expect every candidate to know everything. Recommendation is to use this question to evaluate the following skills:

- Ability to map a business problem to an ML problem (necessary skills for senior candidates)
- Ability to structurally solve a ML problem: In a day to day life if a candidate is working on any ML problem do they have acceptable clear process to do it
- ML problem solving skills: Are they able to
  - define the necessary components at the higher level
  - model individual sub problems clearly
  - make reasonable algorithmic choices
  - come up with clear experimentation plan
  - establish metrics to quantify the model performance
- Breadth and depth of their ML knowledge and ability to apply that to a new problem
- Tradeoffs in their systems choices depending upon the use case
  - data store
  - training pipeline
  - model management
  - serving pipeline

What we are looking for from the candidate

Identify key things that might separate strong candidates from weaker candidates. Also discuss what might be expected of different IC levels if the question is appropriate for multiple levels.

Level	Key expectations
IC2	▪ Overall: Need not follow proper structure, need not drive the interview ▪ Finish Part1 with some hints ▪ Finish Part2 with good understanding of Serving layer and reasonable understanding of other layers ▪ Finish Part3 with good understanding of 3a, 3b and 3c with minimal hints - reasonable depth and can lack some breadth ▪ Optional: Part4 - Finish with few hints
IC3	• Overall: Need not follow proper structure, need not drive the interview • Finish Part1 with some hints • Finish Part2 with good understanding of Serving layer and reasonable understanding of other layers • Finish Part3 with good understanding of 3a, 3b and 3c with minimal hints - good depth and can lack some breadth • Optional: Part4 - Finish with few hints
IC4	• Overall: Expected to follow proper structure, expected to drive the interview • Finish Part1 with almost no hints • Finish Part2 with few hints • Finish Part3 with good depth and breadth - breadth can be sacrificed for commendable depth in one of the areas • Optional: Part4 - Finish with no hints
IC5+	• Overall: Expected to follow proper structure, expected to drive the interview and suggest alternative options based on different scenarios • Finish Part1 with almost no hints • Finish Part2 with almost no hints and have additional suggestions based on different scenarios • Finish Part3 with good depth and breadth - in terms of breadth candidate need not know all the details, at least they should have some level of exposure and basic ideas • Optional: Part4 - Finish with no hints

History Log

This section is used for when interviewers are asking the question to record feedback about the interview - particularly if assessment criteria seem off, the solutions should improve, or there is something confusing in the question. In addition the interviewer can mention what they suggest for changes to the original question creator.

Date	Candidate Level Range	Notes / Feedback
mm/dd/yy		
mm/dd/yy		