# VIDEO RECOMMENDATION

**ONLINE METRICS:**

1. CTR
2. Reaction e.g. like, upvote/downvote, rating
3. Num. of completed videos
4. Watch time as a % of total session time

**LABEL CRITERIA:**

+ve :

1. Clicked
2. >80% watched
3. +ve reaction e.g. like, upvote, high rating

−ve :

1. Not clicked (but have impression)
2. <10% watched
3. −ve reaction e.g. report, dislike, downvote, low rating

**FEATURES:**

**User Features**

Demographic info:
- age
- gender
- location
- Language

Account info:
- account age

**User-Item Interaction features** → all item historic interaction

- avg. session time
- Seq. features
  - Search history
  - Liked videos
  - Watched videos

**Item Features**

- Lang
- Titles & Tags
- Duration
- genre
- time elapsed since release
- time on platform
- engagement (e.g. likes, upvote, watched) (rate & counter feature)
- maturity rating
- Modality of data features (e.g. video feature-engineering)

**Context Features** → user's context when interaction happened

- Time of day
- Day of week
- is_holiday
- device

**Sparse Features**

- User Id
- Item Id

— Rate & Counter features
— Context Features : Sine & Cosine fn for feature-engg. (to show 23 is closer to 1)
— Sparse Features : Deep hash Embedding

MODELLING:

- → Candidate Generation : — Multiple Ways
- → Filtering — Geography restriction
  - — Age restriction
  - — Already watched
  - — De-dupe
- → Ranking — Multi-task learning
- → Re-ranking : — Diversity
  - — Freshness

CANDIDATE GENERATION:

- 2 Tower: User features + User-item historical interaction (User Tower) + Context + sparse features related to user eg. user id

  (Item Tower) : Item features

- Sampling for 2 tower & Bias removal — Popularity — Fairness — Privacy — Position
- Matrix Factorization Vs 2-tower
- ANN: Quantization & Indexing
- Evaluation Metric: Recall based

RANKING

- Pointwise, Pairwise & Listwise : Pros & Cons
- Single Task vs Multi-task learning: Pros & Cons
- Multi-task learning: Shared bottom Vs MMoE
- Single Task or Expert NW in MMoE: ~~Deep~~ and Wide NW or Factorization Machines/DeepFM
- Sampling: Balancing Bias removal — Popularity — Fairness — Privacy — Position

  Train-Test split — time, user basis basis
- Loss fn: Focal loss
- Features: Cand. generation score + Avg. Embedding + Any specific user-item interaction feature
- Evaluation Metric: Precision based — MAP — NDCG

  Diversity

SPECIAL ISSUES IN RECSYS:
- Cold start of users & items
- Serving bias, position bias, popularity bias
- Calibration

# FEED RANKING

→ No. of times feed need to be refreshed each day for each user

→ Any specific engagement to be optimized or weighted engagement
(e.g. like, comment, share, etc.)

MODEL ARCHITECTURE

→ Multi-stage : 1) Candidate selection (not 2-tower, heuristic based)
2) Ranking (multi-task classification)

→ ML Objective : Maximize wt. score based on both explicit & implicit reaction

→ ONLINE METRICS : 1) Impression duration (e.g. time spent viewing the post)
2) Reaction rate
3) Hiding | reporting rate
4) Sessions with reaction / Total # of sessions
5) Retention : re-visit after x duration
6) Avg. Session duration
7) No. of sessions in x days

→ LABEL : Posts with explicit reactions = +ve
(like, comment, share, etc.)

Posts with implicit reactions = +ve
(impression duration > 2 sec for passive users)

| | Like | Comm | Share | Subscr |
|---|---|---|---|---|
| Post 1 → | 1 | 1 | 0 | 0 |
| Post 2 → | 1 | 1 | 1 | 0 |
| Post 3 | 0 | 0 | 0 | 1 |

Posts with hide | report = -ve

→ CANDIDATE SELECTION + (FILTERING) :
1) Posts from 1st degree connections (unseen or impression duration < tx)
2) Post from pages / people you follow
3) Post from groups joined
4) Trending & popular posts in your geography, age grp
5) Posts with high engagement in your network
6) Posts related to your interest

→ FEATURE-ENGINEERING (For RANKING MODEL): Can use cross-features

Item: Textual content, Image Or Videos, Reactions, Hashtags, Post age
~~Post popularity~~, ~~Post engg~~

User: Demographics: age, gender, lang, location, account age
Context: time/day of week, device, is_holiday
User-"all" item historical engagement: Rate & Counter features
e.g. reaction to text/video/image content
avg. no. of hashtags used in last 3 months
top topics interested in

User-"this" item engagement areas: topic similarity, posted by friend/family, mentioned in post

User-post's author engagement: Like/comment/Share rate
Length of friendship

Author: Degree of influence, historical trend of engagement on author's post

Sparse f: Post id, Author Id, User id

→ RANKING MODEL: (+RE-RANKING) Multi-task (Shared layers + Head for each engagement)
Loss fn: sum of loss for each task)

→ OFFLINE METRICS: NDCG@K, MAP@K

# FRAUD/ABUSE/RISK ML SYSTEM DESIGN

**Issues:** 1. Delayed Labels / concept drift / continual learning

- proxy label
- Re-formulate the ML problem with time horizon
- Domain Experts/Manual annotation
- Unsupervised Anomaly Detection: Auto-encoders (reconstruction loss above threshold) (also if labels are unavailable)
- Supervised Anomaly Detection: one-class SVM with one-class classification (if labels of one class known), Isolation forest, training only on majority class labels)

2. Imbalanced data:

- Under-sample majority class / over-sample minority class → SMOTE
- Metrics: Precision/Recall/F1 instead of Accuracy; PR AUC instead of ROC AUC
- Cost fn sensitive learning, e.g. scale-pos wt in Xgboost

3. Explainability (Human in the loop - business rules → Association rule mining; supervised ML cannot stop new, unseen or evolved fraud patterns)

4. Anonymize data for sensitive info

5. Point-in-time values: Using external APIs to fetch data, e.g. email related APIs. Need to make sure that at the time of training, the values of these APIs should be from the point in time when event occurred (e.g. transaction happened)

## TYPES OF FEATURES → Identity / Behavior

- Affinity features: based on counters: how many times two features with certain values were seen together (e.g. how many times email + IP was seen together in last 15 mins)

- Velocity features: no. of transactions from this IP in last 15 mins

- Reputation features: reputation of email domain
  no. of tickets purchased in last 6 months

- External API based features: e.g. credit score of user, lat-long of IP

- Profile based features: e.g. zip code

## MODELING:

i) Supervised: Xgboost / LightGBM

ii) Temporal / Time-series: Convert it into supervised ML by adding temporal info as features e.g. day, time, week etc
+ using a time-window and crafting features e.g. tsfresh

iii) Graph ML: GraphSage → can be trained in batches / can predict on unseen node

# HARMFUL CONTENT / COMMENT

→ Define 'harmful' : different categories

→ Business obj : increase platform safety

→ Some categories of harm ~~need~~ need real-time system and some categories could be batch

→ ML Objective : accurately predict harmful post / comment

→ Real-time or combination of real time + batch

→ If real-time and # of posts/comments per unit time huge : (Multi-stage)
filter by 1. new accounts post / comment
2. Old accounts post / comment : historical reporting

→ Multi-~~class~~ classification (new post posted on platform → classify into diff. catg. of harm)

→ ONLINE METRICS : 1) Missed/FN : # of posts reported by users
2) FP : # of posts reported which were found to be not harmful (valid appeal)
3) Im~~pression~~ count of posts that were reported
4) Some ratio of above

→ LABEL : i) Annotated
ii) User reports (threshold by certain # of reports)
- Sample to make sure all categories of harm are appropriately represented
- wt examples by recency (constantly need to catch up)

→ FEATURES: Poster, Post, Comment, Commenter

(identity features: email, ip, phone no. etc.)

Velocity features (rate & counter features): # times transaction happened in last x hrs etc.

**Poster**
1. Violation history
   - # of violations
   - # of user reports (by time range)
   - # Profane words (by time range)
2. Account age, # of followers, followees

**Post**
- Embedding for diff. modalities
- → Profane word used
- → sentiment
- → context features for post
- → Lang.
- → Reactions

**Comment**
- → Embedding
- → Profane words
- → Sentiment of each comm & aggregated
- → Relevance of comment to post
- → Comment creation velocity
- → Lang.
- → Reactions

**Commenter**
1. Violation history
2. Account age, # of followers, followee

Poster - Commenter historical interaction
Poster - Comment → reaction of poster to each comment
Post - Comment → relevance of comment to post

→ MODELING: ~~Multi-stage~~ : Pros & Cons w.r.t. alternatives

1. Single binary classifier (one model): i) cannot determine catg. ii) diff. to improve in a category if not doing well in that category
2. Multiple binary classifier (multiple models): i) Train & serving multiple models is expensive/inefficient
3. Multi-label (one model): same feature transformation for all categories → not ideal
4. ✓ Multi-task (one model):
   1) Training & serving: not expensive
   2) shared layers transform features beneficial to all categories (prevents redundancy)
   3) Training data for each category contributes to learning of other tasks (especially useful if labeled data for one category is very limited)

→ OFFLINE METRICS: ✓ Recall + False +ve Rate, AUCPR

→ SERVING: Usually in 2 forms:
- ✓ Low confidence prediction → Manual Review
- ✓ High confidence prediction → Remove

Model Score Use

# PERSONALIZED SEARCH

ONLINE METRICS:
1. CTR (Click / impression)
2. Average Position of Click
3. Further engagement e.g.
   - dwell time > t sec
   - content watch duration
   - purchase / conversion
4. Time to success (Click + dwell time / content played / purchased)
5. # of similar searches in a session (Counter metric)

LABELS:

+ve:
1. Clicked OR
2. Clicked + further engagement above a threshold

-ve:
1. Not clicked (but impression)
2. Clicked + further engagement below a threshold

— Human labellers / annotated data

FEATURES:

Query Processing: Typo Correction / Fuzzy Matching, Stemming / Lemmatization, Synonyms, Removing irrelevant / stop words,

| Query Features | Item Features | Searcher Features | Context Features (context when the query happened) | Query-Doc (Not used in 2 tower) | Searcher-Doc (Not used in 2 tower) |
| --- | --- | --- | --- | --- | --- |
| – Intent<br>– Keywords/Term (BERT embedd.)<br>– Location (e.g. lat, long from where Query originated) | – Keywords / Terms (BERT emb.)<br>– Location<br>– Reputation features:<br>  – PageRank<br>  – Author's reputation<br>– Engagement features (rate + counter features)<br>  – clicks received<br>  – dwell time / purchased / watched | – Demographic<br>  – age<br>  – gender<br>  – location<br>  – lang | – time of day<br>– day of week<br>– device<br>– is_holiday | – Similarity measure<br>  – cosine similarity b/w embedd.<br>  – TF-IDF match<br>– Historical engagement for the (query-doc) pair | – Dist. b/w searcher & item |

IMP: Prev. historical queries of user does not matter as it is as user's query diff. things. However, some characteristics e.g. intent, radius / dist; type of content with high engagement is useful

## MODELING

- → Candidate Generation
  - Semantic: 2 tower (Query-User and Item tower)
  - Keyword: Inverted Index (e.g. elastic search)
    - Why needed?
- → Filtering: Geography restriction
  - Age restriction
  - De-dupe
- → Ranking: Single-Task → Deep & Wide N/W, Factorization Machine / DeepFM
  - vs
  - Multi-task → Shared bottom Vs MMoE

## CANDIDATE GENERATION:

- 2 tower
  - → Sampling
  - → Bias Removal (position, popularity, fairness, privacy)
  - → How/which features go in which tower
  - → ANN: Quantization & Inverted Index
- Eval Metric: Recall based

## RANKING:

- Pointwise, Pairwise & Listwise: Pros & Cons
- Xgboost Vs DNN: Pros & Cons
- Single Task vs Multi-task (Usually single-task)
- Training Data: Balancing & Bias Removal & Train-test split
- Cross-features inclusion (if single task)
- Loss fn: Focal loss
- Evaluation Metric:
  - Top-K
  - Precision based
    - MAP
    - NDCG

# IMAGE / VIDEO SEARCH

→ Online Metrics : CTR, Click + dwell time > t sec, Click + dwell time > t sec + video watched / download (copy link, (if not visit link/address) copyright → then only download)

- # of similar searches in a session
- Sessions with atleast 1 click
- Time to success (Click + dwell time > t sec + any further actions) like video watch duration etc.
- Further engagement : Total watch duration, Video completion rate

→ Labels : Annotated

- +ve : Click + dwell time > t sec (+ any further engagement/user actions)
- -ve : impressions only

Sampling, focal loss (multiple modalities or imbalanced)

→ Features: Image :
- Captions derived from image (git-base model)
- Tag supplied by uploader/article linked
- Image classification: list of all objects + overall scene description + characteristics of image: (e.g. close-up, black-and white)
- Image embeddings (ResNet)

Video : (Same as Image)
- + Video embeddings (Video MViT)
- + duration of video
- + uploader
- + Video popularity

→ Model : 2 Approaches : (Can use combination)

Query : word embedding (blue balloon, blue, balloon → all n-grams)

① 2 tower (+ Ranking optional) If corpus size small


- + User features (if personalized)
- + Video/Image features + Video/Image Tags/Captions embedding.

(2) Query : word2Vector (Dropbox approach)

(Lightweight approach) Image/Video: all tags derived from image classification/video segmentation
+
textual features of video/image → word2Vec

Cosine similarity b/w Query embedding vector (word2Vec) + Image/Video emb. Vector (word2Vec)

(Optional) formal training not required as both are same word2Vec embedding techniques

→ Loss fn: Focal loss

→ OFFLINE METRICS: MAP, Precision@K, NDCG, Top-K

→ Serving / Inference: Video/Image Indexing pipeline (to compute embeddings & ANN)
+
Text Indexing pipeline (to compute embeddings) of ~~textual~~ tags, titles,

# TIME SERIES ANOMALY DETECTION

→ Data is non iid (values in series are co-related)

→ Difficult to get labels i.e. Unavailability of labelled data (or few)

→ STL Decomposition of Time-Series Data:
- Trend
- Seasonality
- Residual (Use this component to detect anomalies)

→ Features ~~for~~ cybersecurity Attack (or intrusion detection)

(can convert time-series into supervised ML problem using feature-engineering e.g. tsfresh package)

- Headers of protocol
- Count features / Velocity features
- Metrics e.g. CPU% used, Events e.g. login attempts, Logs e.g. process / app logs, Traces (end-to-end request path thru n/w)
- → N/W Features:
  - I/O packets
  - I/O drop
  - I/O error
- → App / Process Features
  - CPU util
  - I/O memory
  - Threads
  - file descriptors
- → System features
  - system disk I/O
  - System CPU util.
  - System CPU stats (context switches, system calls, interrupts)
  - System memory

→ Train-test split by time window

→ Balance false +ve with false -ve

→ Eval Metrics: MSE, MAE, RMSE, MAPE (Mean Abs % Error)

→ Model:
- Rule-based
- Statistical:
- Traditional ML: Isolation forest / One Class SVM / K-Means
- DL: Auto-encoders

# PEOPLE YOU MAY KNOW

→ Goal: grow the network

→ Assumption abt avg. no. of connections per user = 1000

→ Assumption: social graph of most users not very dynamic i.e. connections don't change significantly over a short period

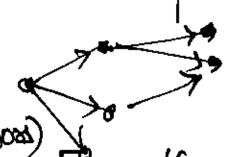
→ ONLINE METRICS:
- Primary: # of conn. req. accepted in the last x days
- Secondary: # of conn. req. sent in the last x days (drawback: does not reflect real growth of n/w since acceptance is also needed)
- Counter: ignore or hide suggestions in the last x days

→ LABELS: +ve: sent + accepted  -ve: hide or ignore suggestions or anything that is not +ve label

→ FEATURE-ENG: Actors: Sender & Receiver

| User Features | Affinity Features: Demographic Affinity | Affinity Features: Education & Work Affinity | Affinity Features: Social Affinity |
| --- | --- | --- | --- |
| - # of connections<br>- # of followers, following<br>- # of pending requests<br>- Account's age<br>- # of received reactions (influence) | - age<br>- gender<br>- lang<br>- city, country | with overlapping time period — School, College, Major<br>Job / Company / Industry | - # of common connections<br>- Profile visits<br>- Common connections wt. by time i.e. how long they have existed |

→ CANDIDATE GENERATION: i) Friends of friends (FoF) (2-hop neighbors)



Avg 1000 conn per user
⇒ 1000×1000 FoF on average
(search space reduction from Billion to Million)

ii) Users from common groups

→ Limit cand.: threshold on # of common connections (e.g. 10, 20 etc.)

→ Metric: Recall oriented

→ Cold-start of new user: i) Based on profile info
ii) Influential users
popular

→ RANKING: i) Single Task : (Binary classification)
ii) Factorization Machines : Adv.
iii) GNN : GraphSage, Link prediction, Node embeddings
adv.

→ OFFLINE METRICS: MAP & NDCG

→ ONLINE VS BATCH PREDICTION:

| ONLINE | BATCH |
|---|---|
| ↳ Pro : Computed only for users who login or visit homepage (fraction of total users) | → Pro : Recommendations calc. beforehand, no delay |
| → Con : Since recomm. are calc. on the fly, if takes long time then user experience is poor | → Con : Need to calc. it for all users |

DECISION: BATCH
- no delay, better user experience
- calc. it only for active users
- social graph does not change quickly ie, recommendations remain relevant for extended period

→ RE-RANKER: Diversity + Bump-up new user

→ POPULARITY BIAS : Limit users who have high no. of connections

→ ~~[illegible]~~

→ DELAYED LABELS: Acceptance can take time, data analysis on historical acceptance time period should inform when a recommended connection is -ve label

# AUTOCOMPLETE OR TYPEAHEAD SYSTEM

→ Latency: Model inference on almost every keystroke p90 < 50 ms

→ Personalization: Users have their own style, need to capture uniqueness of their personal style

→ Fairness & Privacy: Sensitive & confidential info should not be part of suggestions

→ GOAL: i) Min. keystrokes needed to reach first relevant item
ii) Rank user's INTENDED Query at top

→ ONLINE METRICS: i) Avg. Keystroke to get first relevant/click item
ii) Avg. position of click on the suggestion list
Counter Metrics: iii) Avg. No. of similar searches before first click

→ LABELS: — Query log of all users (decayed by time recency) Prefix ~ suffix
— Convert into prefix - suffix pairs e.g. gdp | of Europe
(multiple prefix-suffix pairs for a single query log)
— Remove ~~bias~~ sensitive/confidential info from the query log

→ FEATURES: (Mostly used in Ranking phase)

**Prefix Features**
- Prefix length
- Does it end with a space char? (users most likely to peek at suggestions after a full word)
- Time of typing

**Candidate Features**
- suggestion length (char & words)
- Cand. Freq. in query log for diff. time periods e.g. 1 week, 1 month
- Location Notion?
- Time Notion?
- Cand. freq. in query log submitted by users in same age/gender/region group

**User Features**
- Typing speed
- Location of user
- Age
- Gender
- Language

Demographic features

- Avg. length of suggestions user clicked in the past
- Similarity b/w suggestion words & words in prev. queries in same session e.g. cosine similarity, edit distance

**Previous Search History of User Features**

(cand/Prefix) candidate & Search History

session history (short-term) — all prev. search history across sessions (long term)

— n-gram similarity
— embedding similarity — current cand.

$q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow \dots \; q_t$

(Session history)

- Term combination: added, removed
- Query similarity: cosine similarity
- CTR

→ MODELING: CANDIDATE GENERATION

i) Trie (historical queries along with popularity) & inverted Index (term → location of content)

Issue: A significant proportion of queries issued daily have never been seen previously

ii) From query log, build prefix-suffix pairs → Seq2Seq Model using LSTM

iii) Using LLMs

RANKING: Factorization Machines (After adding features to candidates)

→ OFFLINE METRICS : Cand. Generation: Recall based

Ranking: MRR

Minimum keystroke length (MKS)

→ RE-RANKING / FILTERING PHASE: Remove suggestions having

i) Age restriction ii) Geography restriction

iii) Profane words

Bump up suggestions i) Location-sensitive

ii) Time-sensitive

→ SPELL CORRECTION OR NON-PREFIX MATCHING:

i) Build a model with common misspelt words → mapped correct words

ii) While searching in trie, if no suggestions or less popular suggestions jump to a diff. node paying a cost dictated from conversion table.

Conversion Table is created with misspelt words → correct words along with penalty

iii) Find other words with minimum edit distance