OPEN SOURCE REPO REVIEWS · NOVEMBER 20, 2026

How the X For You Algorithm Ranks Posts: What the Open-Source Code Says About Growing an Account

A code reading of xai-org/x-algorithm: the For You feed scores each post as a weighted sum of predicted actions. Replies, shares and copy-links count most, reports and mutes count most against. Small accounts get a narrow cold-start slot. Model weights and live config are not published.

11 min read · 14 sources · Updated Nov 20, 2026

2026 | For You feed | X algorithm | creator growth | open source | ranking weights | visibility filtering

CONTENTS

The short version 2
How it works 3
Stage 1: where candidates come from 3
Stage 2: enrichment and pre-scoring filters 4
Stage 3: scoring 4
Stage 4: selection, visibility and blending 4
What gets weighed 4
Adjustments after the weighted sum 6
Cold start for small authors 7
What lowers or blocks it 7
What this means in practice 8
What the code doesn't tell us 9
Bottom line 10
Sources 10

The code does not reward a post for raw likes or follower counts. A neural model (Phoenix) predicts how likely each viewer is to reply, share, click, dwell, follow, mute, block or report each post. A fixed formula then turns those predictions into one score, so growth comes from making posts that specific viewers are likely to respond to, mostly by replying and sharing, while staying out of the negative-feedback and safety filters. This report reads the repository at commit `ed78e0645c` (20 November 2026).

5.0	20.0	-234	48 h	<200
Reply weight (base) +15 for mutual-follow originals	Share-via-copy-link weight	Report weight (strongest penalty)	Max post age in the feed	Impressions that make a small author's post a cold-start candidate

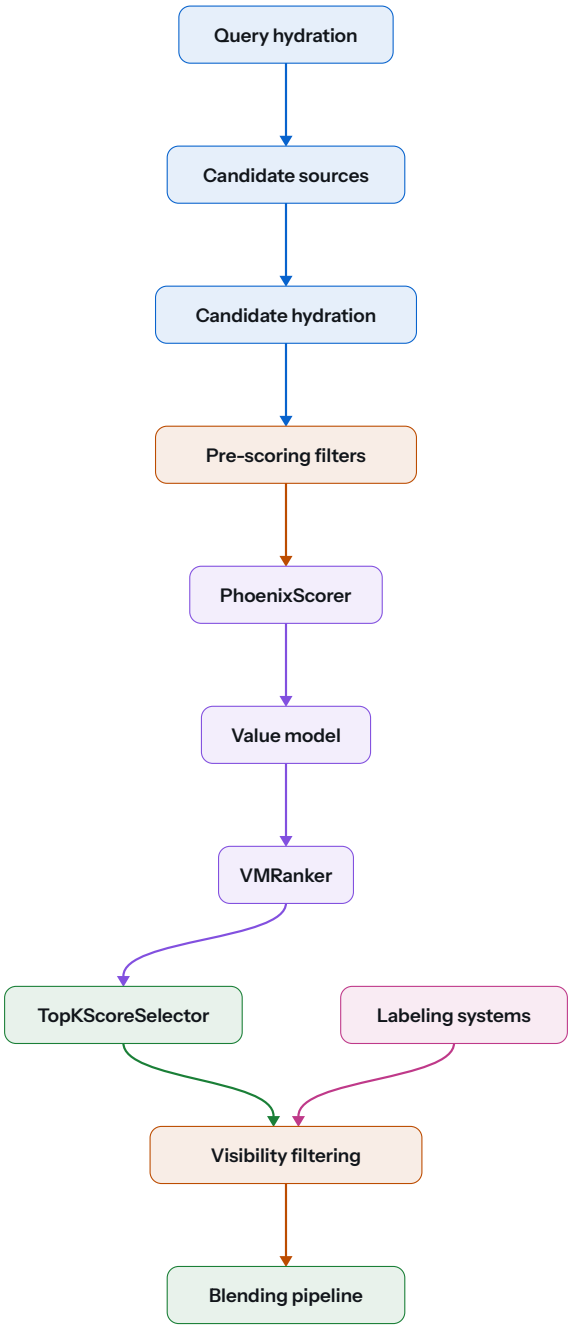
The short version

- **The score is a weighted sum of predicted probabilities**: `Final Score = Σ (weight_i × P(action_i))`. The weights multiply the model's prediction for each viewer, not raw engagement counts README, scoring.rs.
- **Conversation and sharing weigh most.** Defaults are reply **5.0**, quote **5.0**, share-via-DM **5.0**, follow-author **4.0**, share **2.118** and share-via-copy-link **20.0**. A like is only **0.708** param.rs.
- **Negative feedback is heavily penalised.** Report **-234**, mute **-58.8**, not-interested **-53.07**, block **-31.2** param.rs.
- **Mutual follows get a reply boost.** For original posts from someone you follow who also follows you, the reply weight is 5 + 15 = **20** weights.rs, boost doc.
- **Posts older than 48 hours are dropped** before scoring config.rs.
- **Small accounts have a cold-start path.** Original posts from authors with at most 50,000 followers, under 200 impressions and under two hours old can be lifted up the ranking author_cold_start.rs.
- **Posting many times in a row is discounted.** The vm-ranker service defaults decay each additional post by the same author toward a floor of 0.25 vm-ranker/params.rs.

How it works

For You request path

Fetching · Filtering · Scoring · Selection and mixing · Labeling path



Arrows follow the data. Built from the README request path and phoenix_candidate_pipeline.rs.

Stage 1: where candidates come from

Posts come from two pools. In-network posts are recent posts from accounts the viewer follows, held in memory by `thunder/`. Out-of-network posts come from Phoenix retrieval, SimClusters and a shared popular-posts list README. The pipeline definition registers nine sources: Thunder,

popular posts, tweet-mixer, SimClusters, three Phoenix variants (plain, topics, mixture-of-experts), a semantic-ID source and cached posts phoenix_candidate_pipeline.rs.

Phoenix retrieval is a "two-tower" model. One side encodes the viewer's engagement history. The other encodes every post as semantic IDs (codes derived from the post's content embedding) plus a hashed author ID. Posts are returned by dot-product similarity. The code states that production retrieval carries no per-user ID embedding, so a viewer is represented by what they interacted with phoenix README. The README notes an October 2026 update that also trains retrieval to surface posts people read all the way through.

Stage 2: enrichment and pre-scoring filters

Hydrators attach author details, media info, engagement counts, language, subscription status and, importantly, whether the author follows the viewer back (`BidirectionalFollowHydrator`). About 21 filters then run in order (see the blocking section).

Stage 3: scoring

`PhoenixScorer` gets the model's predicted probability for each action. The Phoenix ranking model reads the viewer's recent history and scores each candidate. Candidates cannot attend to each other, so one post's score does not depend on which others are in the batch phoenix README. `value_model.rs` loads the weights and calls `fuse_heads`, which sums `weight × prediction` over 30 terms value_model.rs, scoring.rs.

```rust
fn apply(score: Option<f64>, weight: f64) -> f64 {
    score.unwrap_or(0.0) * weight
}
```

The `VMRanker` scorer then computes local scores, applies the author cold-start logic, and calls a separate `vm-ranker` service. That service reranks using a determinantal point process (DPP), which trades a little score for less similarity between neighbouring posts. If the call fails, the feed is served from the local weighted scores vm_ranker.rs.

Stage 4: selection, visibility and blending

`TopKScoreSelector` keeps the top K by score. `VFFilter` then asks `visibility-filtering/` about each surviving post and removes any answered "drop". Ads, Who to Follow and prompts are blended in afterwards README.

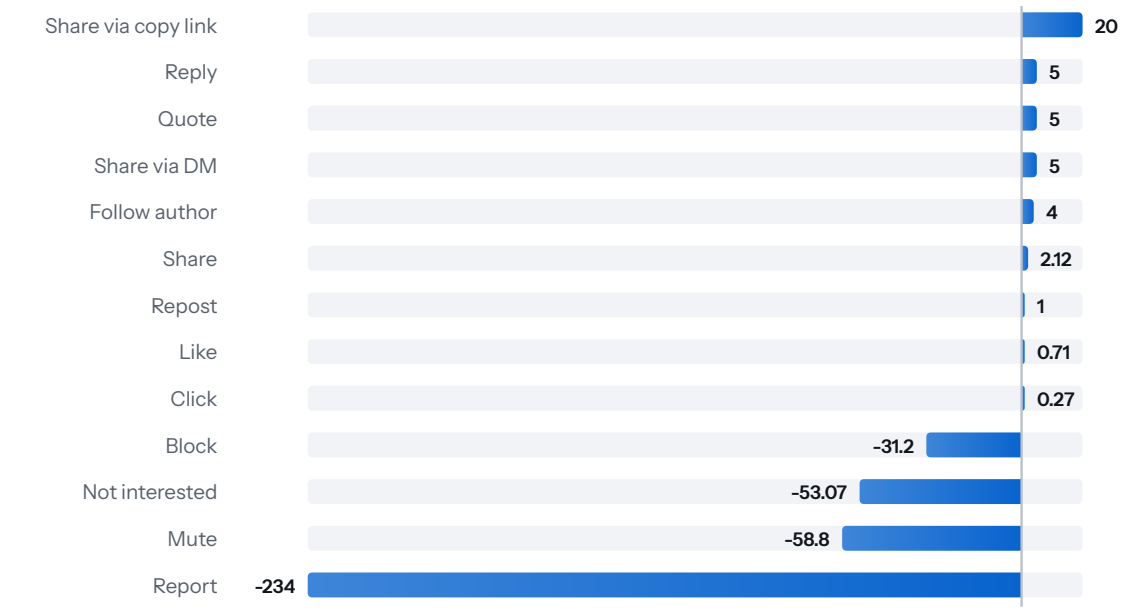
What gets weighed

Weights below are the defaults in `home-mixer/params/param.rs`. The README says cron scripts set these defaults to the primary production values README.

SIGNAL	PLAIN MEANING	WEIGHT	FILE
Share via copy link	Viewer copies the post's link	20.0	param.rs

SIGNAL	PLAIN MEANING	WEIGHT	FILE
Reply	Viewer replies	5.0	param.rs
Quote	Viewer quote-posts	5.0	param.rs
Share via DM	Viewer sends it in a DM	5.0	param.rs
Follow author	Viewer follows the author from the post	4.0	param.rs
Share	Generic share	2.118	param.rs
Repost	Viewer reposts	1.0	param.rs
Like	Viewer likes	0.708	param.rs
Click-dwell time	Predicted seconds on a post after a click	0.385 per second	param.rs
Click	Viewer clicks the post	0.274	param.rs
Open link	Viewer opens a link	0.2	param.rs
Dwell time	Predicted seconds spent on the post	0.0075 per second	param.rs
Photo expand	Viewer opens an image	0.05	param.rs
Dwell	Viewer lingers on the post	0.05	param.rs
Video open, profile click, video quality view	Not counted	0.0	param.rs
Not dwelled	Viewer scrolls past quickly	-0.02	param.rs
Block author		-31.2	param.rs
Not interested		-53.07	param.rs
Mute author		-58.8	param.rs
Report		-234.0	param.rs

Default action weights in For You ranking



Source: home-mixer/params/param.rs defaults. Weights multiply predicted probabilities, not counts.

The repository warns against reading these as count equivalences. A report's weight is large because its baseline probability is "more than 1000x lower than a Like", so it needs a big multiplier to affect ranking at all param.rs. It also states that mass reporting should not suppress reach broadly: predictions are personalised, and actions only count if they occur on a post served in the Home timeline.

Adjustments after the weighted sum

ADJUSTMENT	DEFAULT	EFFECT	FILE
Mutual-follow reply boost	+15 on reply weight	Original (non-reply, non-repost) posts from mutual follows	weights.rs, inputs.rs
Out-of-network factor	0.75	Multiplies out-of-network posts, and in-network replies/reposts	vm-ranker/params.rs
Author diversity decay / floor	0.5 / 0.25	`(1 - floor) × decay^k + floor`, where k = the author's earlier posts in the pool	scoring.rs, vm-ranker/params.rs
New-user OON factor	0.00001	Present, but new-user age threshold defaults to 0	vm-ranker/params.rs

With decay 0.5 and floor 0.25, an author's second post in one ranking pass is multiplied by about **0.625**, the third by about **0.44**, and later ones tend toward **0.25** (my arithmetic from the formula in `diversity_multiplier`).

Cold start for small authors

`cold_start_base_eligible` requires the post to be neither a reply nor a repost and the author's follower count to be at most the cap. The other tests are age and impressions author_cold_start.rs. Defaults:

PARAMETER	DEFAULT
Follower cap	50,000
Impression threshold	200
Max post age	7,200 s (2 h)
Slot min / max	15 / 16
Thompson sampling	on, top-K 2

`apply_cold_start_decisions` raises the chosen post's score to at least the score at a target rank, so it is a lift rather than a guaranteed placement. Thompson sampling picks which low-impression posts get tried based on how they perform scoring.rs, param.rs. The exact slot semantics sit in the part of the file I did not read in full.

10 Jul 2026 **A/B test starts**
Reply boost assigned at 5, 10, 15 or 20 for a small share of users; most users at 0

13 Jul 2026 **Broad rollout at 20**
Boost of 20 sent to many users while other values keep testing

24 Jul 2026 **Cut to 15**
Set to 15 after results and feedback that people saw too little of what they wanted from accounts they did not follow

The timeline comes from the repository's own worked example of how a parameter change appears in the code boost doc.

What lowers or blocks it

CONDITION	EFFECT	FILE
Post older than 48 hours	Removed	config.rs, phoenix_candidate_pipeline.rs
Viewer already saw or was served the post	Removed	README filter table
Out-of-network reply or repost, or reply with missing parent	Removed (`OONRetweetReplyFilter`)	README
Viewer blocks or mutes author; muted keyword hit	Removed	README
SimClusters post by an adult-flagged author, viewer doesn't follow	Removed	README

CONDITION	EFFECT	FILE
Subscriber-only post the viewer can't access	Removed	README
New viewers: out-of-network post below engagement threshold	Removed	README
Holdouts (`InventoryHoldoutFilter`, `FavHoldoutFilter`)	A configured share removed, chosen deterministically per post and viewer	README
Spam, PDNA or bounce labels	Dropped for non-authors	tweet_rules.rs
Misinfo or low-quality labels	Interstitial, not a drop	tweet_rules.rs
Parent, quoted or reposted post was dropped	Post removed too (`AncillaryVFFilter`)	README
Several posts from the same author	Later ones discounted	scoring.rs
High predicted report, mute, block or not-interested	Large negative contribution	param.rs

Visibility rules read labels attached by other systems: content classifiers (`grox/`), event rules (`scarecrow/`, `botmaker/`), account models (`agatha/`, `bdsm/` for inauthentic behaviour, `user-cred-v2/` for follow-graph PageRank) and enforcement. For recommendations to non-followers, extra rules can drop a post that is allowed for followers README, tweet_rules.rs.

What this means in practice

- **Write for replies and conversation. Shown in code.** Reply, quote and DM-share each carry 5.0, against 0.708 for a like param.rs.
- **Make posts people send to someone or copy a link to. Shown in code.** Copy-link share is the largest positive weight at 20.0, and DM share is 5.0. Whether the model predicts these often is learned, not set in the code.
- **Build mutual follows. Shown in code.** Original posts from mutual-follow authors get a reply weight of 20 instead of 5 for those viewers weights.rs. Replies and reposts are excluded from this boost.
- **Publish original posts, not replies, if you are small. Shown in code.** Cold-start eligibility excludes replies and reposts, and needs ≤50,000 followers author_cold_start.rs.
- **Post when your audience is active. Inferred.** Cold start considers only posts under 2 hours old and under 200 impressions, and nothing older than 48 hours is shown at all. Early timing therefore matters, but the code does not say which hours.
- **Space your posts. Inferred.** The diversity decay shows up in vm-ranker defaults, and the home-mixer scorer switches it off locally, so it likely applies in the remote service. A single strong post per ranking pass beats several.
- **Avoid content that triggers mutes, blocks and reports. Shown in code.** One predicted report costs as much as ~330 predicted likes on weight alone (234 / 0.708). That is a per-probability comparison and not a count rule, as the code comments insist.

- **Don’t expect follower count itself to be scored. Inferred.** I found no follower-count term in the weight list. It appears only as the cold-start cap. The model does see hashed author IDs and the viewer’s history, so familiarity with your account can matter.
- **Don’t rely on brigading or coordination. Shown in code.** The comments say off-timeline actions have no ranking impact and mass reporting mainly affects viewers similar to the reporters.

What the code doesn’t tell us

- **No trained model weights.** The repo has training code and synthetic data generation, not the production checkpoints. The probabilities behind the weights, such as how likely any given post is to get a reply, are learned and not visible phoenix README.
- **Live config differs.** Values are read from a feature-switch system, and the repo’s defaults are synced by cron to “primary production values”. Experiments below about 10% of traffic may not appear README.
- **The two copies of the weights differ.** `vm-ranker/params.rs` has different defaults for some values than the home-mixer copy, so they have drifted:

Same parameter, two defaults

5 changes

ITEM	STAT	BEFORE	AFTER	CHANGE
Favorite weight	home-mixer → vm-ranker	0.708	0.5	Change −29%
Not-interested weight	home-mixer → vm-ranker	-53.07	-47.52	Change +1%
Click weight	home-mixer → vm-ranker	0.274	0.3	Change +9.5%
Dwell-time weight	home-mixer → vm-ranker	0.0075	0.004	Change −47%
Video open weight	home-mixer → vm-ranker	0.0	0.07	Change

Source: home-mixer/params/param.rs vs vm-ranker/params.rs. Which copy decides the final order is not established here.

- **The rerank service is only partly visible.** `vm-ranker` is included, but whether its values are the ones used in production, and how its DPP interacts with the local score, is not confirmed from the code.
- **Some labeling rules are withheld.** The README says some `botmaker-rules` are not in the repository to reduce the risk of gaming. Interstitial and notice rendering is also not in the repo.
- **Ads, Who to Follow and other features** are blended in by a part of the code I only skimmed, so their effect on organic position is not covered.
- **Scope.** This covers the For You feed. Other surfaces (search, notifications, Following) use related but separate pipelines.

Bottom line

The For You code ranks by predicted viewer behaviour, with replies, shares, copy-links and follows weighted far above likes, and reports, mutes and blocks weighted far below zero. Practical growth levers visible in the code are conversation-provoking original posts, mutual follows, and the cold-start path for small accounts in the first two hours. The actual probabilities and live parameter values are not published, so treat the numbers as a snapshot of defaults and not a guarantee.

SOURCES

1 x-algorithm README
https://github.com/xai-org/x-algorithm/blob/main/README.md

2 home-mixer/params/param.rs
https://github.com/xai-org/x-algorithm/blob/main/home-mixer/params/param.rs

3 xai-value-model/scoring.rs
https://github.com/xai-org/x-algorithm/blob/main/xai-value-model/scoring.rs

4 xai-value-model/weights.rs
https://github.com/xai-org/x-algorithm/blob/main/xai-value-model/weights.rs

5 xai-value-model/inputs.rs
https://github.com/xai-org/x-algorithm/blob/main/xai-value-model/inputs.rs

6 home-mixer/scorers/author_cold_start.rs
https://github.com/xai-org/x-algorithm/blob/main/home-mixer/scorers/author_cold_start.rs

7 vm-ranker/params.rs
https://github.com/xai-org/x-algorithm/blob/main/vm-ranker/params.rs

8 home-mixer/candidate_pipeline/phoenix_candidate_pipeline.rs
https://github.com/xai-org/x-algorithm/blob/main/home-mixer/candidate_pipeline/phoenix_candidate_pipeline.rs

9 docs/BIDIRECTIONAL_BOOST_CHANGE.md
https://github.com/xai-org/x-algorithm/blob/main/docs/BIDIRECTIONAL_BOOST_CHANGE.md

10 home-mixer/scorers/vm_ranker.rs
https://github.com/xai-org/x-algorithm/blob/main/home-mixer/scorers/vm_ranker.rs

11 home-mixer/scorers/value_model.rs
https://github.com/xai-org/x-algorithm/blob/main/home-mixer/scorers/value_model.rs

12 visibility-filtering/rules/tweet_rules.rs
https://github.com/xai-org/x-algorithm/blob/main/visibility-filtering/rules/tweet_rules.rs

13 phoenix/README.md
https://github.com/xai-org/x-algorithm/blob/main/phoenix/README.md

14 home-mixer/params/config.rs
https://github.com/xai-org/x-algorithm/blob/main/home-mixer/params/config.rs

Researched and written with claude-sonnet-5-5, from the sources listed above. Figures are as reported by those sources. Exported November 29, 2026 from https://intel.innerpulse.net/doc/how-the-x-for-you-algorithm-ranks-posts-what-the-open-source-code-says-about, where the current version lives.