Whatamix: Blending up feeds with composable recsys DAGs

Abstract

Many features within a product present themselves as a feed recommendation problem. For example, within the same application there may be a need to recommend relevant media to a user in a landing page, suggest the most pertinent accounts for that user to message via a share sheet, or short-list a set of recent app updates to send to the user via a push notification. These feed recommendation systems are often built independently across an organization, even though they can typically be described by a set of common steps such as candidate retrieval, feature hydration, model scoring, filtering, blending and so-on. This independent development leads to code duplication, inconsistent patterns for interacting with external services, disjointed logging and observability, and additional maintenance burden. At Whatnot, our approach to solve this problem early on was to develop *Whatamix*, a platform for describing feed recommender systems as directed acyclic graphs (DAGs) with interchangeable nodes that can be used across different use-cases.

Whatamix serves five of the core recommendation surfaces at Whatnot already - including our main livestream and product feeds - with more in active development. In this talk, we would like to share our experience building and deploying Whatamix including many of its most useful capabilities, as well as many of the learnings and challenges we faced along the way.

Discussion Points

- **Separating DAG construction from DAG execution**: All steps required to produce a feed are fully described during DAG construction which occurs prior to execution. Explicit DAG construction allows for handling of conditional nodes and inputs as defined by request-level information. By fully describing the DAG, execution can examine dependencies and optimize the run-time, including scheduling nodes to execute in parallel as their dependencies are satisfied.
- **Designing nodes for re-use**: While a user's feed of livestreams may look quite different to a feed of product listings, the set of steps used to serve each one is conceptually similar. We use reasonable abstractions where possible, for example working with abstract "feed entities'' that may represent livestreams, product listings or more. In this way, separate feeds can directly re-use nodes in different feed graphs since they represent generic steps such as "retrieve feed entities from a KV store" or "run model inference on feed entities".
- **Allowing for feed hierarchy via sub-DAGs**: Often in feeds, we might also want to present "mini" feeds, such as horizontal scrolling feed units within a larger vertical scrolling feed.

Whatamix handles this by treating portions of the DAG as a sub-DAG and similarly mapping a “mini” feed to a general “feed entity” which can be blended along with other objects.

- **Integrating static, dynamic and experimental configuration**: Feeds may be served differently to different users over time for various reasons including new releases, targeted/gated logic such as geo- or time-specific features as well as ongoing experimentation. Whatamix abstracts this away into a single notion of *feed parameters* which are automatically updated at request-time to reflect the appropriate combination of configuration before the DAG is constructed. This removes the pollution of DAGs with implementation-specific branching and allows for rapid experimentation.

- **Consistent logging and attribution**: Since nodes are standardized, along with their execution, we can easily apply desired logging and observability patterns in one place. For example, all DAG nodes are added to sampled DAG execution logging traces and all DAG nodes automatically add real-time logging of execution success, failure and latency statistics. Beyond operations and debugging, we built a framework for assigning important identifiers to ranking requests and proliferating them across our system, allowing for feed ranking attribution of downstream events out-of-the-box.

- **Challenges in node design**: One of the key challenges we have faced is exactly how to design for flexibility. It is almost as easy to be too abstract - resulting in complicated nodes with lots of configuration - as it is to be not abstract enough - resulting in nodes that support a single operation in a specific DAG, diminishing the value of Whatamix. Our approach has been pragmatic, designing nodes that are clearly useful for 2-3 use-cases, or building quick-and-dirty “experimental” nodes for prototyping before generalizing. This is an ongoing effort.

- **Dealing with blocking execution, timeouts and fallbacks**: As our DAGs have increased in complexity, handling nodes that block thread execution, or nodes whose failures will cause the entire DAG to fail, has been a challenge. Increasing timeouts for external service calls, such as model inference, can lead to overall DAG slowdowns and locking up of resources. In addition, some nodes can be considered optional when viewed from a “core product experience” perspective, and so we have built ways to return default inputs as fallbacks. Tuning DAGs in terms of these runtime characteristics is use-case specific and so our approach has been to provide tools and let product and ML partners decide the appropriate tradeoffs.

Bio

Jared Casale is a machine learning engineer in Whatnot’s Core Machine Learning group, where he is helping to build out foundational strategies for applied machine learning. His focus areas include establishing patterns for scalable recommender systems and fleshing out a broader platform for scalable feature and model development, experimentation and deployment. Prior to joining Whatnot, Jared spent over 13 years as an engineer and applied scientist at Microsoft, Amazon and Meta, including 4 years as the tech lead for Instagram’s Feed and Stories ranking teams where he

was pushing the boundaries of heterogeneous media-type blending, whole slate ranking and interactions with ranking and content delivery.

Company Portrait

Whatnot is the largest livestream shopping platform in North America and Europe. Buyers and sellers connect in real time to chat, build relationships and run live auctions. As a relatively young, rapidly growing company, our machine learning teams are focused on balancing the challenge of providing fast-paced impact and optimization with a platform to support our ecosystem at scale in the future.

Relevance to the Workshop

Whatamix provides a consistent approach to ML excellence across the organization by standardizing and reusing common recommendation system components. Any improvements or updates are shared and automatically applied across all use-cases. By standardizing the approach to logging and attribution, Whatamix makes it much easier to manage ML resources effectively as well as tie ML updates directly to product impact.