R&D

Open Source AI Summit, Abu Dhabi, December 2025

How Spotify builds AI with and for the open-source ecosystem

Mounia Lalmas

There are several definitions; here is the one guiding this talk:

“Open-source AI refers to AI systems whose models, code, or knowledge artifacts are shared openly, such as open-weight models, datasets, or documentation, so others can understand, reuse, or adapt them in their own work.”

Outline

Spotify & Our Approach to AI

Who we are and how AI helps us scale

Engineering Foundations

Open-source platform for AI-enabled engineering

Product & AI Innovation

Teaching a model to “speak Spotify”

Looking Ahead

Multi-angle open-source AI

R&D

Spotify & Our Approach to AI

Who we are and how AI helps us scale

Our mission at Spotify

To unlock the potential of human creativity — by giving a million artists the opportunity to live off their work, and billions of fans the opportunity to enjoy and be inspired by it.

713M+
Active users

184
Markets

100M+
Music tracks

500K+
Audiobooks

7M+
Podcast titles

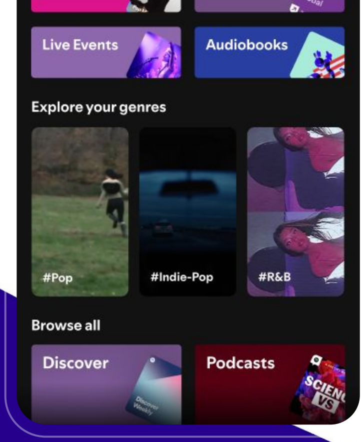
480K+
Video podcasts

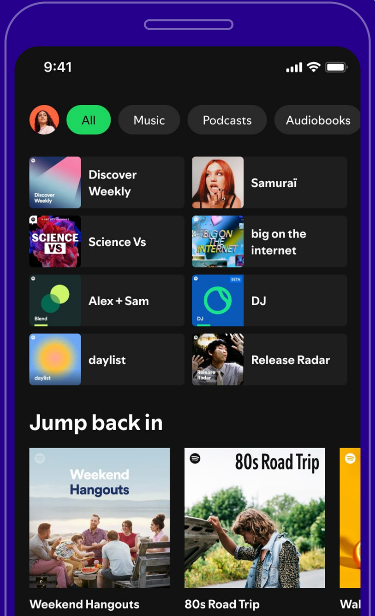
Elevating the Spotify experience

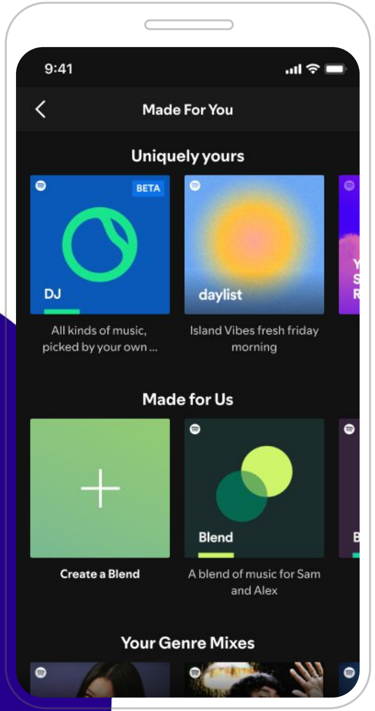
Across surfaces and formats at global scale – music, podcasts, audiobooks.

Powered by

- Engineering foundations
- Large-scale data and model pipelines
- Open ecosystems






9:41
Made For You
Uniquely yours
BETA
DJ
daylist
All kinds of music, picked by your own ...
Island Vibes fresh friday morning
Made for Us
Blend
Create a Blend
A blend of music for Sam and Alex
Your Genre Mixes

Open Source AI at Spotify

Spotify builds with and contributes to open source ecosystems, sharing platforms, research, and tools, and adopting open-weight models to accelerate innovation.

Engineering Foundations

Sharing our open-source platforms (like Backstage), which enable global communities to build better developer experiences.

Product & AI Innovation

Using open-weight LLMs to build generative recommender systems, and sharing methods, findings, and best practices.

Across both, we engage through publications, standards, collaboration, and participation in open source ecosystems.

R&D

Engineering Foundations

Open-source platform for AI-enabled engineering

Developer Platforms for Scale & Velocity

01

Unify documentation, tools, and service metadata in one place.

02

Reduce cognitive load and streamline how teams build, deploy, and operate software.

03

Provide the structured environment that AI systems and agents rely on to work effectively.

Backstage

Spotify's Open Developer Platform

Unify

Brings tools, documentation, service metadata, APIs and standards into **one place**

Scale

Reduces **cognitive load** and gives teams a consistent way to build and operate software

Open Source

Built at Spotify and **open-sourced** so the industry benefits

Backstage at a Glance

Core Platform Components

Software Catalog: Single source of truth for services, dependencies, data assets, websites, ownership, and domains.

TechDocs: Documentation-as-code integrated directly with the Catalog and owned by teams.

Scaffolder / Golden Paths: Reproducible templates for creating new services, pipelines, and infrastructure.

Plugins & Extensions: Integrations for CI/CD, cloud resources, compliance, monitoring, experimentation, and more.

Open-Source Adoption

Open-sourced in 2020 and donated to the CNCF.

Grown to 3,400+ adopting companies across industries.

Community now includes 1,600+ contributors and 230+ plugins, with tens of thousands of contributions each year.

Why Backstage Matters in the AI Era

Backstage gives Spotify a structured, machine-readable map of our software ecosystem, similar in spirit to how foundation models learn structured representations of language and data.

By unifying service metadata, documentation, ownership, dependencies, and workflows, Backstage enables AI to:

- **Reason** about engineering systems
- **Answer questions** accurately with up-to-date context
- **Support discovery** across services, owners, docs, and patterns
- **Take safe, meaningful actions** through standardized workflows and guardrails

Backstage provides the knowledge foundation for **AI-driven engineering,** giving both humans and AI assistants a shared, reliable way to understand and act.

AiKA: Spotify’s AI Knowledge Assistant

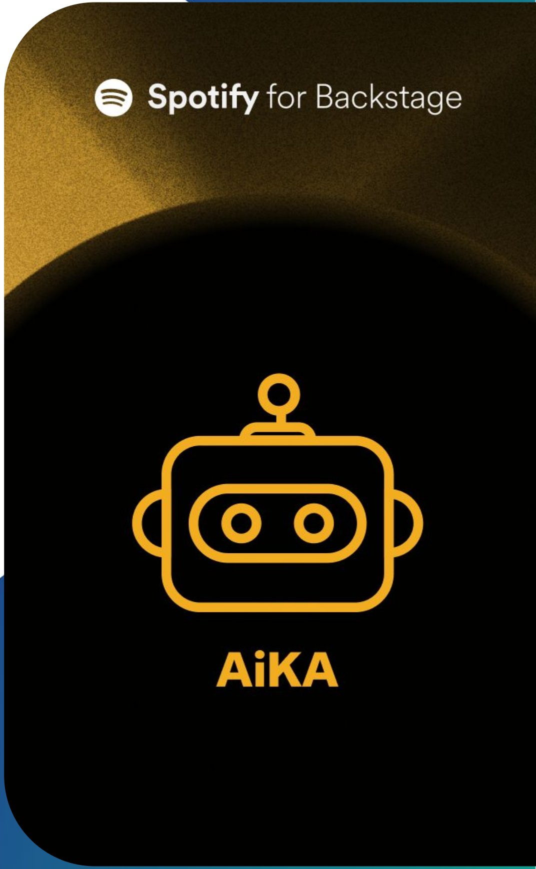
Thanks to Backstage’s structured engineering knowledge, we built AiKA, an AI Knowledge Assistant that helps engineers understand and navigate Spotify systems.

What is AiKA

- A unified AI entry point for understanding and navigating Spotify engineering environment
- A natural-language layer built on top of our metadata, documentation, ownership, dependencies, and workflows
- A foundation for future agentic capabilities in engineering

How AiKA works

- Uses an LLM as the reasoning engine
- Retrieves structured knowledge from Backstage and connected sources
- Uses Model Context Protocol (MCP) to invoke safe, predefined actions
- Runs on top of Backstage metadata and workflows for reliability and accuracy



From AI Assistance to Agentic Workflows

Human–AI Collaboration Flywheel

- Better metadata → better AI assistance
- Better AI outputs → better engineering decisions
- Teams update docs/metadata → AI improves
- Creates a compounding productivity loop

Toward Agentic Workflows

- AI assists with tasks like service creation, dependency updates, migrations
- MCP enables safe, predefined actions
- Scaffolder Workflows connect tools and actions to AiKA
- Lays the groundwork for future AI agents

Open Foundations for AI-Enabled Engineering

Backstage as a platform knowledge layer

- Backstage provides the structured metadata, documentation, ownership, and workflows that AI assistants and future agents depend on.
- It continues to evolve as both an internal platform and an industry-supported project used by thousands of companies.

AI-powered engineering

- With Backstage as a consistent system of record, AI can understand internal systems, trigger workflows safely, and support engineers in maintaining and evolving services more efficiently.

R&D

Product & AI Innovation

Teaching a model to “speak” Spotify

At Spotify personalization is about connecting the things listeners want with the things creators make.

- Entertained
- Focused
- Thrilled
- Motivated
- Sleepy
- Connected
- Relaxed

- Hip Hop
- Sports Podcast
- Daily Mix
- Scenic Route
- This is David Bowie
- Coding Mode
- Chill Mix

Evolution of personalization

Human Curated

- Hand-picked playlists
- Like what you’d hear from a friend or editor
- Example: *Indie Chill* (human-made)

Predictive

- Machine learning predicts and ranks what you might like
- Based on listening history + behavior
- Example: *Discover Weekly*

Generative

- Recommendations adapt to mood, intent, and conversational input
- Produces on-demand curated sets tailored to each prompt
- Example: “*Recommend me some podcasts to go deeper into Spotify's open source work*”

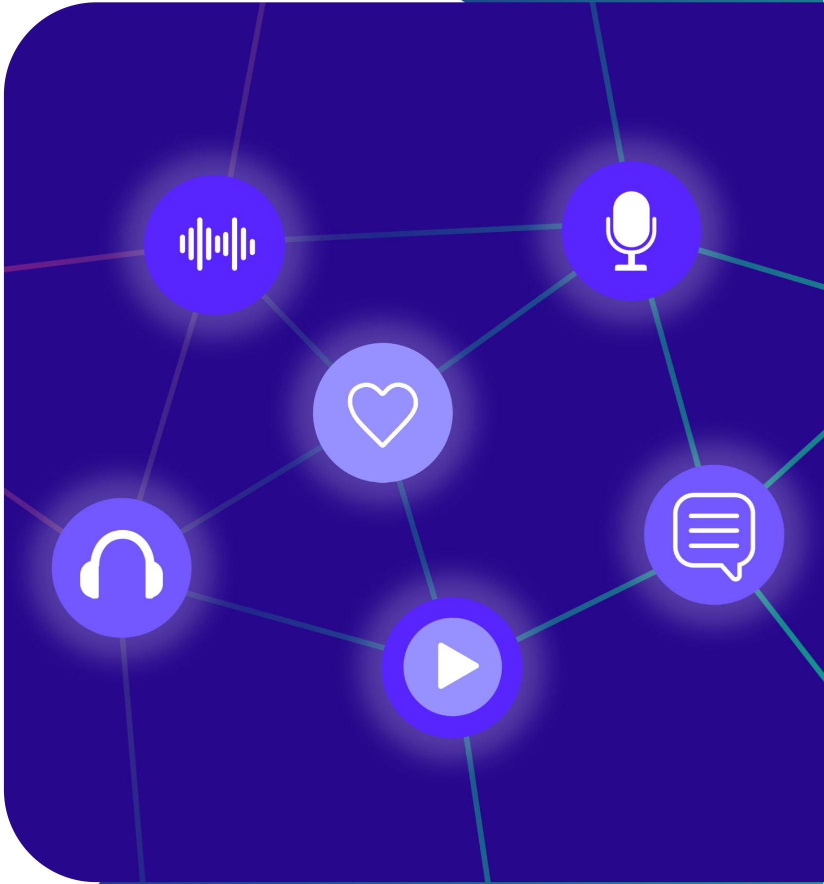
Why LLMs Matter for Personalization

LLMs unlock a new generation of personalized experiences. They bring:

- World knowledge
- Reasoning ability
- Natural-language understanding
- Generative capability

This enables systems that can:

- Understand moods, contexts, intents
- Explain or justify recommendations
- Power more dynamic and conversational experiences



Why Domain Adaptation & Open-Weight Models

Why open-weight LLMs?

- Flexibility to run and tune models ourselves
- Transparent architectures
- Deep integration into Spotify systems

But open-weight LLMs do not know:

- Full breadth and depth of Spotify's catalog
- Our user preferences and listening patterns
- Our content relationships, semantics, and metadata
- How to generate or rank Spotify-specific items

So we adapt them to Spotify's domain

We teach LLMs to "speak Spotify" by grounding them in our catalog, interactions, and semantics, enabling them to generate, understand, and reason over Spotify content.

Using Open-Weight Models for Generative Recommendations

Open-weight LLMs give us world knowledge and flexible architectures.
Domain adaptation lets us turn them into models that truly understand Spotify:

Ground in our catalog

Teach models Spotify’s items, metadata, and relationships

Align with behavior

Adapt to how listeners explore and engage with content

Optimize for goals

Tune for discovery, satisfaction, and engagement

Recommenders use embeddings. LLMs use tokens. They also “speak” different formats.

Modern recommender systems

Represent users & items as **dense continuous vector embeddings** learned from **interaction data**

LLMs

Operate on **discrete token sequences** tied to **a fixed vocabulary** with learned embeddings

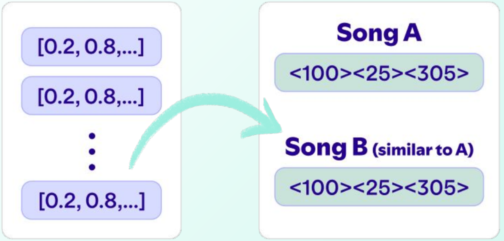
The mismatch

These **distinct data representations are not directly compatible**

LLMs need a way to understand Spotify's catalog, Semantic IDs make this possible.

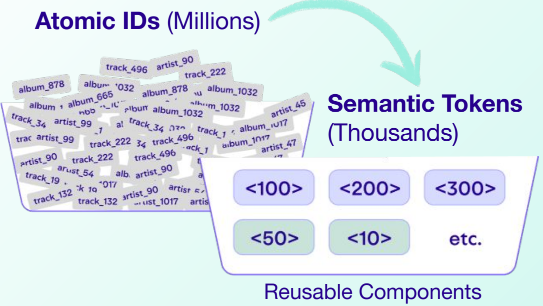
The problem: Atomic item IDs

- Large and sparse vocabulary
- No semantic meaning (e.g., track_491)
- Cold-start problem



The solution: Semantic IDs*

- Convert item embeddings into short token sequences
- Similar items share token components



Why it works:

- Preserves semantic relationships
- Improves scalability, massive reduction in vocabulary size
- Generalizes to unseen items: handle cold-start

*Recommender Systems with Generative Retrieval. NeurIPS 2023
Better Generalization with Semantic IDs: A Case Study in Ranking for Recommendations. RecSys 2024

How we adapt open-weight models to Spotify using Semantic IDs

Make the model read, reason about, and generate using Semantic IDs as naturally as text.

1. **Vocabulary expansion**

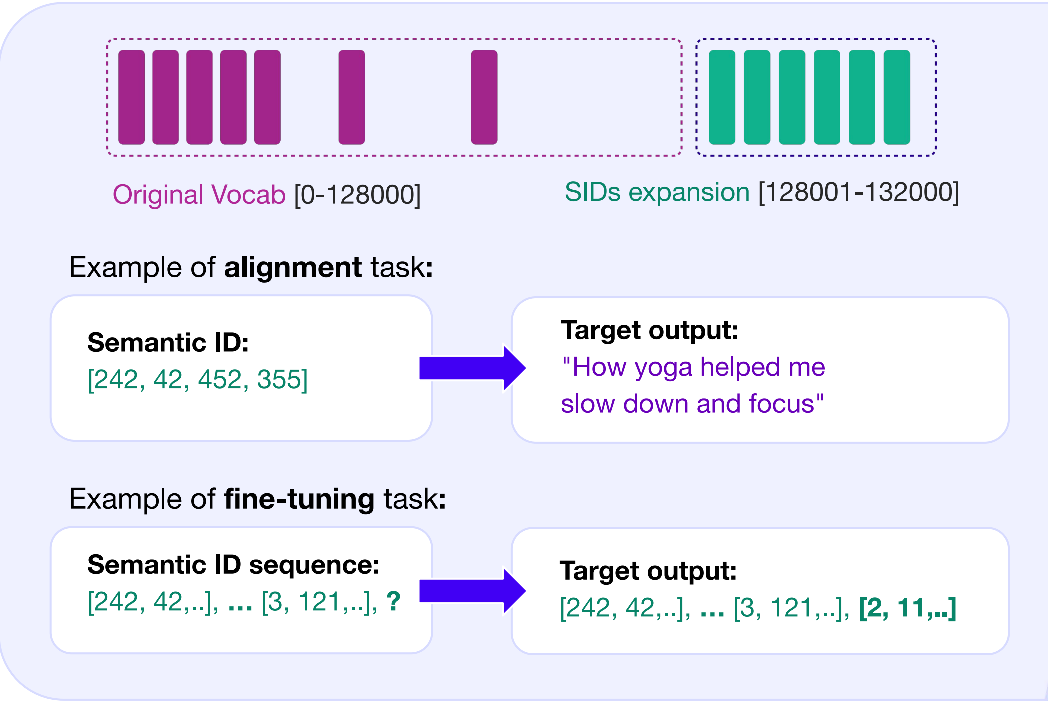
   Expand original LLM vocabulary with Semantic IDs.

2. **Alignment Phase (LLM Frozen)**

   Train Semantic ID token embeddings to align with language.

3. **Fine-tuning Phase (LLM Unfrozen)**

   Train on tasks involving Semantic IDs e.g. next-item prediction.



Original Vocab [0-128000]

SIDs expansion [128001-132000]

Example of **alignment** task:

Semantic ID:
[242, 42, 452, 355]

Target output:
"How yoga helped me slow down and focus"

Example of **fine-tuning** task:

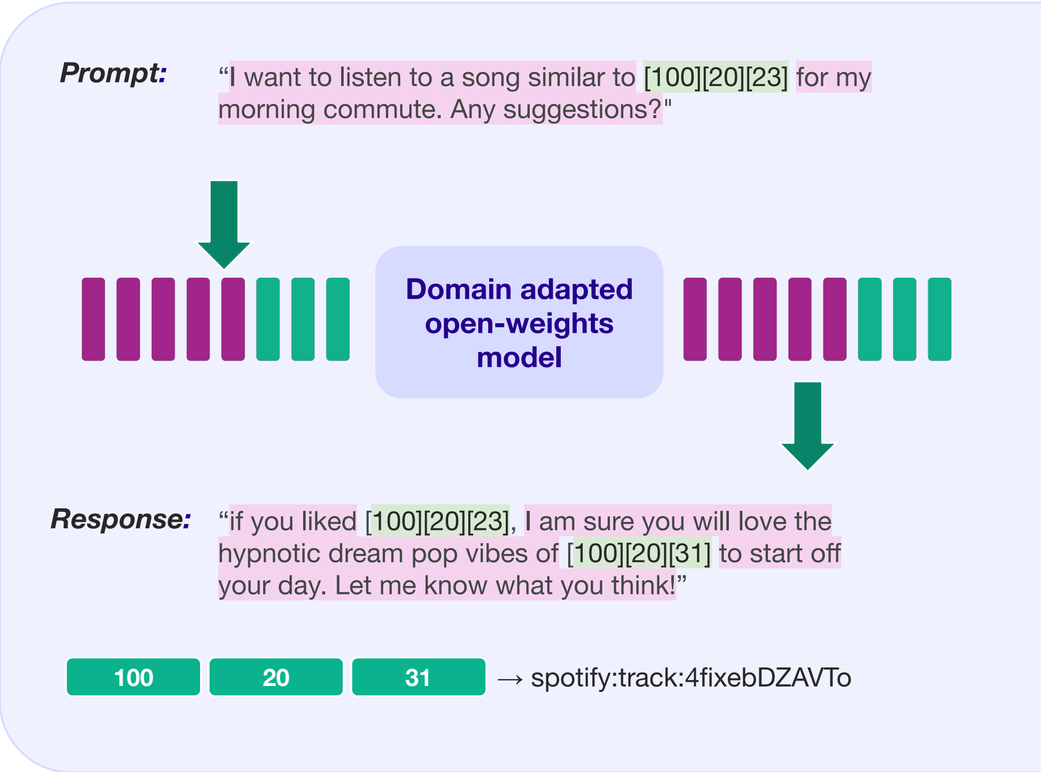
Semantic ID sequence:
[242, 42,..], **...** [3, 121,..], **?**

Target output:
[242, 42,..], **...** [3, 121,..], **[2, 11,..]**

How our domain-adapted model generates recommendations

- Receives a prompt mixing natural language and Semantic IDs.
- Predicts next most likely tokens, including Semantic IDs for catalog items.
- Outputs may contain Semantic IDs, optionally mixed with natural language.

Language handles reasoning; Semantic IDs provide grounding in Spotify's catalog.

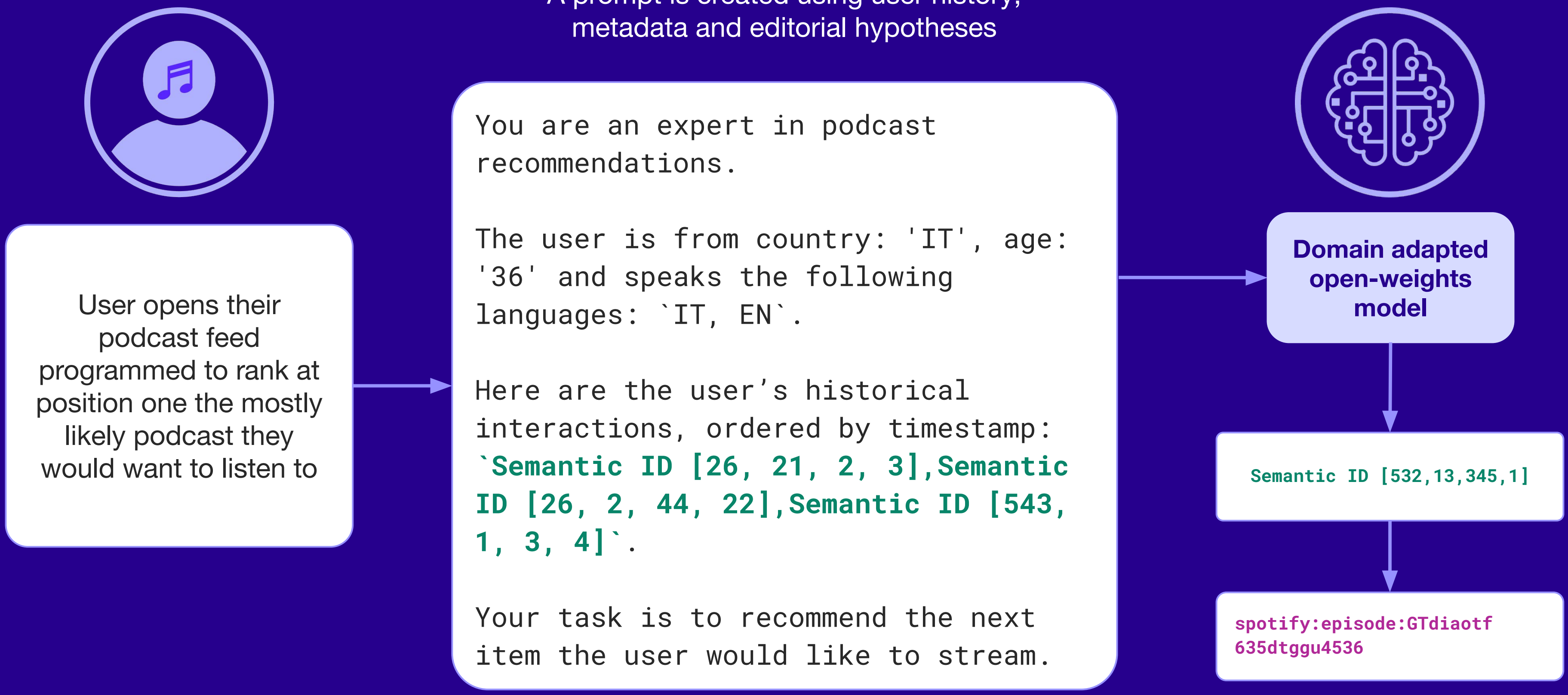


Prompt: "I want to listen to a song similar to [100][20][23] for my morning commute. Any suggestions?"

Response: "if you liked [100][20][23], I am sure you will love the hypnotic dream pop vibes of [100][20][31] to start off your day. Let me know what you think!"

100 20 31 → spotify:track:4fixebDZAVTo

Example for podcast recommendations



User opens their podcast feed programmed to rank at position one the mostly likely podcast they would want to listen to

A prompt is created using user history, metadata and editorial hypotheses

```
You are an expert in podcast
recommendations.

The user is from country: 'IT', age:
'36' and speaks the following
languages: `IT, EN`.

Here are the user’s historical
interactions, ordered by timestamp:
`Semantic ID [26, 21, 2, 3],Semantic
ID [26, 2, 44, 22],Semantic ID [543,
1, 3, 4]`.

Your task is to recommend the next
item the user would like to stream.
```

Our model generates the next item as a Semantic ID, which is mapped to a podcast URI

Domain adapted open-weights model

`Semantic ID [532,13,345,1]`

`spotify:episode:GTdiaotf635dtggu4536`

Key results enabled by Semantic IDs

+8.8%

new-show discovery

+3.5%

re-engagement

(latest AB test results)

+173% BLEU

for Artist → Playlist Title generation

+108%, +32% and +7%

Hit Rate for Artist, Episode and Audiobook Search

+16% Hit Rate

for Episode Discovery

Better natural language behaviour:

- Stronger performance on broad-intent queries
- Clearer, more fluent explanations for recommendations

+22%

Gain from multi-task training

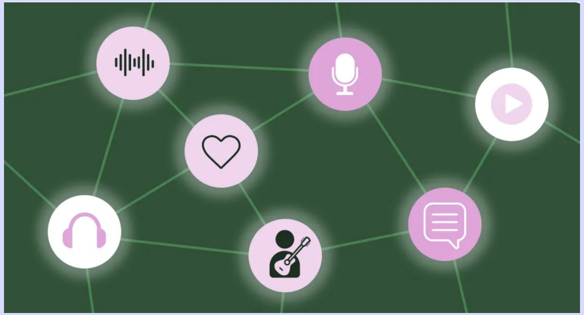
Product Foundations for AI-Driven Personalization

Adapted open-weight models

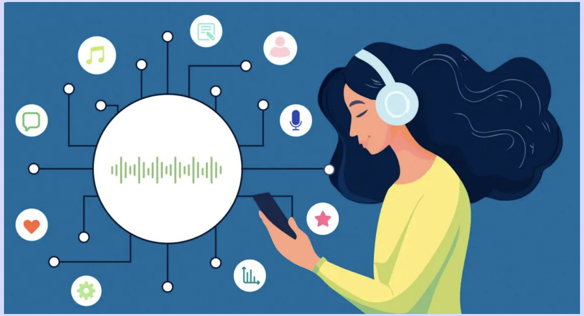
- We build on open-weight LLMs as our flexible, world-knowledge foundation.
- We adapt them to Spotify's catalog and behavior patterns by teaching them to "speak Spotify" through Semantic IDs.

Giving back to the community

- Even as we leverage open-weight models, we share our recipes, methods, and results with practitioners and researchers, contributing back to the broader ecosystem.



Teaching Large Language Models to Speak Spotify: How Semantic IDs Enable Personalization



Contextualized Recommendations Through Personalized Narratives using LLMs | Spotify Research

R&D

Looking Ahead

Multi-angle open-source AI

Open-source AI at Spotify

01

Open-source engineering platforms

Backstage provides an openly shared, structured knowledge layer that powers AI assistance and future AI agents for safe, efficient engineering.

02

Adapted open-weight models for product innovation

We build on open-weight LLMs and adapt them to understand Spotify's catalog and behavior, grounding them using Semantic IDs, domain signals, and adaptation methods.

03

Sharing back with the ecosystem

As we build with open ecosystems, we contribute back: publishing research, open-sourcing methods, sharing findings, and collaborating with industry and academia.

04

A multi-angle approach to open-source AI

Open platforms power AI for development, adapted models power AI for product experiences, together enabling AI that understands our systems and our catalog.

R&D

Thank You

backstage.spotify.com

spotify.github.io

research.atspotify.com