Mathematics Behind Recommender Systems

Academia meets industry

Pavel Kordík

Recombee & FIT CTU Prague

MFF UK Seminar on Applications of Mathematics
April 1st, 2026

Introduction

Recommendations are everywhere

Every time you see:

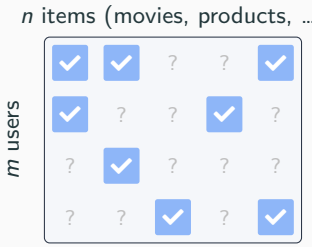
- *"You might also like..."* on Netflix
- *"Customers also bought..."* on Amazon
- *"Discover Weekly"* on Spotify
- *"For You"* in a news feed

... there is a recommender system behind it.

The core challenge:

Millions of items, millions of users.

How to find the *right* item for *each* user?



Interaction matrix: most entries are **unknown**

Recommendation and search are converging

Both solve the **same core problem**: produce a ranked list of items.

Search	Recommendation
User provides a **query**	System infers intent from **user profile**
$\rightarrow$ ranked relevant documents	$\rightarrow$ ranked personalized items
Google, Elasticsearch, e-shop search	Netflix, Spotify, news feeds

The convergence:

- Both optimize **ranking metrics** (not classification)
- Modern search is personalized; modern RecSys supports queries
- Shared infrastructure: embeddings, ANN indices, learning-to-rank

Recombee

Recommendation & Search as a Service

- Founded in 2015 with my PhD students
- Bootstrapped to $6M ARR
- 500+ customers in 40+ countries
- Billions of interactions processed daily
- Personalized recommendation, semantic search, real-time analytics

Customers: The Telegraph, DAZN, Pepper, Prima, Český rozhlas, Mediavine, Neura, 9GAG, and many more across media, e-commerce, video, and marketplaces.

prg.ai Minor students at Recombee HQ, Václavské náměstí 1, Prague

Recombee Lab @ FIT CTU

Joint university-industry research lab at the Dept. of Applied Mathematics, FIT CTU in Prague.

Focus:

- Linear autoencoders & matrix factorization
- Bandit algorithms & exploration
- Evaluation methodology
- NLP for recommendations

Collaborations: Ladislav Peška group at MFF UK, TU Kaiserslautern, SMU Singapore

🔗 `kam.fit.cvut.cz/recombee-lab`

Today's program

1. Linear autoencoders

From EASE to ELSA, VASP, and Sparse ELSA

2. Matrix factorization

SVD, ALS, and describing taste with numbers

3. Bandit algorithms

Exploration vs. exploitation

4. Sequential & language models

ReALM and beeFormer

5. Evaluation & production

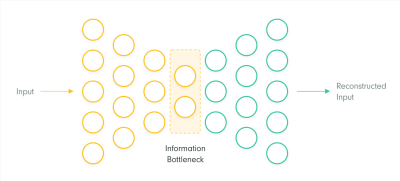
Ranking metrics, offline bias, deployment

6. Research overview & RecSys 2025

Linear Autoencoders

What is an autoencoder?

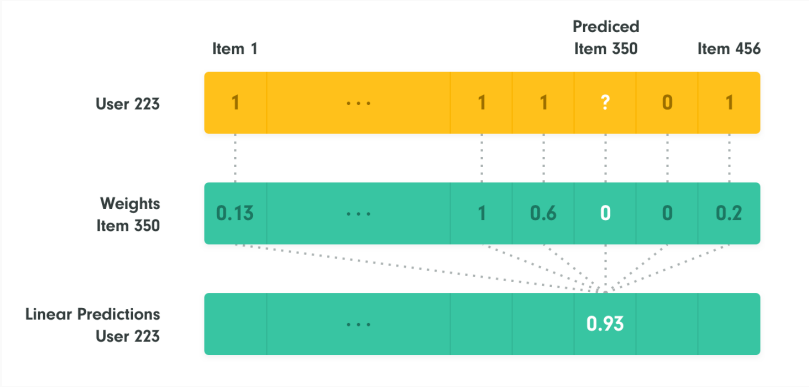
An **autoencoder** learns to compress and reconstruct data through an information bottleneck:



- Input = what a user has consumed (n-dimensional binary vector)
- Bottleneck = compressed "taste profile"
- Output = predicted scores for *all* items
- A linear autoencoder uses no activation functions – just matrix multiplication

recombee.com/blog – Linear Methods and Autoencoders

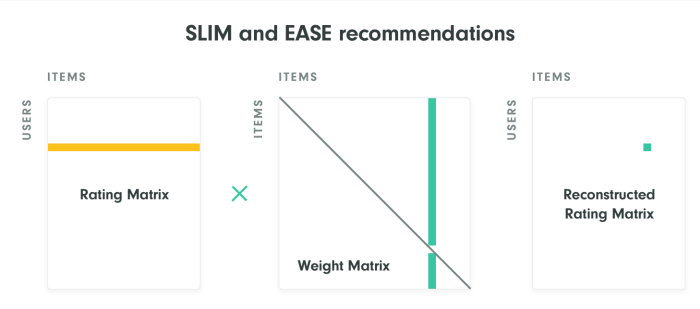
Linear prediction: How it works



recombee.com/blog – Linear Methods and Autoencoders

SLIM and EASE: The shallow approach

No hidden layers – just an $n \times n$ weight matrix $\mathbf{W}$:



Rating Matrix $\times$ Weight Matrix $=$ reconstructed matrix. The diagonal is zeroed out, so an item cannot predict itself.

recombee.com/blog – Linear Methods and Autoencoders

EASE: Embarrassingly Shallow Autoencoders

Optimization: Find $\mathbf{W}$ that best reconstructs the data:

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \|\mathbf{X} - \mathbf{X}\mathbf{W}\|_F^2 + \lambda\|\mathbf{W}\|_F^2, \quad \text{diag}(\mathbf{W}) = \mathbf{0}$$

Closed-form solution (one step!)

$$\mathbf{W}^* = \mathbf{I} - \hat{\mathbf{P}} \cdot \text{diag}(\hat{\mathbf{P}})^{-1}, \qquad \hat{\mathbf{P}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}$$

- Closed-form solution from one matrix inverse
- Negative weights capture dissimilarity
- **Problem:** Inverting an $n \times n$ matrix costs $O(n^3)$

Steck, EASE, WWW 2019

The scalability wall

How bad is $O(n^3)$ in practice?

Some Recombee clients have **hundreds of millions of items** in their catalogs.

Catalog size n	Matrix W	Inverse time (approx.)
10,000	$10^4 \times 10^4$	seconds
100,000	$10^5 \times 10^5$	minutes
1,000,000	$10^6 \times 10^6$	days
10,000,000	$10^7 \times 10^7$	impossible (4 PB RAM)
100,000,000	$10^8 \times 10^8$	utterly impossible

Two approaches to break the wall:

1. **ELSA** (Recombee / FIT CTU): Low-rank factorization $\mathbf{W} = \mathbf{A}\mathbf{A}^\top - \mathbf{I} \to O(nd^2)$
2. **SANSA** (MFF UK + GLAMI): Sparse approximate inverse of the Gram matrix $\to$ $O(n \cdot \text{nnz})$

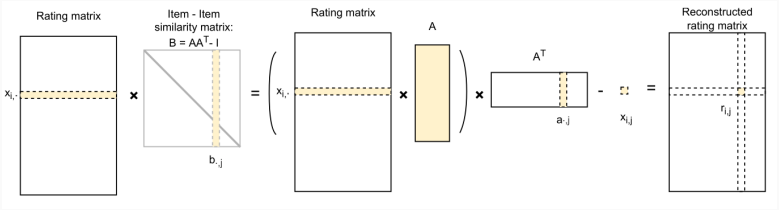
ELSA: Making it scalable

ELSA was developed jointly at **FIT CTU + Recombee**. **Key idea:** Instead of storing the full $n \times n$ matrix $\mathbf{W}$, represent each item as a d-dimensional embedding vector ($d \ll n$):

Each item i gets an embedding $\mathbf{a}_i \in \mathbb{R}^d$. Stacking all items: $\mathbf{A} \in \mathbb{R}^{n \times d}$.

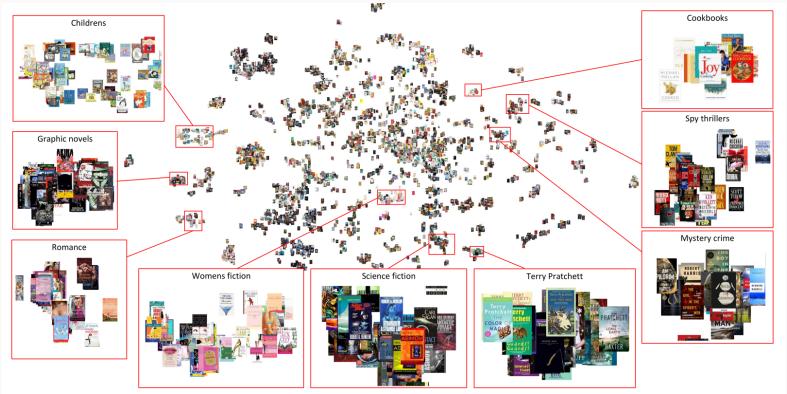
After row normalization, item similarity becomes cosine similarity:

$$W_{ij} = \frac{\mathbf{a}_i^\top \mathbf{a}_j}{\|\mathbf{a}_i\| \, \|\mathbf{a}_j\|} - \delta_{ij}$$



Vančura et al., ELSA, RecSys 2022

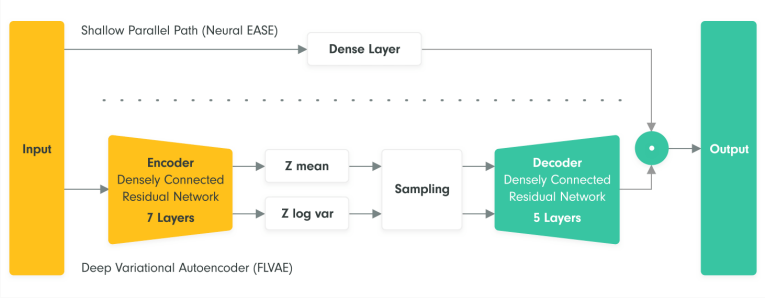
ELSA: What the embeddings look like



Vančura et al., ELSA, RecSys 2022 – Goodreads embeddings

VASP: Combining linear and deep paths

VASP combines the robustness of a linear model with the expressiveness of a deep generative model.

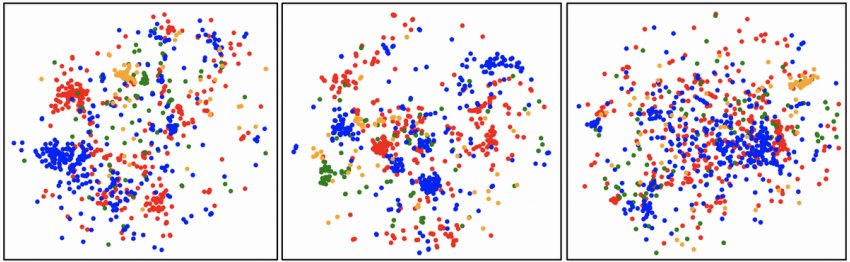


Linear and deep paths are combined via Hadamard $\odot$, so both must agree.

Vančura, Kordík, VASP, RecSys 2021

VASP: What each path captures

t-SNE of MovieLens embeddings, colored by genre:



Left: VASP output. **Center:** EASE (linear) – smooth patterns. **Right:** FLVAE (deep) – complex clusters. Colors: Horror, Children's, Westerns, Noir.

Vančura, Kordík, VASP, RecSys 2021

Sparse ELSA: Interpretable embeddings from interactions

Idea: Gradually prune ELSA embeddings during training:

$$\mathbf{A}_s^{(t)} = \mathcal{S}_{k_t}(\mathbf{A}^{(t)}), \qquad k_0 = d \xrightarrow{\text{exp. decay}} k_T = k$$

- $\mathcal{S}_{k_t}$ keeps only the k_t strongest coordinates and zeros the rest
- Training starts dense ($k_0 = d$) and gradually ends with only k active dimensions

Results (nDCG@100):

	Dense	**10× compr.**
Goodbooks	0.489	**0.491**
Netflix	0.395	0.393

At 10× compression, the loss is very small.

Sparse factors naturally correspond to semantic segments – without any metadata:

- Children's classics
- Detective fiction
- Sci-fi romance

Pure interaction data → interpretable structure.

Matrix Factorization

Intuition: The latent space of taste

Imagine: Each movie can be described by a few numbers – how much action, romance, humor...

And each viewer by the *same* numbers – how much they enjoy each.

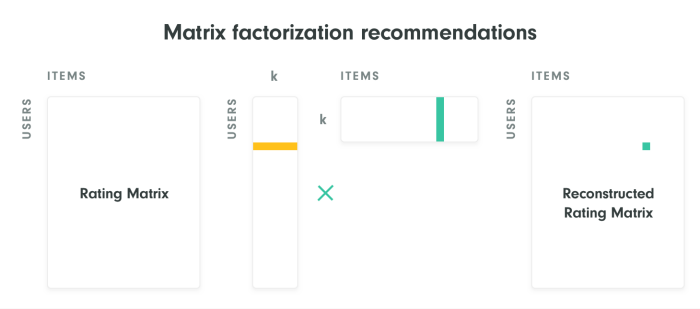
Matrix Factorization

$$\underbrace{\mathbf{X}}_{m \times n} \approx \underbrace{\mathbf{U}}_{m \times r} \cdot \underbrace{\mathbf{V}^\top}_{r \times n}$$

- $\mathbf{U}$ – user "profiles" in r-dimensional latent space
- $\mathbf{V}$ – item "profiles" in the same space
- $r \ll \min(m, n)$ – typically 50–200 dimensions

Prediction: $\hat{X}_{ij} = \mathbf{u}_i^\top \mathbf{v}_j$ (dot product = "compatibility")

Matrix factorization: Visual



Rating Matrix $\approx$ Users $(m \times k)$ $\times$ Items $(k \times n)$. Low-rank factors capture latent preferences and item characteristics.

recombee.com/blog – Linear Methods and Autoencoders

Finding the factorization: ALS

Alternating Least Squares – an elegant iterative approach:

1. **Fix items $\mathbf{V}$**, optimize users $\mathbf{U}$:
   $\mathbf{U}^* = (\mathbf{V}^\top \mathbf{V} + \lambda \mathbf{I})^{-1} \mathbf{V}^\top \mathbf{X}^\top$ (just linear regression!)
2. **Fix users $\mathbf{U}$**, optimize items $\mathbf{V}$:
   $\mathbf{V}^* = (\mathbf{U}^\top \mathbf{U} + \lambda \mathbf{I})^{-1} \mathbf{U}^\top \mathbf{X}$ (same, roles swapped)
3. Repeat until convergence.

Why ALS?

- Each step has a closed-form solution (no gradient descent)
- Parallelizable across rows / columns
- Works for implicit feedback (clicks instead of ratings)

Bandit Algorithms

The exploration–exploitation dilemma

Imagine a restaurant:

- **Exploitation:** Order your favorite steak (certainty)
- **Exploration:** Try a new Thai dish (might be great, might not)

Exploitation

Recommend what the model knows works: safe, but no new information.

Exploration

Try new / uncertain items: risky, but we learn.

Multi-armed bandit: A mathematical framework for decision-making under uncertainty. Named after a gambler facing a row of slot machines.

Thompson Sampling

Idea: For each item, maintain a probability distribution over its quality.

Algorithm

1. For each item i: sample $\theta_i \sim \text{Beta}(\alpha_i, \beta_i)$
   where $\alpha_i = \text{successes} + 1, \quad \beta_i = \text{failures} + 1$
2. Recommend the item with the highest sample: $i^* = \arg\max_i \theta_i$
3. Observe outcome, update α_{i^*} or β_{i^*}

Why it works:

- Uncertain items $\rightarrow$ wide distribution $\rightarrow$ sometimes high sample $\rightarrow$ exploration!
- Reliable items $\rightarrow$ narrow distribution around high value $\rightarrow$ exploitation!
- Natural balance without parameter tuning

BMAB: What if popularity changes over time?

Problem: Standard bandits assume quality is *constant*. In practice: viral videos, breaking news, seasonal trends...

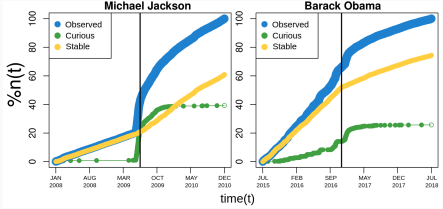
Two types of audience

1. **Loyal:** Stable interest (fans)
2. **Curious:** Burst activity (viral wave)

Model: Mixture of two Poisson processes

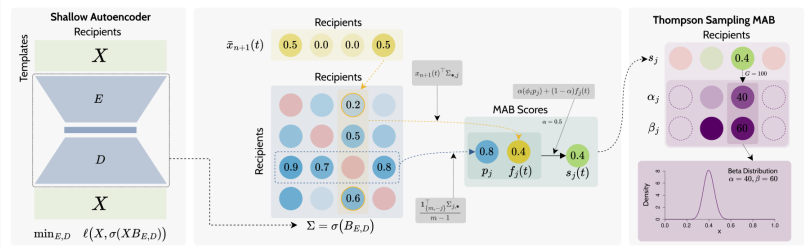
$$N(t) \sim \text{Poi}(\lambda_{\text{loyal}}) + \text{Poi}(\lambda_{\text{curious}}(t))$$

Burst detected $\rightarrow$ reset priors $\rightarrow$ rapid adaptation.



Stable + bursty audiences in real traffic.

Active Recommendation for Email Outreach



We do this to reduce spam: send fewer emails, keep open rate high, and still learn which recipients respond to new templates. Shallow autoencoders score cold-start templates and Thompson Sampling adaptively chooses recipient batches in a 15M-interaction outreach platform.

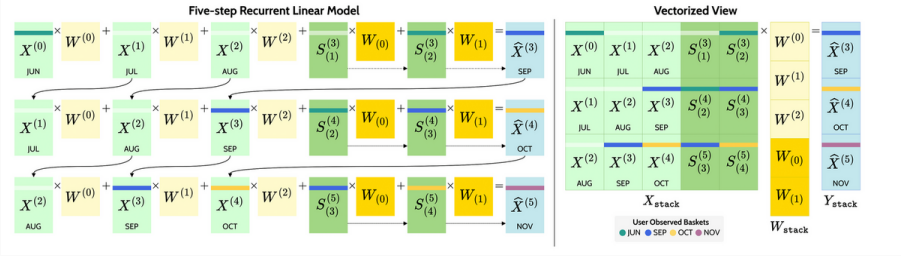
Zid, Alves, Kordík — CIKM 2025

Sequential & Language Models

ReALM: Predicting the next shopping basket

Rows = users, columns = visits. User 1 always buys fruit; diapers grow S→M→L.

ReALM: The model



- Autoregressive linear model
- Captures temporal dependencies
- Closed-form analytical solution – no SGD

Where LSTM/Transformers train for hours, ReALM converges in **seconds**.

Zmeškalová et al., ReALM, RecSys 2025

The cold-start problem

What if we know nothing about an item?

A new movie in the catalog – nobody has seen it. Collaborative filtering fails.

Solution: We have *text descriptions*! Title, genre, synopsis...

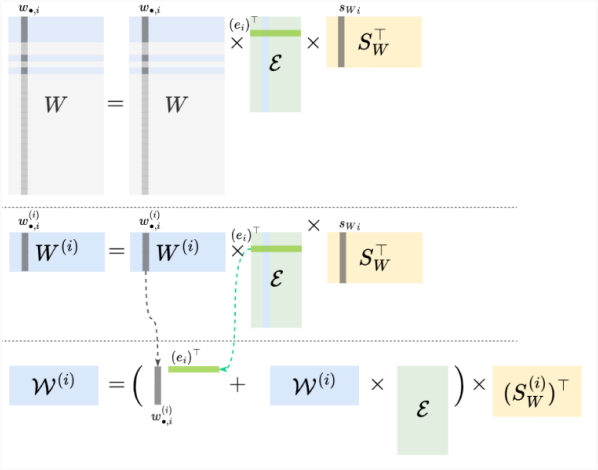
Sentence Transformers

Models that convert text into numerical vectors (embeddings).
Similar texts $\rightarrow$ similar vectors.

But: Semantic similarity $\neq$ recommendation similarity.
"Star Wars" and "Documentary about George Lucas" are semantically close, but fans of one may not want the other.

IndAE: Autoencoders for cold-start items

Problem: EASE/ELSA cannot recommend new items with zero interactions. We still want to score a brand-new item from its metadata before anyone clicks on it.



beeFormer: Architecture

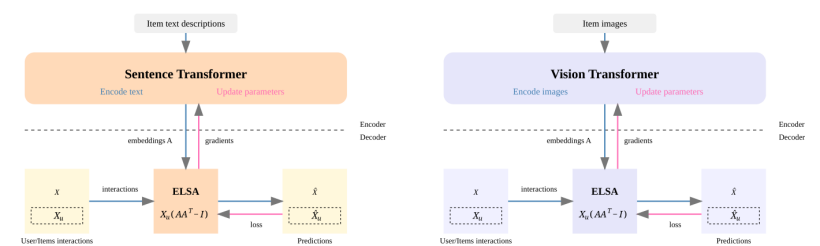


Fig. 1. BeeFormer is a Transformer-based Architecture for Vision-Language Tasks. It can directly use raw text, process it with a Vision Encoder, and handle multiple modal and textual inputs through either a cross-modal or text sequence processor.

Transformer encodes item text/images into embeddings $\mathbf{A}$. ELSA loss $\mathcal{L} = \|\mathbf{X} - \mathbf{X}(\mathbf{A}\mathbf{A}^\top - \mathbf{I})\|_F^2$ measures reconstruction quality. Gradients flow back into the Transformer – it learns to predict *user behavior*, not just semantic similarity.

beeFormer: Results

Zero-shot transfer

Trained on MovieLens + Amazon, tested on Goodbooks-10K:

Model	Recall@20
Semantic ST	0.1146
beeFormer	**0.2649**

+131% improvement

Cold-start (ML-20M)

New items with no interactions:

Model	Recall@20
Heater	0.3114
beeFormer	**0.4630**

+49% improvement

Multi-domain training yields +8% – knowledge accumulates across domains.

SHIELD: Making semantic search safe

Problem: Semantic search understands meaning, so harmful intent is harder to catch than with keywords alone.

SHIELD: 3-stage framework

1. **Generate** realistic harmful/sensitive queries with LLMs
2. **Filter** high-quality examples with a reward model
3. **Classify** queries in the moderation pipeline

Results:

Method	Accuracy
BM25 keyword	93.2%
Semantic + FAISS	96.5%
MoralBERT	**98.4%**

Open source:

`flpspacek/SHIELD`

`spacefi1/moralBERT`

Applicable beyond marketplaces too.

Spaček, Vančura, Kordík — UMAP 2025

Evaluation & Production

How do we evaluate a ranked list?

Unlike classification (accuracy) or regression (RMSE), ranking cares about order and position.

Key metrics

- **Precision@K:** Fraction of top-K items that are relevant
- **Recall@K:** Fraction of all relevant items found in top-K
- **NDCG@K** (Normalized Discounted Cumulative Gain):
  Rewards relevant items appearing *higher* in the list

NDCG:

$$\text{DCG@K} = \sum_{i=1}^{K} \frac{\text{rel}_i}{\log_2(i+1)}, \qquad \text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}} \in [0,1]$$

Position 1 contributes $\frac{1}{\log_2 2} = 1.0$; position 10 only $\frac{1}{\log_2 11} = 0.29$.

The problem with offline evaluation

In theory: Compute NDCG on held-out data, pick the best model, deploy.

In practice: This often fails. Why?

The core bias: offline data reflects the *old* system

We only observe interactions with items the **previous recommender** chose to show. A new model that recommends *different* items looks bad offline – those items were never shown, so there is no positive signal for them.

This creates a systematic bias toward exploitative algorithms:

- Models that mimic the old system score well offline
- Models that **explore** new items score poorly – even if users would love them
- Popular items dominate the test set $\rightarrow$ popularity-chasing wins offline

Our contribution: Better offline–online correlation

Solution: Leave-*Last*-One-Out + popularity penalization

$$\text{recall}^{\beta}_{\text{LLOO}} \quad \text{with weights } p(i)^{-\beta}, \quad \beta \approx 0.30$$

Method	MSR
Standard LOO-CV	12.9%
LLOO + β	**34.3%**

Leaving out the *last* interaction respects chronology, so evaluation reflects what the user did next rather than a random held-out event. Popularity penalization reduces the bias toward models that only repeat what the old system already showed.

Kasalický, Alves, Kordík — evalRS @ KDD 2023

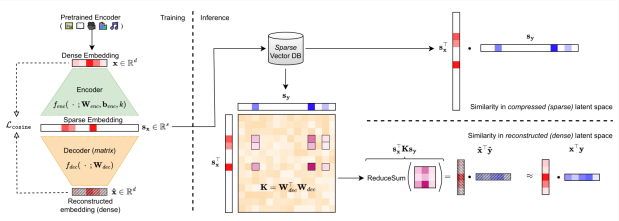
RepSys: Interactive evaluation framework

Open-source app for interactive comparison of recommendation models.

`cowjen01/repsys`

CompresSAE: How it works

We use CompresSAE because modern embedding models are accurate but too memory-hungry at production scale.

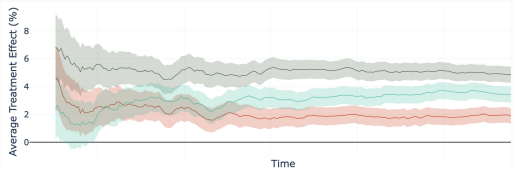


- Training learns a sparse code that reconstructs the dense embedding
- Inference compares sparse codes directly or via a kernel

Kasalický et al., CompresSAE, RecSys 2025

CompresSAE: Online A/B test results

Problem: 100M items $\times$ 768-dim = 307 GB memory for embeddings.



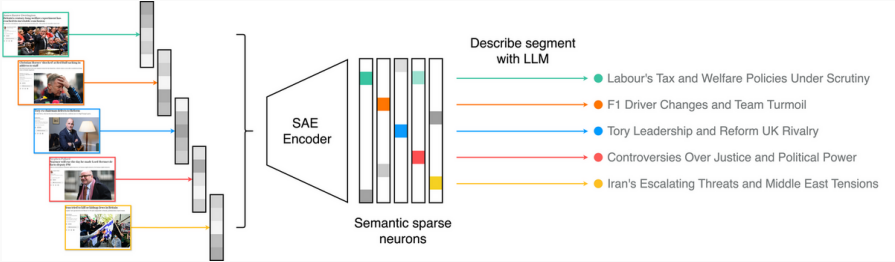
	CTR lift	Size
Dense 768	+4.86%	307 GB
CompresSAE	**+3.44%**	**26 GB**

12$\times$ compression, 1.35% CTR loss.

15 seconds on H100.

Kasalický et al., CompresSAE, RecSys 2025

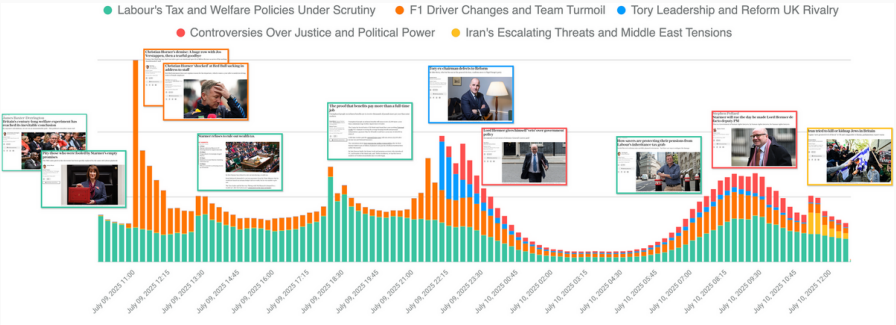
The Telegraph: Segment-aware editorial analytics



SAE encoder maps articles to **semantic sparse neurons**. An LLM describes each segment. Editors see which reader segments engage with which topics.

The Telegraph case study, INRA @ RecSys 2025

The Telegraph: Segment dynamics in real time



Content consumption by segment over time. Labour/Tax, F1 Racing, Tory Politics, Justice, Middle East.

Research Overview & RecSys 2025

Beyond accuracy: Fairness, diversity, serendipity

Fairness

Minimum Exposure Guarantees
(Lopes, Alves et al., Expert Syst. 2024)
ILP post-processing ensuring each item group receives fair exposure.

SAGEA (RecSys 2025 poster)
Fairness-preserving group recommendations via sparse autoencoder aggregation.

Diversity & more

Ontology-based diversity
(Kuznetsov, Kordík, 2023)
Improving cold-start diversity.

Conv4Rec: Serendipitous recommendations.

Regionalization (ACM TSAS 2024)
Spatially-constrained collaborative filtering.

Publications – core methods

Linear autoencoders:

- VASP *RecSys 2021*
- ELSA *RecSys 2022*
- Eval. CFAE @ Scale *ACM TORS 2025*
- Sparse ELSA *WWW 2026*

Factorization & theory:

- Uncertainty-Adj. MF *TNNLS 2023*
- Generalization bounds *ICML 2024*
- Conv4Rec *TNNLS 2025*
- Probabilistic modeling *RecSys 2025*

Sequential & language:

- beeFormer *RecSys 2024*
- CompresSAE *RecSys 2025*
- ReALM *RecSys 2025*
- LLM alignment *ACM TIST 2024*

Publications – applications and evaluation

Bandits, NLP & sequential:

- BMAB *RecSys 2021*
- BPoP *WWW 2024*
- SHIELD (search safety) *UMAP 2025*
- Active Rec. (email) *CIKM 2025*

Applications & evaluation:

- IndAE (cold-start AE) *RecSys 2024*
- Offline/Online eval *KDD 2023*
- Telegraph *RecSys 2025*
- Fairness, diversity, ...

25+ publications at RecSys, WWW, ICML, KDD, TNNLS, TORS, TIST

Open-source projects

ELSA

`recombee/ELSA`

Scalable linear autoencoder, PyPI

beeFormer

`recombee/beeFormer`

Sentence Transformers for RecSys

CompresSAE

`recombee/CompresSAE`

Post-hoc embedding compression

RepSys

`cowjen01/repsys`

Interactive RecSys evaluation

`github.com/recombee`

Research Posters

See all →

recombee.com/research

RecSys 2025 in Prague

The 19th ACM Conference on Recommender Systems

22–26 September 2025, Prague

General Chairs:

M. Bielikova, P. Kordík, M. Schedl

1000+ attendees, around 60% from industry

Google, Amazon, Netflix, Spotify, ...

SlidesLive recordings

Recombee

Empowering Discovery Through Research-Driven Personalization

Recombee delivers real-time personalized recommendations and semantic search powered by the latest in AI research — from proprietary transformer models to reinforcement learning and LLMs.

Our RecSys 2025 Contributions

- The Future is Sparse: **Embedding Compression for Scalable Retrieval** in Recommender Systems
- **Probabilistic Modeling, Learnability and Uncertainty Estimation** for Interaction Prediction in Movie Rating Datasets
- **Recurrent Autoregressive Linear Model** for Next-Basket Recommendation (ReALM)
- SAGEA: **Sparse Autoencoder-based Group Embeddings Aggregation** for Fairness-Preserving Group Recommendations
- **Segment-Aware Analytics** for Real-Time Editorial Support in Media Groups: Lessons from The Telegraph

Who We Are

Prague-based AI company serving billions of recommendations monthly. Trusted by global brands including DAZN, The Telegraph, 9GAG, Audiomack, and Crexi.

Research & Innovation

Efficient Retrieval
Sparse embeddings reduce memory cost while preserving quality.

Uncertainty & Learnability
New provable results and abstention strategies.

Next-Basket Recommendation
ReALM combines interpretability & speed.

Fair Group Recommendations
SAGEA balances fairness and accuracy.

Editorial Analytics
Real-time LLM-based pipelines at The Telegraph.

Why Recombee

- **Real-Time & Scalable** (30k+ recs/s, <200 ms latency)
- **Flexible & Customizable** (ReQL filters, boosters, business logic)
- **Research-Driven** (Beeformer, reinforcement learning, LLM-based semantic search)
- **GDPR-Compliant & Enterprise-Ready**

Explore Our Research

Try Recombee Free Trial

Explore More at recombee.com, Proud Sponsor of RecSys 2025.

Thank you!

research@recombee.com github.com/recombee
recombee.com/research

RECOMBEE RESEARCH TEAM

Rodrigo Alves, Ph.D.

Head of Research

Vojtěch Vančura, Ph.D.

ML Researcher

Petr Kasalický

ML Researcher

Dan Bohuněk

Applied Scientist

Jaroslav Hradil

Applied Scientist

Special thanks to Martin Spišák, former team member