Artificial Intelligence Applied to the Web

Chapter 4 Part 5 - Recommendation Systems IV

Diego Cornejo, Felipe Hernández and Juan Velásquez

University of Chile
Departament of Industrial Engineering

Spring 2025

Outline

Hybrid Recommendation Approaches

Introduction

- The three main approaches (Collaborative, Content-Based, Knowledge-Based) all have varying degrees of success depending on the domain.
- Each has its own strengths and weaknesses, such as:
  - Data sparsity and cold-start problems.
  - Significant effort required for knowledge acquisition.
- A **hybrid system** combines the strengths of different algorithms to overcome these shortcomings and improve recommendations.

Hybrid Recommendation Approaches

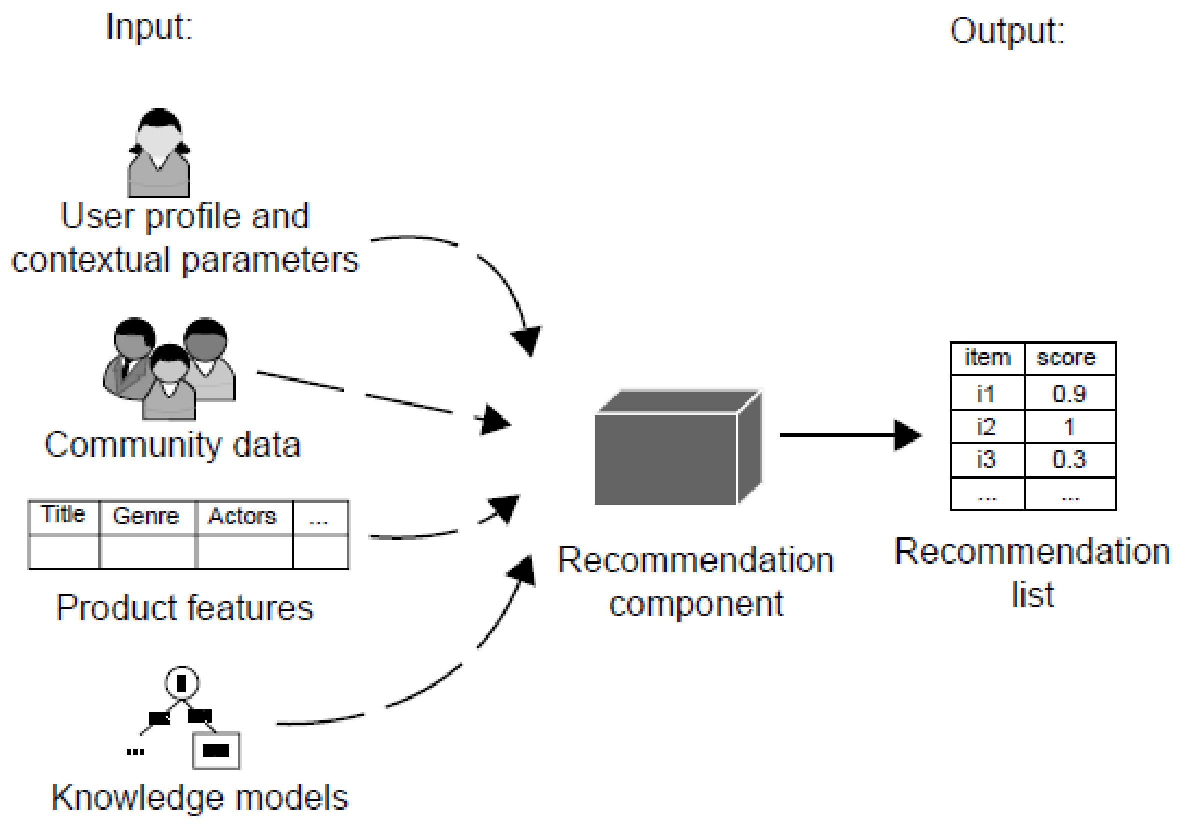
The Recommender "Black Box"

- A recommender system can be seen as a black box that transforms various inputs into a ranked list of recommended items.
- Potential inputs include:
  - User profiles and contextual parameters.
  - Community data (ratings).
  - Product features (content).
  - Knowledge models (rules, constraints).
- **None of the basic approaches can fully exploit all of these potential inputs on their own.**

INGENIERÍA INDUSTRIAL
UNIVERSIDAD DE CHILE

Hybrid Recommendation Approaches

The Recommender “Black Box”



Recommendation Paradigms

The Formal Problem

- A recommendation problem can be formalized as a utility function $\text{rec}(u, i)$, that predicts the usefulness of an item i for a specific user u.
- The system's task is to identify the top n items from a catalog that achieve the highest utility scores for that user.
- How this utility score is calculated depends on the underlying recommendation paradigm.

Recommendation Paradigms

The Three Base Paradigms

- **Collaborative Filtering**:
  - Assumes users with similar past behaviors will have similar future preferences.
  - *Tell me what's popular among my peers.*
- **Content-Based**:
  - Follows a "more of the same" approach.
  - *Show me more of the same what I've liked.*
- **Knowledge-Based**:
  - Uses an additional source of explicit personalization knowledge.
  - *Tell me what fits based on my need.*

Recommendation Paradigms

Input Data Requirements

- The choice of paradigm determines the type of input data required.
- All paradigms require access to a **user model**, which can contain:
  - Items the user has rated.
  - Demographic information.
  - Contextual parameters (time of day, location, companions).
- Not all hybridization variants are possible in every application, as parts of the user model may not be available.

Recommendation Paradigms

Data Needs at a Glance

- Each paradigm selectively requires different data sources beyond the user profile:

Paradigm	User profile and contextual parameters	Community Data	Product Features	Knowledge Models
Collaborative	Yes	Yes	No	No
Content-Based	Yes	No	Yes	No
Knowledge-Based	Yes	No	Yes	Yes

Hybridization Designs

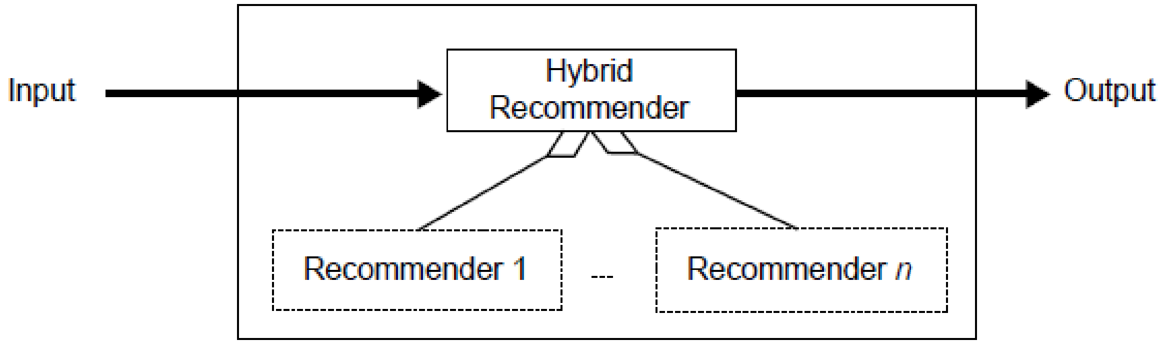
How to Combine Algorithms

- The second key dimension of a hybrid system is its design—the method used to combine two or more algorithms.
- There are three base designs for combining recommendation components:
  - Monolithic Hybridization
  - Parallelized Hybridization
  - Pipelined Hybridization

Hybridization Designs

Monolithic Design

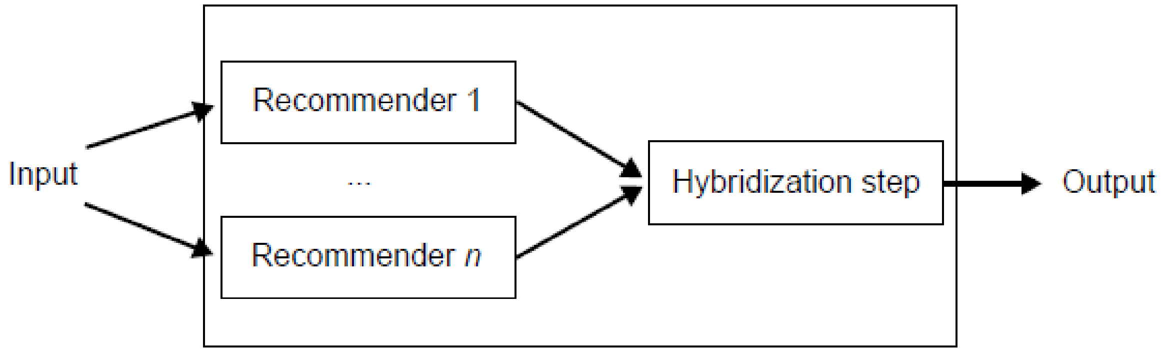
- A single recommender component that integrates multiple approaches by preprocessing and combining different knowledge sources *before* the main algorithm runs.
- The algorithm's behavior is modified to use different types of input data.



Hybridization Designs

Parallelized Design

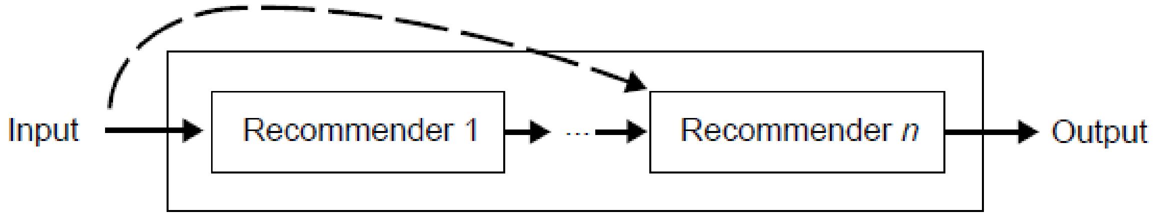
- Several different recommenders operate independently (in parallel).
- Each system produces its own list of recommendations, and a final hybridization step combines these separate lists into one.



Hybridization Designs

Pipelined Design

- Several recommenders are joined together in a sequence or "pipeline".
- The output of one recommender becomes the input for the next one in the sequence, which then refines the results.



Monolithic Hybridization Design

Monolithic Hybridization

- Unlike other designs that combine the *results* of separate recommenders, monolithic hybrids consist of a **single, integrated component**.
- This single component is modified to use multiple types of knowledge sources from the start.
- Hybridization is achieved through *preprocessing steps* that transform different types of input data into a unified representation that one main algorithm can use.
- The two main strategies in this category are:
  - **Feature Combination**
  - **Feature Augmentation**

Monolithic Hybridization Design

Feature Combination Hybrids

- It's a single recommender that uses a diverse range of input data types from the beginning.
- It combines features from different sources (e.g., collaborative and content-based) into a new, single set of hybrid features.
  - Example: Instead of just using a user's ratings (collaborative), the system might also look at the genre of the books they bought (content). It could then create a new feature like **"User likes many mystery books"**.
- This allows the system to find more nuanced similarities between users. For instance, two users who seem equally similar based on ratings might look very different once their preferred genres are considered.

Monolithic Hybridization Design

Feature Augmentation Hybrids

- A more complex monolithic design where the output of one recommendation approach is used to augment the input data for the main recommendation algorithm.
- The implementation of the contributing recommender is strongly interwoven with the main component.

Monolithic Hybridization Design

Feature Augmentation Hybrids

- Example: Content-Boosted Collaborative Filtering
  - *Step 1*: For every item a user **hasn't** rated, a content-based recommender first **predicts** a rating.
  - *Step 2:* The original user-item matrix is filled in with these predictions, creating a dense "pseudo-ratings" matrix.
  - *Step 3:* A collaborative filtering algorithm then runs on this new, complete matrix to make the final recommendations.
- This helps to overcome data sparsity, as the system can work with a full set of ratings, even if most are just predictions.

Parallelized Hybridization Design

Parallelized Hybridization

- Employ several recommenders side by side and aggregate their outputs.
- Some strategies are:
  - Mixed
  - Weighted
  - Switching

Parallelized Hybridization Design

Mixed Strategy

- A strategy that combines the results of different recommender systems at the user interface level.
- Instead of merging the scores into a single ranked list, the outputs from different techniques are presented together, often side-by-side.
- The recommendation result for user u and item i of a mixed hybrid strategy is the set uo-tuple $\langle \text{score}, k \rangle$ for each of its n constituting recommenders rec_k:

$$\text{rec}_{\text{mixed}}(u,i) = \bigcup_{k=1}^{n} \langle \text{rec}_k(u,i), k \rangle$$

Parallelized Hybridization Design

Mixed Strategy: Examples and Application

- *Example 1: Separate Lists*
  - A web page might show a list of “Top Rated Items” (from a collaborative filter) next to a list of “Similar Items” (from a content-based filter).
- *Example 2: Bundled Recommendations*
  - In tourism, one recommender might suggest hotels while another suggests activities. A mixed hybrid can bundle these into a consistent travel package (e.g., ensuring the hotel and activities are near each other).
  - This often requires a *conflict resolution* step (like a Constraint Solver) to ensure the bundled items are compatible.

Parallelized Hybridization Design

Weighted Strategy

- Combines recommendations from two or more systems by calculating a weighted sum of their individual scores.
- The final score for a user u and the item i is determined by:

$$\text{rec}_{\text{weighted}}(u, i) = \sum_{k=1}^{n} \beta_k \times \text{rec}_k(u, i)$$

- Where:
  - β_k is the associated weight of recommender rec_k.
  - All items scores must be on the same scale, and the sum of all weights must equal 1.

Parallelized Hybridization Design

Weighted Strategy: Examples and Application

- Assuming $\beta_1 = \beta_2 = 0.5$

Item	**rec$_1$ score**	**rec$_2$ rank**	**rec$_2$ score**	**rec$_2$ rank**	**rec$_w$ score**	**rec$_w$ rank**
Item1	0.5	1	0.8	2	0.65	1
Item2	0		0.9	1	0.45	2
Item3	0.3	2	0.4	3	0.35	3
Item4	0.1	3	0		0.05	
Item5	0		0		0	

Parallelized Hybridization Design

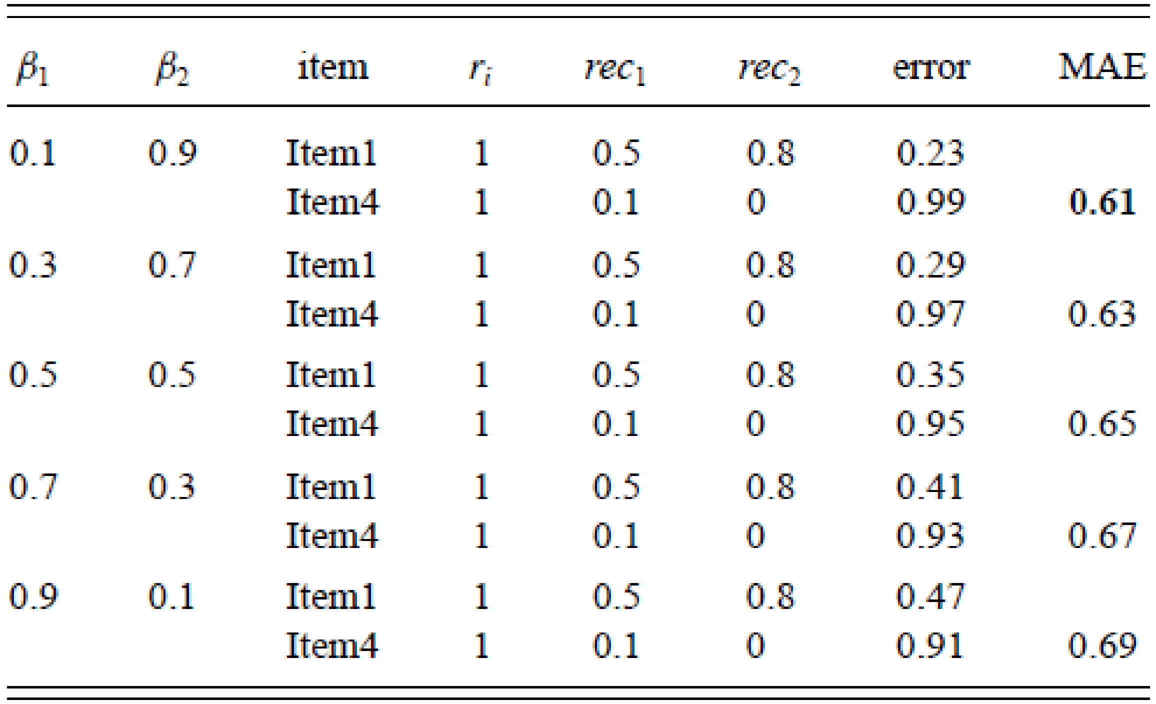
Dynamic Weighted Strategy

- A **static** weighting scheme is not always optimal, as different algorithms may perform better for different users or items.
- A **dynamic weighting** approach addresses this by adjusting the weights for each user.
- The goal is to find the weights that **minimize the prediction error** for that specific user.

$$\mathrm{MAE} = \frac{\sum_{r_i \in R} \sum_{k=1}^{n} \beta_k \times |\mathrm{rec}_k(u, i) - r_i|}{|R|}$$

Parallelized Hybridization Design

Dynamic Weighted Strategy



β_1	β_2	item	r_i	rec_1	rec_2	error	MAE
0.1	0.9	Item1	1	0.5	0.8	0.23	
		Item4	1	0.1	0	0.99	**0.61**
0.3	0.7	Item1	1	0.5	0.8	0.29	
		Item4	1	0.1	0	0.97	0.63
0.5	0.5	Item1	1	0.5	0.8	0.35	
		Item4	1	0.1	0	0.95	0.65
0.7	0.3	Item1	1	0.5	0.8	0.41	
		Item4	1	0.1	0	0.93	0.67
0.9	0.1	Item1	1	0.5	0.8	0.47	
		Item4	1	0.1	0	0.91	0.69

Parallelized Hybridization Design

Switching Strategy

- A hybrid design that uses an “oracle” or switching mechanism to decide which single recommender should be used in a specific situation.
- Instead of combining scores, the system selects **one** of its available algorithms based on a predefined **switching condition**.
- Only the output of the selected recommender is shown to the user.

$$\exists_1 k : 1 \ldots n \quad \mathrm{rec}_{\mathrm{switching}}(u, i) = \mathrm{rec}_k(u, i)$$

Switching Hybrids

Examples of Switching Conditions

- The switching condition can be based on various factors:
  - **Cold-Start Problem:** The system could use a Knowledge-Based recommender for new users and then "switch" to Collaborative Filtering once enough rating data is available.
  - **Optimizing Results:** A system might try different algorithms in a specific order. For example, first tries a content-based kNN; if that fails, it switches to collaborative filtering, and finally falls back to a long-term interest model.
  - **Handling Insufficient Results:** A system could use a primary algorithm and then switch secondary strategy if the primary one doesn't produce enough recommendations.

Pipelined Hybridization Design

Pipelined Hybridization

- Implement a staged process in which several techniques sequentially build on each other before the final one produces recommendations.
- Variants differentiate themselves mainly according to the type of output they produce.
  - Cascade hybrids
  - Meta-level hybrids

Pipelined Hybridization Design

Cascade Hybrids

- A hybrid design where recommenders are arranged in a sequence or "pipeline."
- Each recommender in the sequence **refines** the list of recommendations from the one before it.
- The output of one stage serves as the input for the next.
- Formally, we define it as:

$$\text{rec}_{\text{cascade}}(u,i) = \text{rec}_n(u,i)$$

where $\forall k \geq 2$:

$$\text{rec}_k(u,i) = \begin{cases} \text{rec}_k(u,i) & : \text{rec}_{k-1}(u,i) \neq 0 \\ 0 & : \text{else} \end{cases}$$

Pipelined Hybridization Design

Cascade Hybrids: The Key Rule

- The core principle is that a recommender in the sequence can only **re-rank or remove** items from its predecessor's list.
- **It cannot introduce new items.** If an item was filtered out in an early stage, it cannot be brought back in a later stage.
- This creates a multi-stage filtering and refining process.

Pipelined Hybridization Design

Cascade Hybrids: Advantages and Disadvantages

- They are a natural choice when one recommender produces a large, unsorted list of candidates.
  - Example: A Knowledge-Based system finds all products that meet a user's requirements, and a second, cascaded algorithm (like Collaborative Filtering) then ranks those products for personalization.
- The final recommendation list can sometimes be very small (or even empty) if each stage filters out too many items.
  - A common solution is to combine it with a **switching hybrid** as a fallback strategy.

Pipelined Hybridization Design

Meta-level Hybrids

- A pipelined design where one recommendation technique is used to build a model that serves as the input for the main, principal recommender.
- The first recommender doesn't output a list of items, but rather a learned **model** or set of features. The second recommender then uses this model to produce the final recommendations.

Pipelined Hybridization Design

Example: Collaboration via Content

- The **Fab** system is a classic example of this design.
- *Stage 1 (Content-Based Model):*
    - A content-based system first builds a user profile based on the content of items the user has liked (e.g., a vector of preferred keyword categories).
- *Stage 2 (Collaborative Recommendation):*
    - The main collaborative recommender then uses these **content-based profiles** to find users with similar tastes.
    - It recommends items that these similar peers have liked.
- This approach performs better than base techniques, especially when users have only a few rated items in common.

Pipelined Hybridization Design

Example: Collaborative Knowledge

- This variant combines Collaborative Filtering with a Knowledge-Based recommender.
- *Stage 1 (Collaborative Model):*
  - A collaborative filtering step retrieves **constraints** or rules from a user's peers.
  - Example rule: `"for_whom" = "gift" -> "brand" = "Montecristo"`.
- *Stage 2 (Knowledge-Based Recommendation):*
  - The main knowledge-based recommender applies this collection of peer-generated constraints to the product database to derive the final item recommendations.

Discussion and Summary

Key Takeaways

- **No "One-Size-Fits-All" Solution:** No single hybridization variant is universally best; the right choice depends on the specific application and problem.
- **Hybrids Are Better:** It is well-accepted that all base algorithms can be improved by being hybridized with other techniques.

Discussion and Summary

The Research Gap

- A major reason there's little research comparing different hybrid strategies is the **lack of appropriate datasets**.
- Most well-researched domains are collaborative movie recommendations or content-based news, making it hard to generalize findings for other areas.
- Therefore, no empirically backed conclusions about the definitive advantages of different hybridization variants can be drawn.

Artificial Intelligence Applied to the Web

Chapter 4 Part 5 - Recommendation Systems IV

Diego Cornejo, Felipe Hernández and Juan Velásquez

University of Chile
Departament of Industrial Engineering

Spring 2025