Personalized Machine Learning

Invariant Models

Rodrigo Alves

November 13, 2025

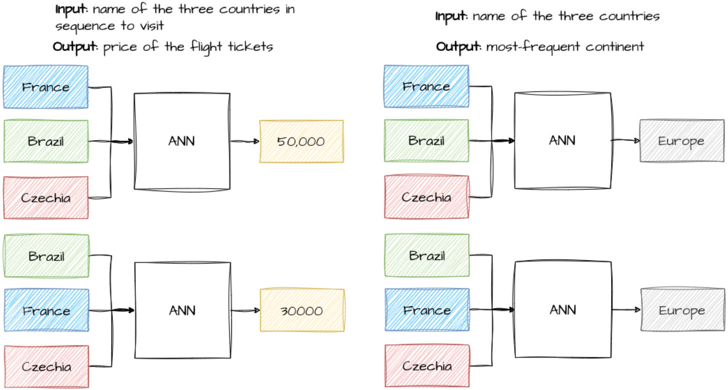
Bias in Recommender Systems

- Recommender systems are susceptible to biases, impacting fairness and accuracy.
- Types of bias: user, item, demographic, etc.
- Challenges: unfair treatment, limited diversity, stereotype reinforcement.

Position bias: it is a tendency to prioritize items in prominent positions, reinforcing popular items.

- Note that, however, often the recommender can be programmed to show some items in the first positions.
- Also, note that the models actually aim to **'bias'** the recommendation by putting the more relevant items in the first positions, and because of that, some PML algorithms focus on that.
- However, we can have some ML problems that are invariant by nature.

Invariant Models



Invariant Models

Domain invariance: For instance, Graph Neural Networks (GNNs) can be employed in Collaborative Filtering, where user-item interactions are modeled as a graph. GNNs are utilized to capture interactions, irrespective of their domain.

Time invariance: It is common to assume that the model is not influenced by time. For example, the order in which users rate the items is not considered in the evaluation of their taste.

Permutation-equivariant models: These models demonstrate equivariance concerning input permutations.

Odd-One-Out Problem

- The odd-one-out problem is a widely explored concept in Recommender Systems.
- Traditionally, it involves predicting user choices, such as identifying which item a user is most likely to click from a given list.
- Our lecture, however, focus to a more nuanced application within neuroscience, particularly in the context of triplets.
- **Triplets Problem:** Consider presenting three images to an individual: $\mathcal{I}_a, \mathcal{I}_b, \mathcal{I}_c$.
  - The task is to predict which pair of images exhibits the **closest conceptual similarity** based on given options.
  - Among goals we aim to develop models capable of **predicting similarity** within new triplets.
  - Simultaneously, we aim to **construct embeddings** that capture how humans perceive and understand conceptual relationships.

Triplets Problem





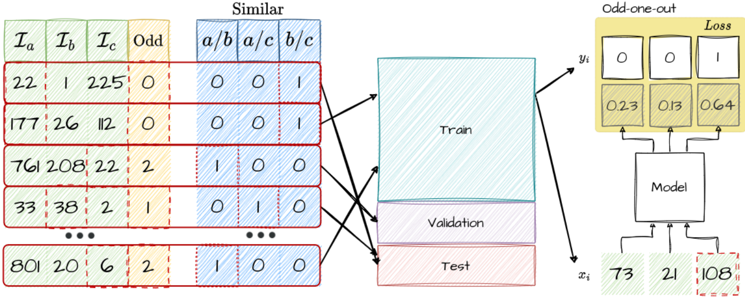

Triplets Problem: Data

I_a	I_b	I_c	Odd
22	1	225	0
177	26	112	0
761	208	22	2
33	38	2	1
…	…	…	…
801	20	6	2

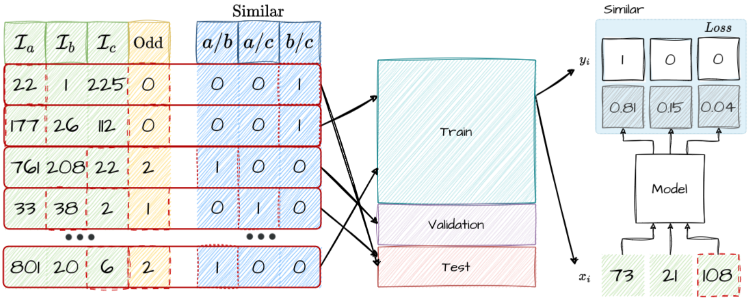
Similar

a/b	a/c	b/c
0	0	1
0	0	1
1	0	0
0	1	0
…	…	…
1	0	0

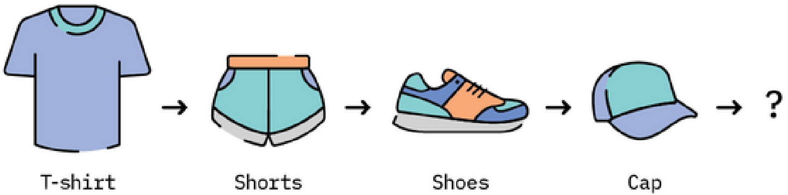
Triplets Problem: Odd-one-out



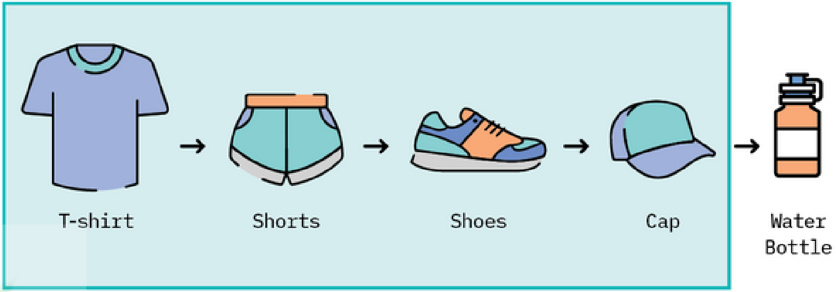
Triplets Problem: Similar



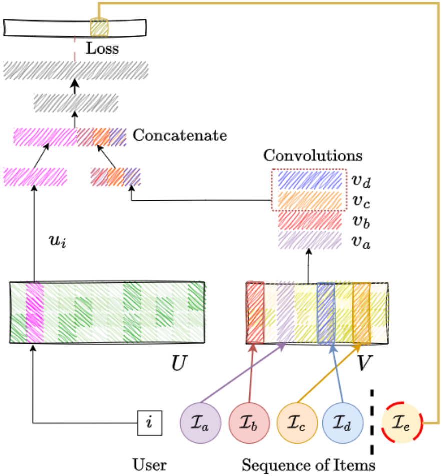
Sequential Recommendation (RECAP)



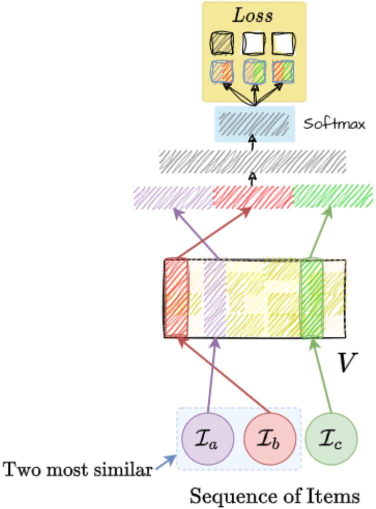
Sequential Recommendation (RECAP)



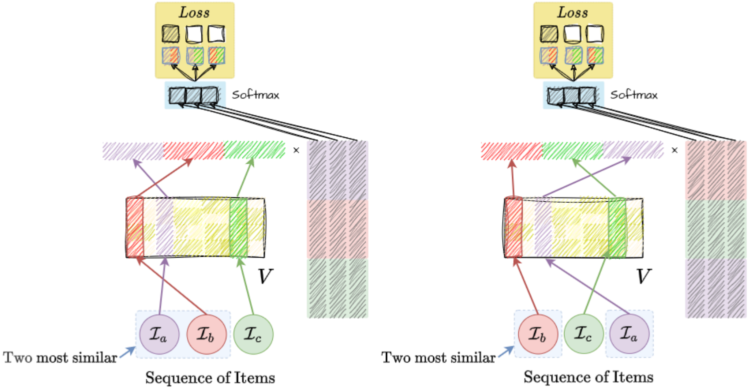
Sequential Recommendation (RECAP)



A First Try


Loss
Softmax
V
$\mathcal{I}_a$
$\mathcal{I}_b$
$\mathcal{I}_c$
Two most similar
Sequence of Items

Is it a Invariant Model?



SPoSE

- SPoSE is a model developed to learn individual representations from triplets.
- It is an acronym for **S**parse **Po**sitive **S**imilarity **E**mbedding.
- These representations predict a latent similarity structure between objects, capturing most of the explainable variance in human behavioral judgments.
- The model is based on the concept of probability. Let $S(a, b)$ be a function representing the similarity between a and b. One way to compute the probabilities of the three possible embeddings x_1, x_2, and x_3 such that they add up to one, can be given by:

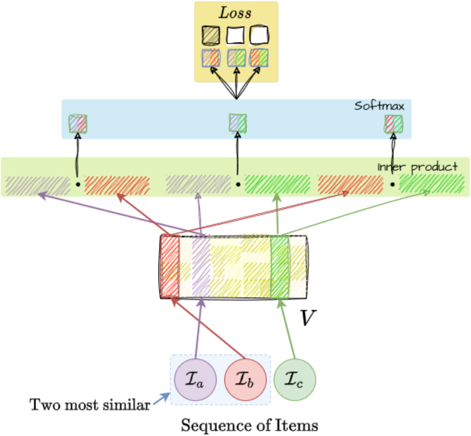
$$P(x_1, x_2) = \frac{e^{S(x_1,x_2)}}{e^{S(x_1,x_2)} + e^{S(x_1,x_3)} + e^{S(x_2,x_3)}}$$

SPoSE

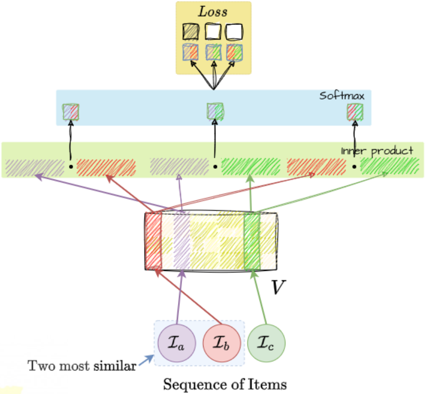
- Similarity can be computed in various ways.
- The authors of the SPoSE paper tried two methods: one based on **Euclidean distance** and another on **cosine similarity**.
- Experimentally, **cosine similarity** showed to be more effective.
- Given a set of triplets $T = \{t_1, t_2, \cdots t_n\}$,where $t_i = \{a_{i,1}, a_{i,2}, a_{i,3}\}$ and item $a_{i,1}$ is more similar to item $a_{i,2}$ (in other words, $a_{i,3}$ is the odd-one-out item),an embedding vector of item $a_{i,j}$ is represented by $x_{a_{i,j}}$. With the aim to learn $X = \{x_1, x_2, \cdots x_m\}$, we have

$$\text{argmin}_{x_j} \sum_{i=1}^{n} \log \frac{e^{x_{a_{i,1}}^\top x_{a_{i,2}}}}{e^{x_{a_{i,1}}^\top x_{a_{i,2}}} + e^{x_{a_{i,1}}^\top x_{a_{i,3}}} + e^{x_{a_{i,2}}^\top x_{a_{i,3}}}} + \lambda \sum_j |x_j|_1$$

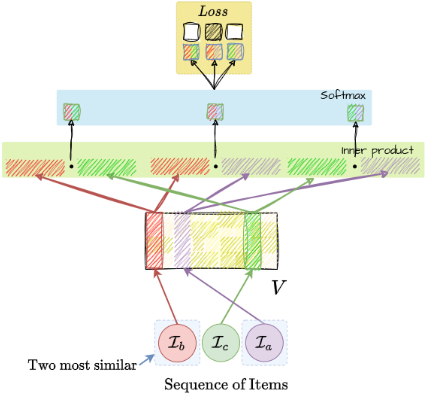
SPoSE


Loss
Softmax
Inner product
V
$\mathcal{I}_a$
$\mathcal{I}_b$
$\mathcal{I}_c$
Two most similar
Sequence of Items

SPoSE



Loss
Softmax
Inner product
V
$\mathcal{I}_a$
$\mathcal{I}_b$
$\mathcal{I}_c$
Two most similar
Sequence of Items

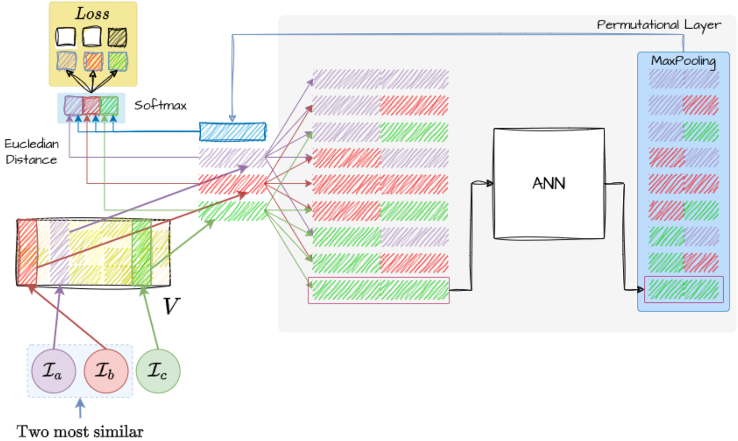


Loss
Softmax
Inner product
V
$\mathcal{I}_b$
$\mathcal{I}_c$
$\mathcal{I}_a$
Two most similar
Sequence of Items

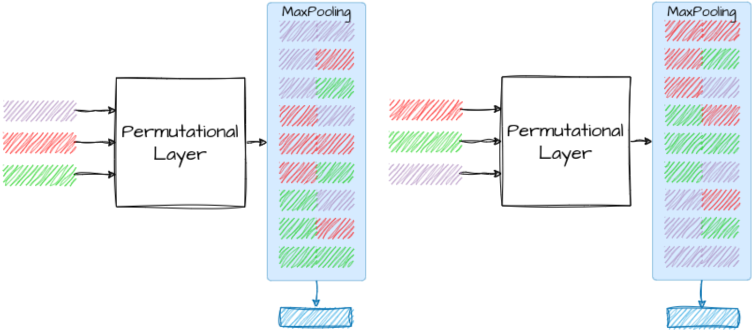
Permutational Layer

- Component of neural networks for handling variable-length input sequences.
- Processes **permutations of input elements**, allowing the model to handle different orders of sequence elements.
- Extracts features **irrespective of the original positions**, enhancing the model's understanding of relationships.
- Particularly useful in tasks where the order of elements should not influence the model's predictions, like set-based or graph-based data.
- Commonly applied in set classification tasks, where predictions are based on set properties rather than element order.

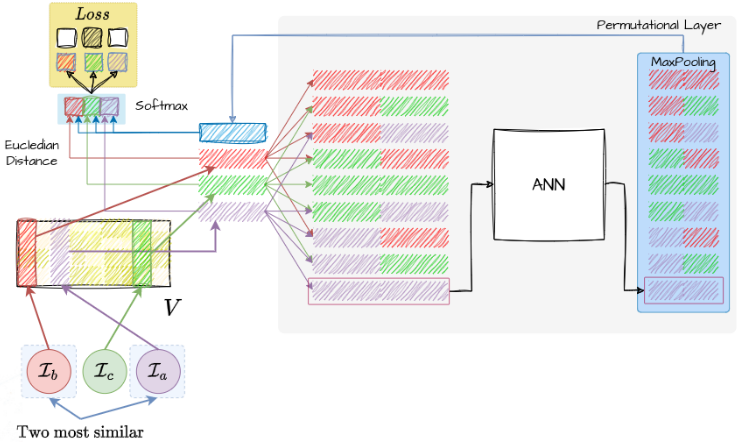
Permutational Layer



Permutational Layer



Permutational Layer



Obrigado :) - Faculty of Information Technology