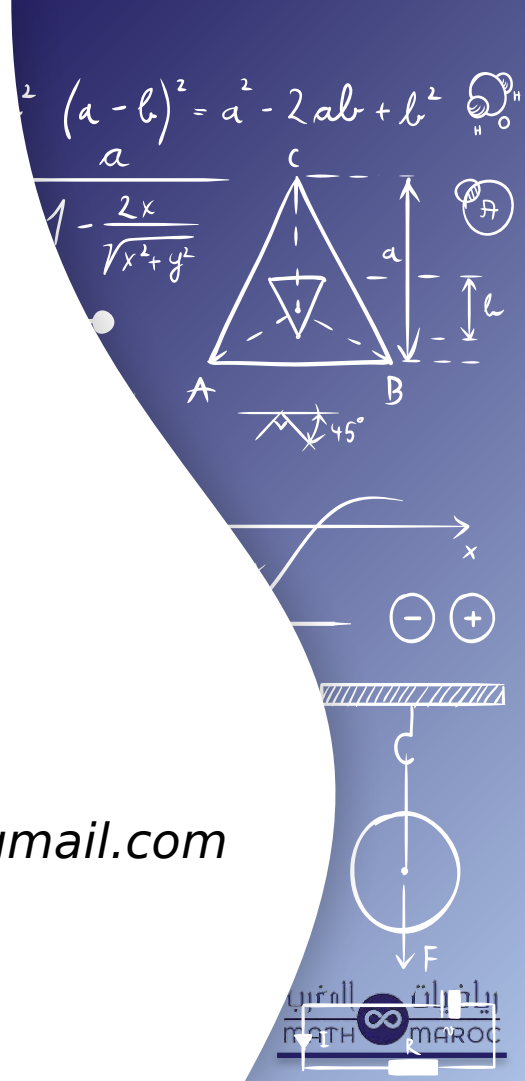
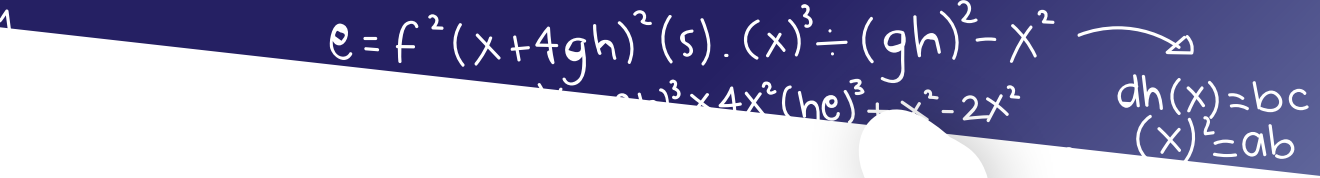
Math behind Artificial intelligence

RS and CVNN

Afaf EL WAFI - elwafiafaf@gmail.com





01

Recommendation systems

Personalize the offer with ML

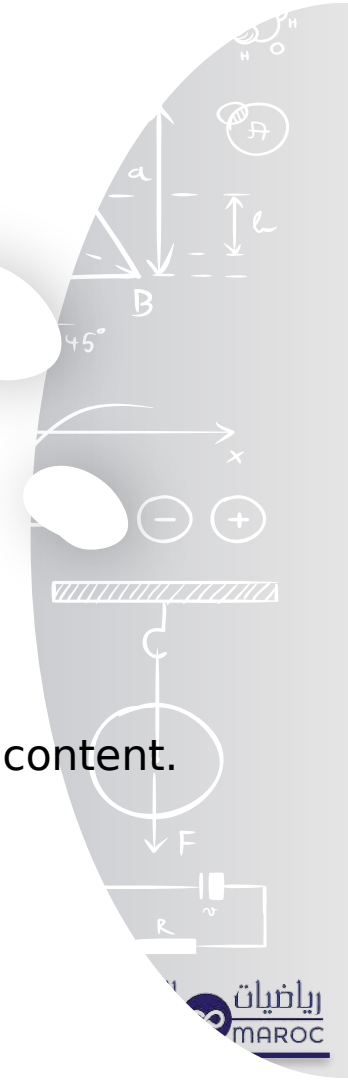
Figures

NETFLIX 2 out of 3 watched hours come from recommendations

YouTube increases its watch times by 50% per year

amazon 35% of all sales are generated by recommendations

facebook 8 milliards view on daily average due to a personalized content.



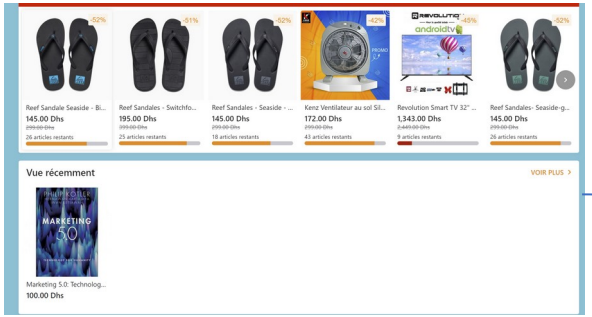
Recommendation systems

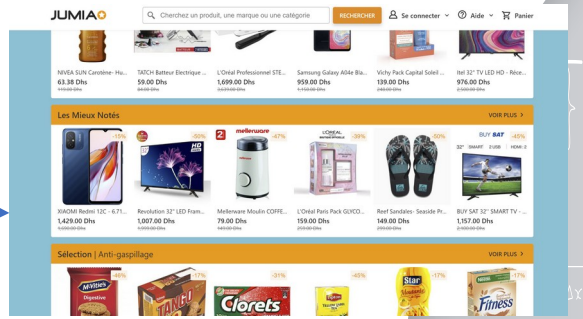
Dramatic increase of content-delivery services and multimedia content

Efficient mechanisms required for discovering preferred content

Recommendation systems (RS) address these problems

- Exploit information about user preferences
- Accurately & efficiently recommend the appropriate content





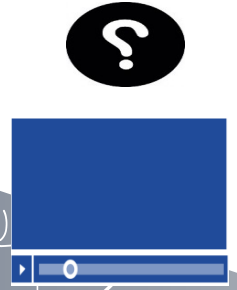
User feedback

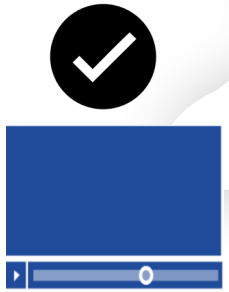
Explicit feedback: Users indicate their preference for items

- Ratings scales (1-5 stars, 1-10 stars)
- Binary values (liked/disliked ✉ 0/1)

Implicit feedback: Users interact with items

- Purchase history
- Viewing duration
- Click patterns





Rating matrix

items




Alice	3	3	?	...	2
	?	?	4	...	1
	5	4	?	...	?
	⋮	⋮	⋮	⋮	⋮
	3	?	?	...	3

users

rating: level of preference

unknown preferences

ratings matrix

- Sparse matrix: Most values are unknown
- Predicting: The task of filling the unknown values

Non-exhaustive list of RS models

Collaborative Filtering models
Inspect rating patters to find similar users/items

Content-based models
Analyze attributes of items for building user profiles

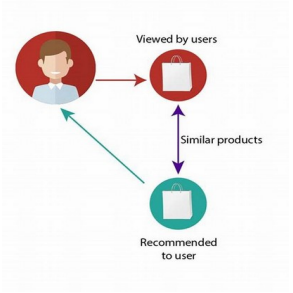
Knowledge-Based Recommender Systems
Allow the users to explicitly specify what they want
based on the similarities between customer requirements

Demographic Recommender Systems •
Map specific demographics to ratings or buying propensities

Context-Based Recommender Systems
Time, Location, Social (structural recommendation)

Hybrid and Ensemble-Based Recommender Systems
Generalization of classification/regression modeling in which the prediction is performed in entry-wise fashion rather than row-wise fashion

Content based filtering



Preprocessing and feature extraction

- Text embeddings : BOW , TDF-IDF, word2Vec...
- Feature selection
- Image preprocessing

Learning User Profiles and Filtering

Regression and Classification models
0/1 rating = Classification
Numerical ratings Regression

Similarity

$$Consine\ Similarity = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2}\sqrt{\sum_{i=1}^{n} {B_i}^2}}$$

$A = [1,1,1,0]$

$B = [0,1,0,1]$

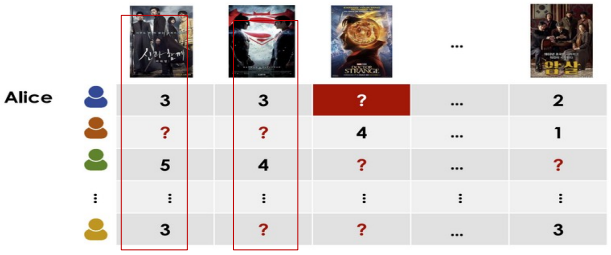
$\|A\| = \sqrt{1^2 + 1^2 + 1^2 + 0^2} = \sqrt{3}$

$\|B\| = \sqrt{0^2 + 1^2 + 0^2 + 1^2} = \sqrt{2}$

$A \cdot B = 1 * 0 + 1 * 1 + 1 * 0 + 0 * 1 = 1$

$$Consine\ Similarity = \frac{A \cdot B}{\|A\|\|B\|} = \frac{1}{\sqrt{3} \cdot \sqrt{2}} = 0.41$$

Collaborative Filtering: K-Nearest Neighbors Approach



Item-based collaborative filtering (IBCF)

Assumption: Users have similar tastes for similar items

- Item similarity: agreement within users rated both items
- Prediction: weighted sum of similar items' ratings

User-based collaborative filtering (UBCF)

Assumption: Similar users have similar preferences

- User similarity: agreement on co-rated items
- Prediction: weighted sum of similar user's ratings

Item-based Collaborative Filtering Models

- Similar items receive similar ratings
- Predict using the user's own ratings of neighboring items
- Provide relevant recommendations

U / I	1	2	3	4	5	6
A	7	6	7	4	5	4
B	6	7	?	4	3	4
C	?	3	3	1	1	?
D	1	2	2	3	3	4
E	1	?	1	2	3	3

Ratings matrix

I/U	1	2	3	4	5	6
A	1.5	0.5	1.5	-1.5	-0.5	-1.5
B	1.2	2.2	?	-0.8	-1.8	-0.8
C	?	1	1	-1	-1	?
D	-1.5	-0.5	-0.5	0.5	0.5	1.5
E	-1	?	-1	0	1	1
Cosine(1, j)	**1**	**0.735**	**0.912**	**-0.848**	**-0.813**	**-0.999**

Centered Ratings matrix

Example: items 2 and 3 are similar to 1, item 1 is predicted to have 3 ratings from user C

$$r_{C,1} =$$

Mean rating of user U is defined as:

$$\mu_u = \frac{\sum_{k \in I_u} r_{uk}}{|I_u|}$$

Compute Cosine similarity between items

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R}_u)(R_{u,j} - \bar{R}_u)}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R}_u)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R}_u)^2}}$$

set of users rated both items

rating on item *i*

rating on item

deviation from the average rating

Prediction of rating for item *i*

$$p_{u,i} = \frac{\sum_{j \in S} sim(i,j) R_{u,j}}{\sum_{j \in S} |sim(i,j)|}$$

most similar items to *i*

similarity as weight

rating on similar item

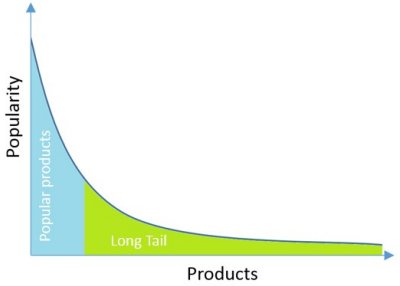
Long tail problem

- Some movies may be very popular, and they may repeatedly occur as commonly rated items by different users.

- The solution to this problem is **weighting**

- If is the number of ratings of item j, and m is the total number of users, then the weight of the item j:

$$w_j = \log\left(\frac{m}{m_j}\right) \quad \forall j \in \{1 \dots n\}$$



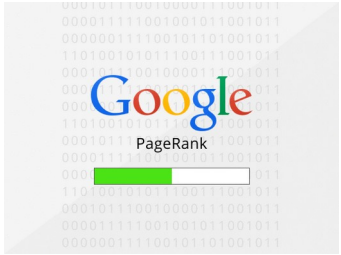
$$\text{Pearson}(u,v) = \frac{\sum_{k \in I_u \cap I_v} w_k \cdot (r_{uk} - \mu_u) \cdot (r_{vk} - \mu_v)}{\sqrt{\sum_{k \in I_u \cap I_v} w_k \cdot (r_{uk} - \mu_u)^2} \cdot \sqrt{\sum_{k \in I_u \cap I_v} w_k \cdot (r_{vk} - \mu_v)^2}}$$

Less popular items will have more weights

Graph Models for neighborhood-Based Methods

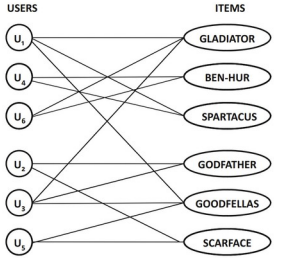
A random walk is a mathematical object, known as a stochastic or random process, that describes a path that consists of a succession of random steps on some mathematical space such as the integer

Defining Neighborhoods with Random Walks:



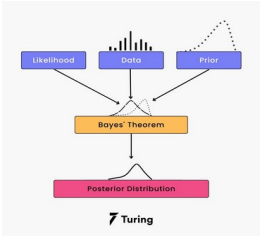
	GLADIATOR	GODFATHER	BEN-HUR	GOODFELLAS	SCARFACE	SPARTACUS
U_1	1			5		2
U_2		5			4	
U_3	5	3		1		
U_4			3			4
U_5				3	5	
U_6	5		4			

(a) Ratings matrix



(b) User-item graphs of specified ratings

Naïve Bayes model



The naive Bayes model is a generative model.

Bayes rule in probability theory:

$$P(A|B) = \frac{P(A) \cdot P(B|A)}{P(B)}$$

Item-Id ⇒ / User-Id ⇓	1	2	3	4	5	6
1	1	-1	1	-1	1	-1
2	1	1	?	-1	-1	-1
3	?	1	1	-1	-1	?
4	-1	-1	-1	1	1	1
5	-1	?	-1	1	1	1

Like/Dislike Ratings matrix

l distinct values of the ratings $v_1 \ldots v_l$.
As in the case of the other models discussed in this chapter, we assume that we have an $m \times n$ matrix R containing the ratings of m users for n items. r_{uj} is the rating if item j by user u .

$$P(r_{31} = 1|r_{32}, r_{33}, r_{34}, r_{35}) \propto P(r_{31} = 1) \cdot P(r_{32} = 1|r_{31} = 1) \cdot P(r_{33} = 1|r_{31} = 1) \cdot \\ \cdot P(r_{34} = -1|r_{31} = 1) \cdot P(r_{35} = -1|r_{31} = 1)$$

$$P(r_{uj} = v_s|\textit{Observed ratings in } I_u) = \frac{P(r_{uj} = v_s) \cdot P(\textit{Observed ratings in } I_u|r_{uj} = v_s)}{P(\textit{Observed ratings in } I_u)}$$

$$P(r_{uj} = v_s|\textit{Observed ratings in } I_u) \propto P(r_{uj} = v_s) \cdot P(\textit{Observed ratings in } I_u|r_{uj} = v_s)$$

$$P(r_{uj} = v_s|\textit{Observed ratings in } I_u) \propto P(r_{uj} = v_s) \cdot \prod_{k \in I_u} P(r_{uk}|r_{uj} = v_s)$$

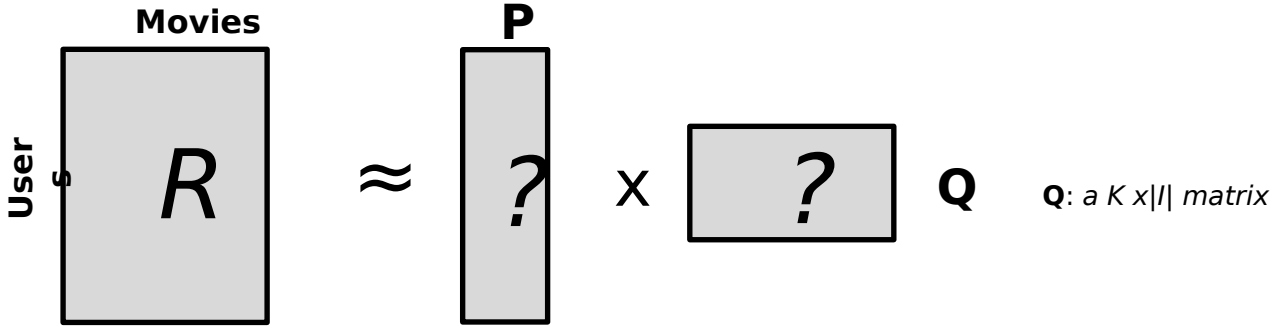
$$P(r_{31} = 1) = 2/4 = 0.5$$
$$P(r_{32} = 1|r_{31} = 1) = 1/2 = 0.5$$
$$P(r_{33} = 1|r_{31} = 1) = 1/1 = 1$$
$$P(r_{34} = -1|r_{31} = 1) = 2/2 = 1$$
$$P(r_{35} = -1|r_{31} = 1) = 1/2 = 0.5$$

Matrix - Factorization

Factorize a matrix: to find out two (or more) matrices such that when you multiply them you will get back the original matrix

Application perspective: can be used to discover latent features underlying the interactions between two different kinds of entities

P: *a |U| x K matrix*



Q: *a K x|I| matrix*

R: *Matrix of size |U| x |I| that contains all the ratings that the users have assigned to the items*

Factorization is a general way of approximating a matrix when it is prone to dimensionality reduction because of correlations between columns (or rows).

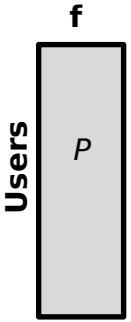
Matrix - Factorization

Each user & item is characterized with a vector of factors

User-factors vector

$$\begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_f \end{bmatrix}$$

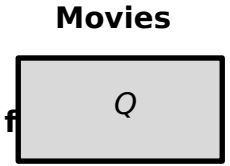
***P**: vector that represents the strength of the associations between a user and the features*



Item-factors vector

$$\begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_f \end{bmatrix}$$

***Q**: vector that represents the strength of the associations between an item and the features*



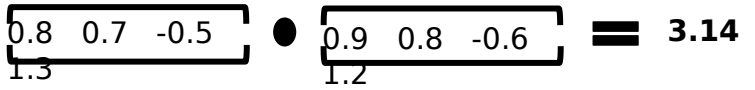
Factors - *f*: mathematic artifact for computing relationship

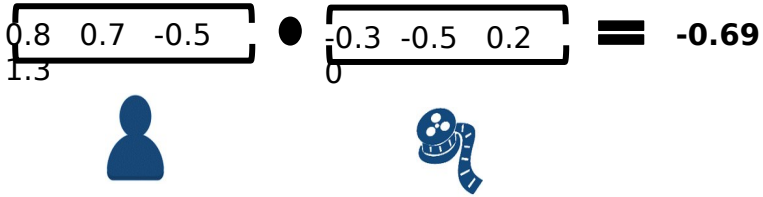
Matrix - Factorization

Preference - prediction is the dot product of user & item factors

Item factors: The extent to which an item has some characteristics

User factors: Level of preference for the corresponding characteristics





To get the prediction of a rating of an item by a user , calculate the dot product of the two vectors corresponding to and

$$\hat{r}_{ij} = p_i^T q_j = \sum_{k=1}^{k} p_{ik} q_{kj}$$

How do we obtain the P and Q matrices?

Matrix - Factorization

Minimization problem

1. First we initialize the matrices with some values
2. We calculate how ***'different'*** their product is to M
3. We try to minimize this difference iteratively.

So, it is really a minimization problem!!!!

Gradient Descent: Aims at finding a local minimum of the difference

Difference: The error between the estimated rating and the real rating. It can be calculated for each user-item pair:

$$e_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2$$

We consider the squared error because the estimated rating can be either higher or lower than the real one.

Matrix - Factorization

Gradient Descent

Local minimum of Error

To minimize the error, need to know in which direction we have to modify the values of and

We need to know the gradient at the current values.
Therefore, we differentiate the error equation with respect to and

Matrix - Factorization

Differentiation of Error Equation

$$= \frac{\partial \left(r_{ij} - \widetilde{r}_{ij} \right)^2}{\partial p_{ik}} = 2(= 2($$

$$= 2(= 2( )$$

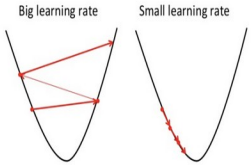
$$= 2)') = -2$$

$$= -2 = -2$$

Having obtained the gradient, we can now formulate the update rules for both and

$$p'_{ik} = p_{ik} + \alpha \frac{\partial}{\partial p_{ik}} e_{ij}^2 = p_{ik} + 2\alpha e_{ij} q_{kj}$$

$$q'_{kj} = q_{kj} + \alpha \frac{\partial}{\partial q_{kj}} e_{ij}^2 = q_{kj} + 2\alpha e_{ij} p_{ik}$$



Cost function

$$\min_{q^*, p^*} \sum_{(u,i) \in \kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2)$$

α: constant that determines the rate of approaching the minimum

Other Matrix Factorization Metho

Method	Constraints	Objective	Advantages/ Disadvantages
Unconstrained	No constraints	Frobenius + regularizer	Highest quality solution Good for most matrices Regularization prevents overfitting Poor interpretability
SVD	Orthogonal Basis	Frobenius + Regularizer	Good visual interpretability Out-of-sample recommendations Good for dense matrices Poor semantic interpretability Suboptimal in sparse matrices
Max. Margin	No coonstraints	Hinge loss + margin regularizer	Highest quality solution Resists overfitting Similar to unconstrained Poor interpretability Good for discrete ratings
NMF	Non-negativity	Frobenius + Regularizer	Good quality solution High semantic interpretability Loses interpretability with both like/dislike ratings Less overfitting in some cases Best for implicit feedback
PLSA	Non-negativity	Maximum Likelihood + regularizer	Good quality solution High semantic interpretability Probabilistic interpretation Loses interpretability with both like/dislike ratings Less overfitting in some cases Best for implicit feedback

Singular Value Decomposition (SVD)
Columns of U and V are constrained to be mutually orthogonal

Maximum Margin Factorization

- Borrows ideas from support vector machines to add a maximum margin regularizer to the objective function and some of its variants
- particularly effective for discrete ratings

Non-negative Matrix Factorization
Provide high-level interpretability

Probabilistic Latent Semantic Analysis (PLSA)

A probabilistic variant of non-negative matrix factorization.

Evaluating recommender systems

Hit rate:
Generate a top-N recommendation for a user. If one of the recommendations in a user's top-end recommendations is something they actually rated, you consider that a hit

HR =

Average reciprocal hit rate (ARHR):

ARHR =

Normalized Discounted Cumulative Gain NDCG:
Gain is just the relevance score for each item recommended.

- **Cumulative Gain(CG):**
  Cumulative gain at K is the sum of gains of the first K items recommended. $CG_{@K} = \sum_{i=1}^{K} G_i$

- **Discounted Cumulative Gain DGC?**
  Discounted cumulative gain weighs each relevance score based on its position. $DCG_{@K} = \sum_{i=1}^{K} \frac{G_i}{log_2(i+1)}$

$$NDCG_{@K} = \frac{DCG_{@K}}{IDCG_{@K}}$$

$$IDCG_{@K} = \sum_{i=1}^{K^{ideal}} \frac{G_i^{ideal}}{log_2(i+1)}$$

Evaluating recommender systems

Example

Order	Recommender: Movie Order	Recommender: Movie Relevance	Ground Truth: Movie Order	Ground Truth: Movie Relevance
1	d10	3	d3	4
2	d3	4	d10	3
3	d1	3	d1	3

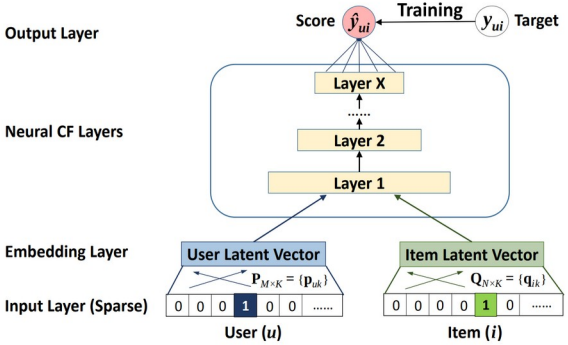
$$DCG_{@3} = \frac{3}{log(1+1)} + \frac{4}{log(2+1)} + \frac{3}{log(3+1)}$$

$$IDCG_{@3} = \frac{4}{log(1+1)} + \frac{3}{log(2+1)} + \frac{3}{log(3+1)}$$

$$NDCG_{@3} = \frac{7.02}{7.4} = 0.94$$

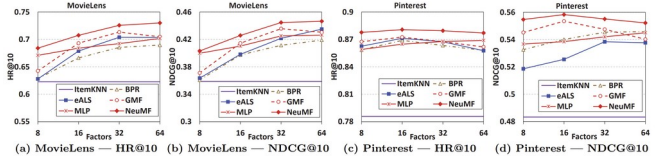
Neural Collaborative Filtering (NCF)

Neural Collaborative Filtering (Multi-layer Perceptron)



$$y_{ui} = \begin{cases} 1, & \text{if interaction (user } u\text{, item } i\text{) is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

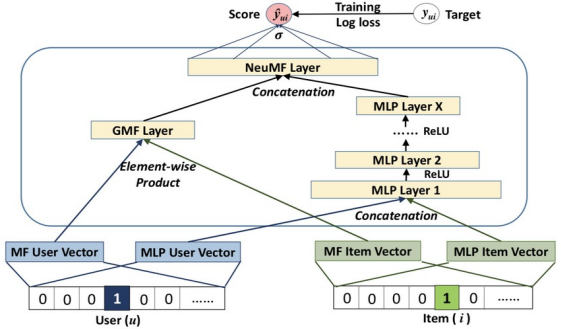
$$\mathbf{z}_1 = \phi_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix},$$
$$\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2),$$
$$\ldots\ldots$$
$$\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L),$$
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \phi_L(\mathbf{z}_{L-1})),$$



(a) MovieLens — HR@10 (b) MovieLens — NDCG@10 (c) Pinterest — HR@10 (d) Pinterest — NDCG@10

Figure 4: Performance of HR@10 and NDCG@10 *w.r.t.* the number of predictive factors on the two datasets.

Neural Matrix Factorization Model (MLP + GMF)

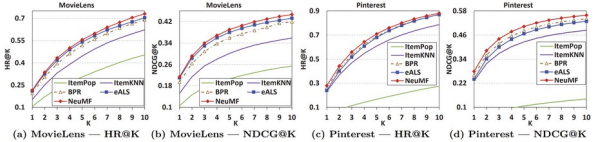


We define the mapping function of the GMF layer as

$$\phi_1(\mathbf{p}_u, \mathbf{q}_i) = \mathbf{p}_u \odot \mathbf{q}_i$$

Where it denotes the element-wise product of vectors.

$$\phi^{GMF} = \mathbf{p}_u^G \odot \mathbf{q}_i^G,$$
$$\phi^{MLP} = a_L(\mathbf{W}_L^T(a_{L-1}(\ldots a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2)\ldots)) + \mathbf{b}_L)$$
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \begin{bmatrix} \phi^{GMF} \\ \phi^{MLP} \end{bmatrix}),$$



(a) MovieLens — HR@K (b) MovieLens — NDCG@K (c) Pinterest — HR@K (d) Pinterest — NDCG@K

Figure 5: Evaluation of Top-K item recommendation where K ranges from 1 to 10 on the two datasets.

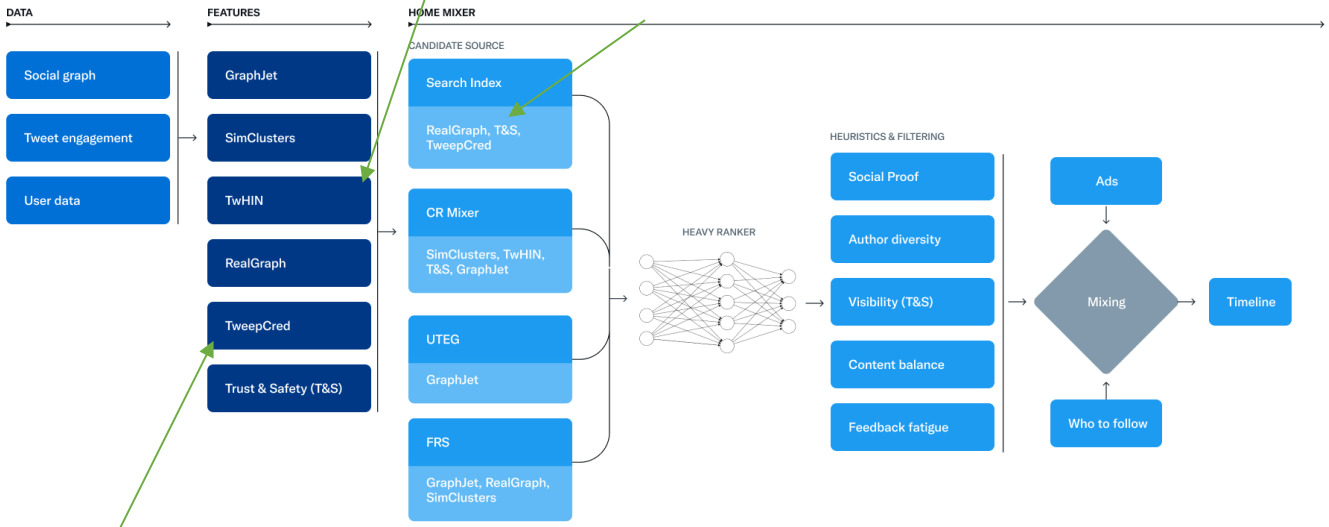
Twitter use case: Home mixer

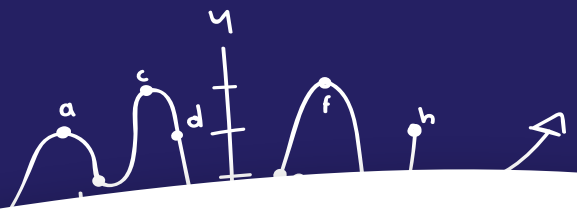
The recommendation pipeline is made up of three main stages that consume these features:

1. Fetch the best Tweets from different recommendation sources in a process called **candidate sourcing.**
2. **Rank** each Tweet using a machine learning model.
3. Apply **heuristics and filters**, such as **filtering** out Tweets from users you've blocked, NSFW content, and Tweets you've already seen.

Dense knowledge graph embeddings for Users and Tweets.

Real-graph : Gradient boosting tree





$e = f^2(x+4gh)^2(s).(x)^3 \div (gh)^2 - x^2$

$f = gh^2 + (s)(x+2h)^3 \times 4x^2(he)^3 + x^2 - 2x^2$

$g - x^2 \div (x)(2x)^2 + (hfe)^2 4x^3 (3h)(f)^2(e)^2 + x^2 4s^2$

$(3)^2(f)^3 + x(4x)^2$

$dh(x) = bc$

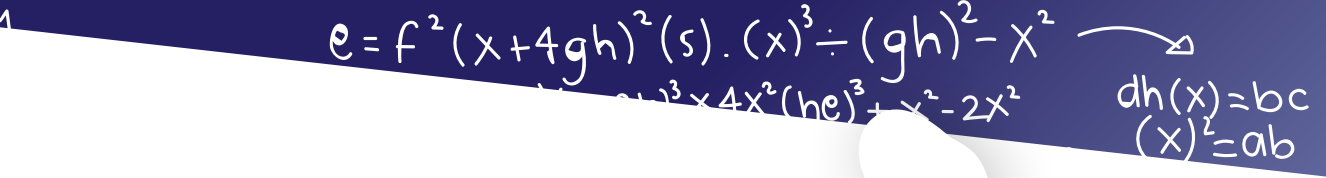
$(x)^2 = ab$

$4x^2 + (ef)^2$

Selecting the **best** recommendation algorithm is **challenging**
Depending on input, the performance of a recommender varies

Hybrid recommendation systems address these limitations

- Integrate various recommenders
- Focus on their strengths & weaknesses
- Provide more accurate recommendations

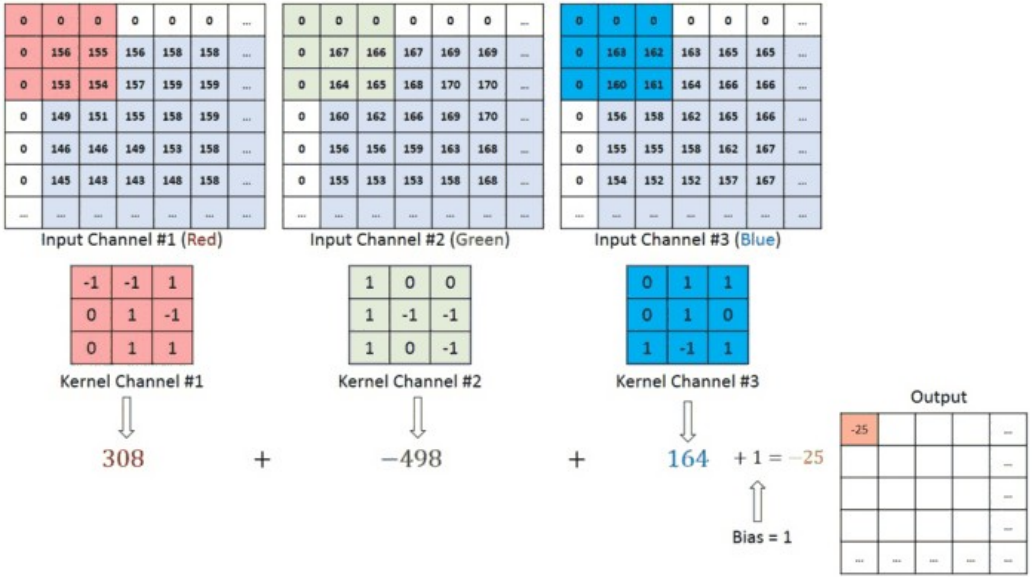


02

Complex valued neural networks

Leveraging the power of NN

Convolutional neural network



CNNs have been the go-to for Computer Vision primarily due to their ability to build feature maps. Even with the rise of Vision Transformers, CNNs have held strong

The CNN can be very expensive as we add more and more layers to extract high resolution features. **You must slide the window across multiple times**.

What if we did some Math to find a function that can help us spot some low-level features directly?

Complex functions in image preprocessing (Hybrid models)

(a) Source Lena image. (b) Edge detection using Haar wavelet transform (c) Edge detection using CoShRem transform

Figure 1: wavelet transform fails to capture curvilinear edges which are successfully captured by the CoShRem.

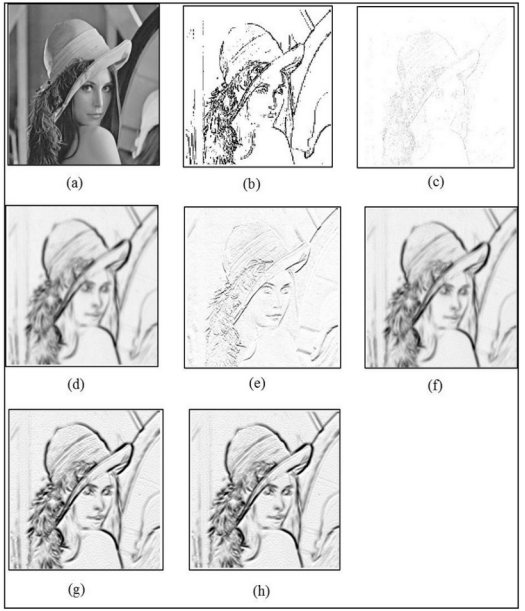
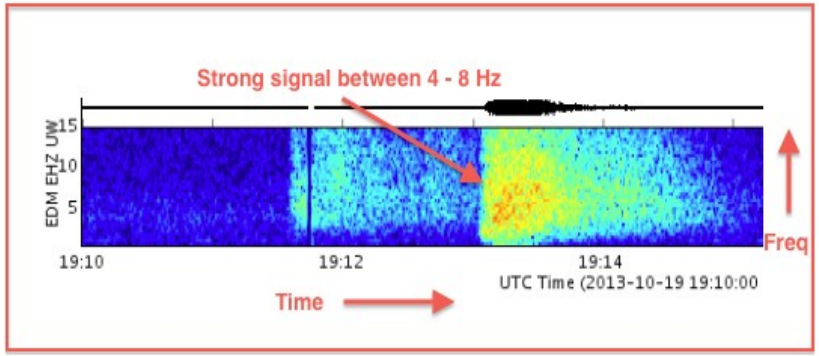


Fig. 2. *Original image (b) Haar (c) Laplacian, (d) Gabor, (e) Sobel (f) Laplacian of Gabor [proposed] (g) Sobel of Gabor and [proposed] (h) Laplacian of Sobel of Gabor [proposed].*

Phase in complex valued data (radar data)

Hidden information in the phase

In electronic signaling, **phase is a definition of the position of a point in time (instant) on a waveform cycle**.
Phase Congruency extracts stable features - edges, ridges and blobs - that are contrast invariant.



Complex-Valued Neural Networks (CoSh Network)

2 Contributions

- **A Compact hybrid and efficient CNN architecture**. CoShNet uses a fixed 2nd-generation wavelet transform to produce informative and stable representations for classification. In addition, through a series of tests, CoShNet has demonstrated state of the art results using a fraction of parameters ($49.9k$) versus ResNet-18 with $11.2m$. The architecture does not require any regularization nor hyperparameter tuning and is extremely stable with respect to initialization and training.

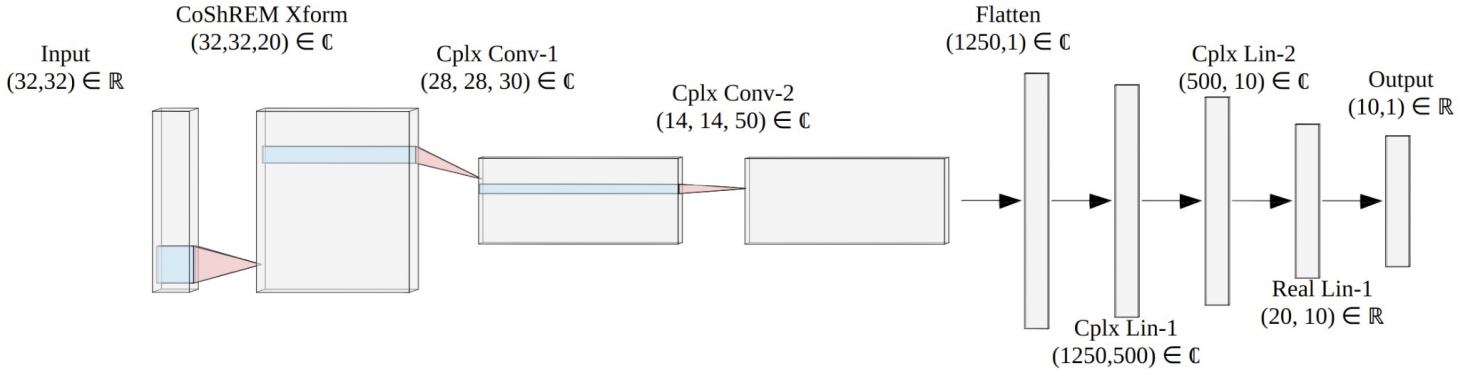
- **Exceptional Generalization**. Training with the $10k$ samples from Fashion and testing using the remaining $60k$ samples, we are able to obtain Top-1 accuracy close to 90%.

- **Complex-valued neural network** (CVnn) a CVnn has many attractive properties absent in its real valued counterpart. Nitta [9][10] studied the critical-points of real vs. complex neural networks and showed that a lot of critical-points of regular CNNs are local minima, whereas CVnn's critical-points are mostly saddle-points. Nitta [11] also showed that the decision surface 4.1 of a CVnn has more representational power and is self regularizing. However, a good implementation in the complex domain is non-trivial. In S. 4 we will layout the important components for a efficient CVnn that performs competitively.

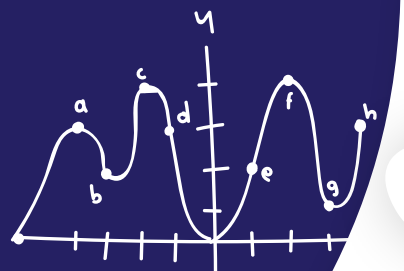
- **Fully complex CVnn** CoShRem is unique among CVnns as a fully complex network - all operations are performed using complex operators.

- **Training a compressed model**. We present a novel way of training a highly compressed version of CoShNet by using tensor compression and DCFNet [12] while preserving all of its good qualities. Unlike most efforts on model compression, the compressed model is trained in one pass.

Complex-Valued Neural Networks (CoSh Network)



Model	Epochs	# of Parameters	Size Ratio	Top1 Accuracy (60k)	Accuracy (10k)
ResNet-18	100	11.18M	229	91.8%	88.3%
ResNet-50	100	23.53M	481	90.7%	87.8%
CoShNet (base)	20	1.37M	28	**92.2%**	**89.2%**
tiny-CoShNet	20	49.99K	1	91.6%	88.0%



$(x)^2 = ab$

$dh(x) = bc$

Thanks!

Do you have any questions?

elwafiafaf@gmail.com

رياضيات المغرب
MATH MAROC

Resources

1. CoShNet: A Hybrid Complex Valued Neural Network using Shearlets
2. Recommendation sytems
3. Recommender Systems with Random Walks: A Survey Laknath Semage
4. Bayesian networks for recommendation systems
5. Fundamental of MF for RS