CONTEXT AWARE REC..

CHP 12

↳ not a technique but an improvement over an existing technique

Context

→ an imp factor in personalized recommendation

↳ extra info used to improve rec

↳ 2 views of context

↳ Representational

↳ attributes that dont change over time

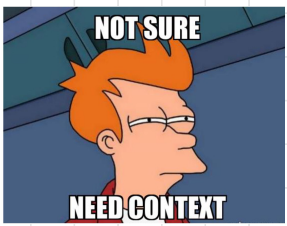
e.g first language, date of birth, siblings

↳ Interactionable

↳ change over a period of time

↳ an active relation b/w attributes and user activity

e.g budget, after watching movie its taste changed



Types of attributes

1. Fully Observable : contextual factors are known explicitly
2. Partially Observable : only some info is known explicitly

e.g lives in Pakistan so must know Urdu → so spotify rec. urdu songs

3. Unobservable : no explicit info

e.g blood group, my current location

Types of architectures for using context

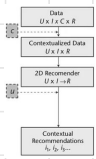
1. Contextual Prefiltering
2. Contextual Post filtering
3. Contextual Modeling

1. Contextual Pre-Filtering

change data

↳ from avl set of products
remove irrelevant products
↳ send the filtered set to a rec. algo

e.g Partially info → i speak urdu
before suggesting remove all
songs of other languages
then recommend



- Context is used to select some set of data and then predict like a traditional recommender system.

Example: *If someone wants to watch a movie on Saturday, then only use data (movies) that were rated on Saturdays*

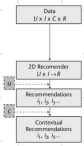
PRO
↳ some context is better than no context
↳ minimised computational cost → as less data
↳ easy

CON
↳ low serendipity → on the basis of context randomness hota dia

2. Contextual Post-Filtering

arrange results

↳ give all data
↳ after being recommended set of products
remove irrelevant products from the set
↳ ratings are predicted,
then filtered using context
↳ only rearranges algos result



Example: *If someone wants to watch a comedy movie, then generate recommendations over all movies, then filter out/push back all other genre (assign weights)*

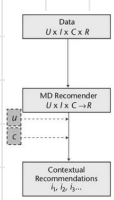
PRO
↳ recommendations can be ranked w.r.t context
↳ has serendipity
↳ easy

CON
↳ cant evaluate performance of a rec system technique
↳ has computational cost → as all data used

3. Contextual Modeling

example

↳ dont change data or prediction
↳ modify algo code such that
it involves context



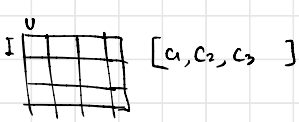
- The context is used right in the model. It is more complex and could be implemented by multiple machine learning models

PRO
↳ way more accurate
↳ balanced serendipity → as using all data

CON
↳ higher complexity
↳ difficult to implement in practice

Contextual Modeling of Collaborative Filtering

↳ change prediction model
↳ use weights



Transformers

CNP 12

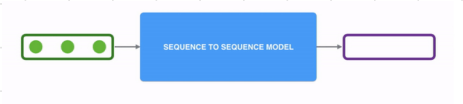
Background

↳ In some problems, we need neural network to remember the context

1. Sequence transduction problems
   ↳ input → sequence
   ↳ output → sequence
2. Machine translation e.g English to Spanish
   ↳ input a sequence
   ↳ output a sequence to another language
3. Image captioning
   ↳ input → image features sequence
   ↳ output → sequence of words

1. Machine Translation

↳ for this problem we use Seq2seq (sequence to sequence) models → Deep Learning



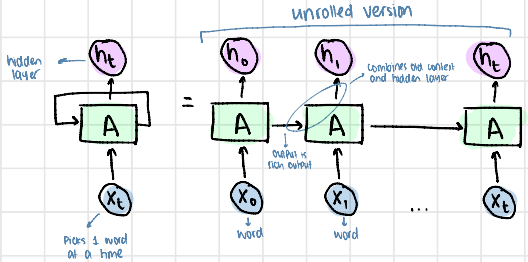
↳ To translate correctly we need to maintain context
↳ by making the neural networks understand the dependencies b/w 2 words
↳ to do this the model needs to remember past information → previous input

↳ normal NN dont remember previous inputs
   ↳ in terms of weights remembers
   ↳ weights are info of previous info → implicit

solution

Recurrent Neural Networks (RNN)

↳ uses loops to allow info to persist b/w inputs
↳ short term dependency
↳ each word is an input, which is encoded as the output of the hidden layer
↳ which is fed to the next layer
   ↳ along with another input word in the sequence



long term dependency

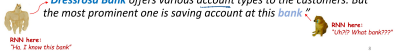
↳ to remember past context

❑ The output from the final encoder is passed to the second part of the network to get ***decoded*** into the target sequence

CON

↳ slow training
↳ cant grasp long term dependencies b/w words
   ↳ doesn't works well for long input sequence as over a period of time it forgets
   e.g 1000 words
   till 1000th word, it forgets 1st word

❑ **Example:**
"***Dressrosa Bank*** offers various account types to the customers. But the most prominent one is saving account at this bank."
(might remember it till here)

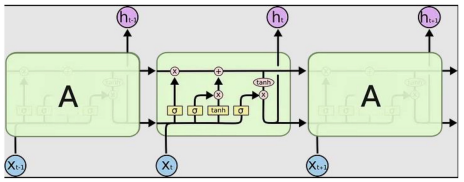


↳ the RNN will forget Dressrosa Bank

RNN with LSTM (Long Short Term Memory)

↳ Solves long term dependency problem in RNN

↳ It allows some input info to "float across" the encoder layers



CON

↳ Lack of parallelization

↳ Slow training

↳ can grasp long term dependencies

UNTIL they become too long

SOLUTION

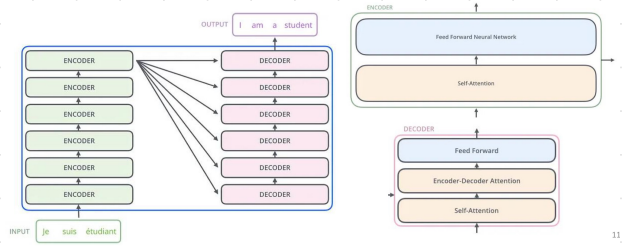
Attention

↳ a technique that allows input word information to be passed all the way to the decoders

Transformer Architecture

↳ It uses self attention and

↳ boots the training speed by allowing parallelization



Why Rec. system in Transformers

↳ at times users don't directly ask for a product

- ↳ no ratings → no collaborative filtering
- ↳ no features → no content based
- ↳ no user requirements → no case based

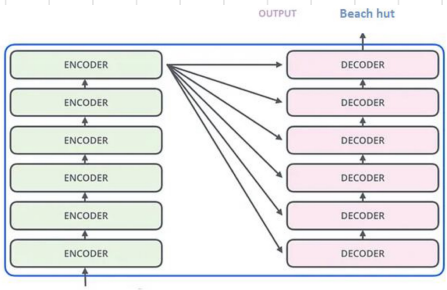
↳ what if recommendations are to be made w/o any of these?

i.e. translating user activity or dialogue to product recommendation

- Collaborative filtering requires historical interactions, content-based filtering requires user's rating profile, neural recommendations require both item and user feature embedding

↳ Transformers can grasp the context from users dialogue then translate it into a vector of relavent products

- Acquiring the input sequence from user involves both technical and ethical considerations
  e.g. should we use the microphone of user's mobile to gather the relevant context



Future Direction

1. Dialogue based rec.
2. Deduction based rec.
3. Generative based rec.

e.g. you message your friend abt a farmhouse
you start getting ads for farmhouse

e.g. you think something
and it starts showing on your feed.

Evaluation Metrics

Chp 14

1. Precision @ k (P@k)

↳ of the top k relavent items that were recommended
the no. of items that are actually relavent

Precision = $\frac{\text{Relavent Retrived}}{\text{Total Retrived}}$

$$P = \frac{\#\ of\ items\ recommended\ that\ are\ relevant}{\#\ of\ all\ items\ that\ can\ possibly\ be\ relevant}$$

Precision@k = $\frac{\text{True Positive in subset k}}{\text{True Positive in subset k + False Positive in subset k}}$

$$P@k = \frac{\#\ of\ items\ recommended\ that\ are\ relevant}{\#\ of\ items\ that\ were\ recommended\ i.e., k}$$

2. Average Precision @ k (AP@k)

↳ for a single user
the mean of precision@i for i = 1,2,... k

$$AP@k = \frac{1}{m} \sum_{k=1}^{n} P@k \text{ if } k^{th} \text{ item is relavent}$$

(n: no. of items; m: no. of possible relavent items)

$$AP@k = \frac{1}{\#\ of\ all\ possible\ relevant\ items} (P@1\ if\ 1^{st}\ items\ is\ relevant + P@2\ if\ 2^{nd}\ item\ is\ relevant + \cdots + P@k\ if\ k^{th}\ item\ is\ relevant)$$

3. Mean Average Precision @ k (MAP@k)

↳ the mean of AP@k over all users

$$MAP = \frac{\sum_{k=1}^{K} AP@k}{k}$$

$$MAP = \frac{Sum\ of\ AP@k\ for\ k = 1, 2, \ldots, upto\ K\ items\ that\ are\ recommended}{K}$$

Question 1: For each of the given scenarios and the choices provided, identify the most suitable technique that should be used and provide a brief reason for your selection.

i. An internet service provider that wants to recommend customized bundles to the users.

↳ customized bundles are user specific by users requirements and constraints

a) User-based collaborative filtering
b) Matrix factorization
c) Content-based filtering
d) Constraint-based systems ✓

ii. An e-commerce website that provides visitors an option to buy either new or used apparel.

↳ as it computes similarity b/w items, so it helps recommend similar products (used/new)

a) Matrix factorization
b) Item-based collaborative filtering ✓
c) User-based collaborative filtering
d) Transformer-guided systems

iii. An online courses platform with a selection of hundreds of technical courses that are available in both paid and premium.

↳ as it focuses on course features, allowing personalized rec. based on users past preferences

a) Content-based filtering ✓
b) User-based collaborative filtering
c) Matrix factorization
d) Naïve Bayes collaborative filtering

iv. An online hotel reservation and tourism website.

↳ as for specific cases eg travel preferences destination features

a) Case-based systems ✓
b) Constraint-based systems
c) Naïve Bayes collaborative filtering
d) Item-based collaborative filtering

v. A video sharing platform that allows users to like or dislike a video.

a) Content-based filtering
b) Item-based collaborative filtering ✓
c) Naïve Bayes collaborative filtering
d) Case-based systems

↳ as analyses users interactions with specific videos to rec. similar content

a) How does information overload affect the recommendation problem?
Information Overload: Increases the complexity of identifying relevant recommendations, leading to decreased user satisfaction and system efficiency.

b) Name any four paradigms of recommender systems.
Collaborative Filtering
Content-Based Filtering
Demographic
Social tagging
Trust based
Hybrid Recommender Systems

c) Can Collaborative Filtering handle outdated ratings? Justify your answer with a reason.
Outdated Ratings: Collaborative Filtering can handle outdated ratings if temporal information is incorporated (e.g., time decay functions).

d) Explain the terms contextual pre-filtering, contextual modelling, and contextual post-filtering.
Pre-filtering: Filters data before generating recommendations.
Modelling: Incorporates context directly into the algorithm.
Post-filtering: Filters recommendations based on context.

e) Why do we use significance weighting?
Significance Weighting: Reduces the impact of noisy or unreliable data.

f) What is a long-tail problem?
Long-Tail Problem: Popular items dominate recommendations, reducing exposure to less-known items.

g) What is the difference between model-based and memory-based recommender systems?
Model-Based vs Memory-Based:
Model-Based: Learns patterns using algorithms (e.g., Matrix Factorization).
Memory-Based: Relies on historical interactions (e.g., Collaborative Filtering).

h) How well does a content-based system perform in terms of scalability?
Scalability of Content-Based Systems: Moderate scalability; it depends on the number of features rather than the number of users.

i) Name two major advantages of Naïve Bayes Filtering over regular Collaborative Filtering.
Advantages of Naïve Bayes:
Simple implementation.
Handles sparse datasets well.

j) Will Pearson Correlation be an appropriate similarity measure to use if the underlying
distribution of ratings is uniform? Justify your answer with a reason.
Pearson Correlation: Not appropriate for uniform distributions, as it assumes linear relationships.

Memory Based

↳ uses entire data everytime rating is predicted → goes data calculates formula EACH TIME

↳ User based CF

Model Based

↳ uses data once to create a model then makes new prediction w/o using the entire data again → pre stores traversed data

↳ Item based CF → get similarity b/w items and store them offline

↳ Content based RS

Question 2: Collaborative Filtering (CLO: 1) — **20 points** (8 + 8 + 4)

a) Consider the following interaction matrix extracted from *MovieLens* dataset:

	John Wick	Wall-E	Jaws	Avengers
Rohail	5	1	?	2
Saif	1	5	2	5
Ahsan	2	?	3	4
Qasim	4	3	5	?

Mean
2.667
3.25
3
4

Use item-based collaborative filtering (with adj. cosine) to predict *R*(Ahsan, Wall-E).

	JW	WE	J	A
R	2.333	-1.667		-0.667
S	-2.2	1.75	-1.25	1.75
A	-1	?	0	1
B	0	-1	1	

b) For the interaction matrix presented in Part (a), write Python code snippet to perform the following operations:

I. Read the matrix from ratings.csv into a dataframe named "*inter_matrix*".

II. Print all the ratings given by the user "*Saif*".

c) For the matrix in Part (a), find the user that is most similar to the active user (*Ahsan*) using Pearson Correlation.

a)

$$WE,JW = \frac{(-1.667)(2.333)+(1.75)(-2.2)+(-1)(0)}{\sqrt{1.667^2+1.75^2+1^2}\cdot\sqrt{2.333^2+2.2^2+0^2}} = -0.923$$

$$WE,J = \frac{(1.75)(-1.25)+(-1)(1)}{\sqrt{1.75^2+1^2}\cdot\sqrt{1.25^2+1^2}} = 0.988 \checkmark$$

$$WE,A = \frac{(-1.667)(-0.667)+(1.75)(-1.25)}{\sqrt{-1.667^2+1.75^2}\cdot\sqrt{-0.667^2+1.25^2}} = -0.314 \checkmark$$

$$R_{WE,A} = \frac{(3)(0.988)+(4)(-0.314)}{|0.988|+|0.314|} = -3$$

b)

```python
import pandas as pd

# I. Read matrix from CSV
inter_matrix = pd.read_csv('ratings.csv')

# II. Print ratings by Saif
saif_ratings = inter_matrix.loc['Saif']
print(saif_ratings)
```

c)

Question 5: Content-based Filtering (CLO: 2) **20 points** (15 + 5)

a) Consider the following interaction table for news articles and keywords where *"Article 2"* is the current (active) item. The table is populated with pre-computed TF-IDF values. Perform the given tasks.

	Covid-19	Tesla	Economy	Export	Magnitude
Article 1	3.05	5.07	0.10	1.48	6.1
Article 2	0.30	3.39	5.06	0.90	6.164
Article 3	0.40	4.66	0.30	1.32	4.869
Article 4	0.44	2.50	1.30	0.07	2.859

I. Calculate normalized TF-IDF values for this matrix.
II. Find the article that should be recommended to the user.

b) For the interaction table presented in Part (a), write Python code snippet to save it to a dataframe named "*news*" and then calculate the pairwise cosine similarity on it.

Normalised TF-IDF

	Covid	Tesla	Eco	Export	cosine similarity with A2
A1	0.5	0.831	0.016	0.243	
A2	0.049	0.549	0.821	0.146	
A3	0.082	0.957	0.062	0.211	
A4	0.154	0.876	0.457	0.025	

python Copy code

```python
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity

# Save table to DataFrame
data = {'Covid-19': [...], 'Tesla': [...], 'Economy': [...], 'Export': [...]}
news = pd.DataFrame(data)

# Pairwise cosine similarity
similarity = cosine_similarity(news)
print(similarity)
```

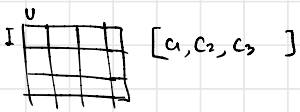
Question 6(a): For a context-aware laptop recommendation system that uses contextual modelling. Write a modified item-based collaborative filtering algorithm. The algorithm should be concise and must clearly highlight how context is added.

Include context as an additional factor in similarity calculations or matrix factorization.

Contextual Modeling of Collaborative Filtering

↳ change Prediction model

↳ use weights



Question 6(b): Consider the set *R* containing 50 items recommended to the user, set *P* containing the items actually liked by the user. And set *M* containing all items. Write a Python code snippet to find precision@*50*.

```python
def precision_at_k(R, P, k=50):
    relevant_items = set(P)
    recommended_items = set(R[:k])   # gets top K items
    precision = len(relevant_items & recommended_items) / K
    return precision

# Example usage
R = [...]  # Recommended items
P = [...]  # Liked items
print(precision_at_k(R, P))
```

R: 50 → recommended items

P: → relavent items

M: → all items

P@50: $\frac{P \cap R_K}{K}$

r

Question 3: Probabilistic Filtering (CLO: 1) — **20 points** (15 + 5)

a) Consider the following matrix which represents data for the performance rating given by auditors to each of the branches. The rating system is {1, 2, 3}.

	Branch 1	Branch 2	Branch 3
Auditor 1	1	1	2
Auditor 2	2	?	1
Auditor 3	3	1	3

Find the missing rating using Naïve Bayes Collaborative Filtering based on users.

user based

b) Explain the role of serendipity in probabilistic collaborative filtering approach.

Likelihood

$P(x|r_{B2}=1) = P(r_{B1}=2|r_{B2}=1) \times P(r_{B3}=1|r_{B2}=1)$

$= \frac{0+0.01}{2+0.05} \times \frac{0+0.01}{2+0.05} = 2.37 \times 10^{-5}$

$P(x|r_{B2}=2) = P(r_{B1}=2|r_{B2}=2) \times P(r_{B3}=1|r_{B2}=2)$

$= \frac{0+0.01}{0+0.05} \times \frac{0+0.01}{0+0.05} = 0.04$

$P(x|r_{B2}=3) = P(r_{B1}=2|r_{B2}=3) \times P(r_{B3}=1|r_{B2}=3)$

$= \frac{0+0.01}{0+0.05} \times \frac{0+0.01}{0+0.05} = 0.04$

rating is either 2 or 3

Prior

$P(r_{B2}=1) = \frac{2+0.01}{2+0.05} = 0.98$

$P(r_{B2}=2) = \frac{0+0.01}{2+0.05} = 4.87 \times 10^{-3}$

$P(r_{B2}=3) = \frac{0+0.01}{2+0.05} = 4.87 \times 10^{-3}$

Posterior

$= 2.32 \times 10^{-5}$

$= 1.948 \times 10^{-4}$

$= 1.948 \times 10^{-4}$

CLO # 1: Describe different techniques in making personalized recommendations in various scenarios.

[Estimated Time: *35 mins*, **15 points (7.5+7.5)]**

Question 2(a): Consider the following subset of anime recommendations dataset. Use item-based collaborative filtering with adjusted cosine similarity to predict the rating ***R*(U3,Gintama)**. Assume ***k*** = 2.

	Gintama	Inuyasha	MHA	Cowboy Bebop
User 1	2	1	5	4
User 2	4	5	2	
User 3		4		
User 4			3	2
User 5	3	2	2	1
User 6	5	3	1	5

Question 2(b): For the data given in Q2(a), calculate posterior probabilities **P (r=1|X)** and **P (r=2|X)** using user-based Naïve Bayes collaborative filtering. Assume **α** = 0.01 and **β** = 0.05.