UNIVERSITAT DE BARCELONA

Treball de Fi de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Revisió i anàlisi d'algorismes de recomanació basats en la sensibilitat de preu

---

Alumne: Marc Caminal Salinas

Directora: Dra. Maria Salamó
Realitzat a: Departament de Matemàtiques i Informàtica
Barcelona, 10 de juny de 2024

Abstracte

Amb el ràpid creixement del món de la informàtica, les empreses de venta de productes han decidit operar de forma digital, és a dir, venent els seus productes mitjançant pàgines web, el que avui es coneix com a comerç electrònic. A més a més l'ésser humà sempre ha optat per seguir l'opinió dels experts abans d'adquirir un producte. La compra on-line no permet la recomanació personalitzada per part d'un empleat com sí passa en una botiga física i és per aquest motiu que naixen els sistemes de recomanació.

En els darrers anys hi ha hagut un gran avanç en el món de la informàtica gràcies a l'aprenentatge automàtic. Això ens porta a investigar noves formes de realitzar les recomanacions, com pot ser estudiar els efectes del preu sobre la decisió d'un usuari abans de comprar un producte o la categoria d'aquest i considerar aquests elements per seleccionar les recomanacions que es mostren a l'usuari.

A la literatura han aparegut nous sistemes de recomanació que tenen en compte la sensibilitat de preu, els coneguts com a *price-aware recomenders*. El principal objectiu d'aquest projecte és realitzar una revisió i un anàlisi d'aquests algorismes de recomanació basats en la sensibilitat de preu. L'estudi consta de dos mètodes basats en la sensibilitat de preu i una sèrie d'algorismes que no ho fan servir que, juntament amb tres conjunts de dades ens ajudaran a saber sí la sensibilitat de preu és un element útil a considerar durant la recomanació.

Abstract

With the rapid advancement of the electronic world, companies have started selling their products online through what is known as e-commerce. Humans have also relied on the help of experts before deciding whether or not to buy a product. The combination of these two factors has resulted in the development of economic recommender systems.

Lately, there has been a significant step forward in the computer science world with the further development of Machine Learning. This phenomenon encourages us to investigate further ways to improve recommender systems, such as price sensitivity and category interest.

In the literature there has arisen a new wave of recommender sistems that incorporate price in their algorithms, known as price-aware recommenders. The main goal of this project is to analyze price-aware economic recommender systems. The project involves analyzing two price-based methods and comparing them to a set of non-price-aware methods from traditional literature using three datasets. The primary objective is to discover whether price sensitivity is an important element to include in economic recommender systems or if it does not make a significant difference.

Agraïments

Abans de continuar aquest projecte, m'agradaria destinar unes línies per agrair a la gent que ha fet possible la realització d'aquest. Primer de tot, donar les gràcies a la meva tutora, la Doctora Maria Salamó, a David Contreras i a Fernando Medina per l'ajuda aportada durant els mesos en els que s'ha realitzat el projecte i per guiar-me i orientar-me durant aquest temps. També agrair-los-hi el temps dedicat a les reunions cada dues setmanes per ajudar-me a resoldre dubtes.

M'agradaria agrair també a totes aquelles persones que d'alguna forma o altra han format part d'aquest projecte. Tots aquells investigadors que apareixen en la bibliografia i aquells que han destinat hores a investigar els recomanadors basats en la sensibilitat de preu i n'han desenvolupat de nous. La seva dedicació i el seu entusiasme vers la ciència es admirable i serveix un propòsit molt lloable cap al món on vivim.

Voldria agrair a la meva família, els meus amics i els companys de classe que han estat a la meva vora durant els quatre anys de carrera. Ara ja estic al final d'aquesta i, sense el seu suport i escalf, no ho hagués pogut aconseguir.

Índex

1 Introducció **1**
- 1.1 Àmbit del Projecte . . . 1
- 1.2 Descripció del Projecte . . . 2
- 1.3 Objectius del Projecte . . . 3
- 1.4 Planificació del Projecte . . . 3
- 1.5 Descripció de l'estructura . . . 4

2 Estat de l'Art **5**
- 2.1 Taxonomia dels sistemes de recomanació . . . 5
- 2.2 Economic recommender systems . . . 5
  - 2.2.1 Recomanadors basats en la sensibilitat de preu . . . 6
  - 2.2.2 Recomanadors basats en la utilitat econòmica. . . . 7
  - 2.2.3 Recomanadors basats en la consciencia dels beneficis . . . 7
  - 2.2.4 Recomanadors basats en mètodes promocionals . . . 8
  - 2.2.5 Métodes de sostenibilitat a llarg termini. . . . 9
- 2.3 Estudi sobre PASVD . . . 10
- 2.4 Estudi sobre PEDR . . . 12
- 2.5 Estudi sobre PUP . . . 15
- 2.6 Estudi sobre CoHHN . . . 18
- 2.7 Estudi sobre PASBR . . . 20

3 Anàlisi, disseny i implementació **24**
- 3.1 Llibreries . . . 24
- 3.2 Algorismes de recomanació . . . 24
  - 3.2.1 CoHHN . . . 25
  - 3.2.2 GC-SAN . . . 26
  - 3.2.3 GRU4Rec . . . 27
  - 3.2.4 LESSR . . . 28
  - 3.2.5 PASBR . . . 31
  - 3.2.6 KNN-Based . . . 33
- 3.3 Escenari experimental . . . 34
  - 3.3.1 Preguntes de recerca . . . 34
  - 3.3.2 Execució del codi . . . 34
- 3.4 Detall de les dades . . . 36

4 Anàlisi de resultats **37**
- 4.1 Mètriques d'avaluació . . . 37
- 4.2 Anàlisi comparatiu d'algorismes . . . 37
- 4.3 Discussió i Resum final . . . 42

5 Conclusions i Treball Futur **45**
- 5.1 Conclusions . . . 45
- 5.2 Treball Futur . . . 45

6 Bibliografia **47**

1 Introducció

Durant aquesta secció s'introduiran els conceptes necessaris per a entendre el projecte, així com l'estructura d'aquest i els objectius a complir al llarg de la seva realització.

1.1 Àmbit del Projecte

En el món del comerç electrònic és clau mantenir la satisfacció dels clients així com maximitzar els beneficis de l'empresa. Una de les vies per aconseguir aquests dos objectius és la recomanació. En aquest Treball de Fi de Grau (TFG) ens centrarem en l'estudi dels sistemes de recomanació. En la literatura ens trobem dos tipus de recomanadors. Per una banda ens trobem els recomanadors basats en contingut, que basen les recomanacions en els diferents productes que interessen a un usuari al llarg de les sessions, és a dir, recomanara productes de les mateixes categories dels productes amb que l'usuari ha interactuat de forma positiva. Per l'altra banda tenim els recomanadors basats en filtratge col·laboratiu, on les recomanacions es generen a partir de les similituds entre diferents usuaris, és a dir, si a dos usuari els hi agrada un mateix producte, es pot suposar que el que li agradi a un d'ells, li agradara al altre. En aquest projecte ens centrarem especialment en els recomanadors basats en la sensibilitat de preu, un tipus de sistema de recomanació basat en filtratge col·laboratiu que afegeix la variable del preu dels productes per a realitzar les prediccions.

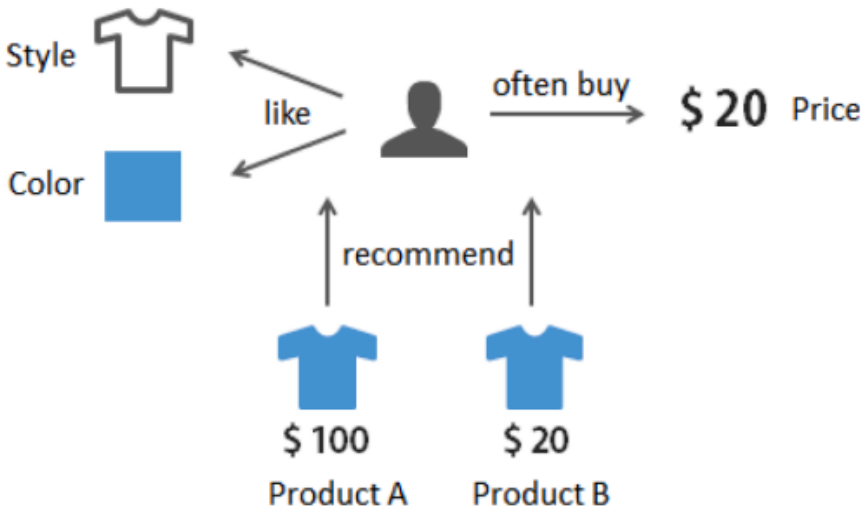

Figura 1: Exemple de recomanació

A la Figura 1 es pot observar com funciona un sistema de recomanació i com el preu és un factor important per a decidir una compra. És important remarcar que tots els recomanadors basats en la sensibilitat de preu són mètodes amb filtre basat en contingut, és a dir, recomanen en base a ítems similars i no a ítems que ha comprat gent amb gustos similars.

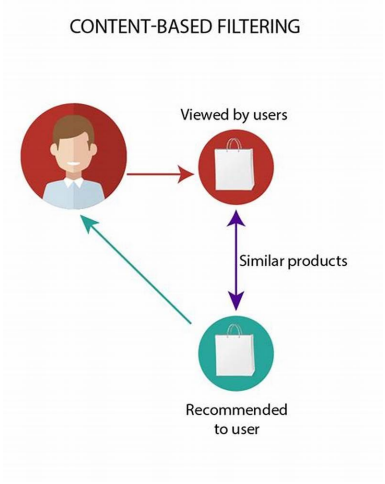

Figura 2: Recomanació basada en ítems

A la Figura 2 podem observar com funciona un sistema de recomanació basat en el filtratge col·laboratiu. Com ja s'ha explicat abans, si un usuari selecciona, compra o interactua amb un ítem, el sistema recomanara ítems de categories similars.

En aquest projecte s'ha decidit estudiar aquests sistemes de recomanació basats en preu ja que són molt nous, la majoria d'estudis que s'han trobat daten de l'any 2023, per tant, no ha donat temps a comparar realment aquests nous mètodes amb la resta de mètodes de la literatura tradicional de recomanació.

1.2 Descripció del Projecte

El projecte consisteix en tres parts ben diferenciades. La primera part és dedica a l'estudi del concepte de recomanació a nivell general així com els tipus de recomanadors que hi ha segons el seu objectiu o els paràmetres que utilitza per a generar les prediccions. Un cop definits els recomanadors i els seus tipus, ens centrem en l'estudi de noves propostes de recomanadors basats en la sensibilitat de preu.

La segona part consisteix en l'estudi extensiu de diferents mètodes de recomanació basats en preu estudiats en l'apartat anterior. Concretament, es compilaran els codis amb diferents conjunts de dades i es replicaran els resultats obtinguts pels creadors dels algorismes.

La tercera i última part consisteix en comparar una part d'aquests mètodes estudiats a la segona part i un conjunt d'algorismes que no utilitzen la sensibilitat de preu. Per a realitzar aquestes comparacions s'executaran els codis escollits amb els mateixos conjunts de dades i els paràmetres que maximitzin els seus resultats. Un cop obtinguts els resultats es discutiran

aquests per arribar a una conclusió final sobre els algorismes de recomanació basats en la sensibilitat de preu.

1.3 Objectius del Projecte

Amb l'objectiu d'assegurar l'èxit i la veracitat d'aquest projecte, s'ha plantejat un objectiu principal, l'anàlisi i comparativa dels algorismes de recomanació que usen la sensibilitat de preu i una sèrie d'objectius secundaris:

- Entendre els tipus de recomanadors que existeixen.
- Estudiar i definir els millors sistemes de recomanació actuals.
- Trobar el codi de dos sistemes de recomanació basts en preu per a estudiar més profundament
- Generar resultats per diferents conjunts de dades amb els sistemes de recomanació seleccionats
- Comparar els sistemes de recomanacions entre si amb el mateix conjunt de dades. Per aquest satisfer aquest objectiu es defineixen uns sub-objectius:
  - Adaptar els codis per a poder fer servir els mateixos conjunts de dades
  - Generar gràfiques per ajudar a entendre els resultats.
  - Analitzar els resultats obtinguts mitjançant els gràfics generats.

1.4 Planificació del Projecte

Mes	Febrer			Març				Abril				Maig				Juny
Setmana	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1. Investigació 1.1 Què son els recomanadors? 1.2 Tipus de recomanadors 1.3 Recomanadors basats en preu																
2. Implementació de codi 2.1 CoHHN 2.2 Baselines 2.3 PASBR																
3. Resultats i redacció de memòria 3.1 Anàlisi de resultats																

Taula 1: Diagrama de Gantt

Durant les primeres setmanes del TFG ens hem centrat en la fase 1, la lectura i la creació de resums de múltiples articles i estudis sobre els sistemes de recomanació i més concretament, dels sistemes de recomanació basats en la sensibilitat de preu. D'aquesta manera es pot entendre correctament els objectius de la recomanació, així com veure l'estructura dels estudis realitzats sobre cada recomanador per poder realitzar un estudi final en condicions.

Un cop llegits i resumits tots els articles proposats, arribem a la segona fase, la implementació de codi. Aquesta fase consisteix en buscar i descarregar el codi de diferents recomanadors estudiats anteriorment i comprovar els resultats amb diferents sets de dades. Una vegada acabades les proves amb els diferents algorismes, seleccionem uns conjunts de dades global per a analitzar i comparar els resultats de tots els recomanadors i així poder veure en que destaca cada un.

L'última fase i la més llarga, es la redacció de la memòria, la qual s'ha començat la primera setmana, amb els resums dels articles escollits a la fase 1, i ha seguit fins al final del projecte.

1.5 Descripció de l'estructura

1.Introducció: En el primer capítol s'exposa l'àmbit del projecte i mostra una breu descripció del tema del projecte. Aquest capítol també declara els objectius del projecte juntament amb el temps estimat per a cada tasca. Finalment es presenta un breu detall de cada capítol.

2. Investigació: Aquest capítol exposa la taxonomia sobre els recomanadors i es resumeixen els tipus de recomanadors i es justifica quins són els mètodes estudiats a fons en aquest projecte.

3. Implementació de codi: En aquest capitol s'exposen els mètodes escollits per a la demostració pràctica. S'expliquen les llibreries del projecte. S'analitza el codi dels mètodes seleccionats i s'introdueix l'entorn on s'han realitzat les proves.

4. Anàlisi de resultats: Aquest capítol inclou l'experimentació i es mostren els resultats obtinguts pels mètodes amb tres conjunts de dades. S'analitzen aquests resultats mitjançant una sèrie de gràfics.

5. Conclusions: En aquest capítol es realitza una breu reflexió sobre el projecte i possibles extensions de cara al futur.

2 Estat de l'Art

En aquest capítol es troba la taxonomia dels sistemes de recomanació amb la que relacionar els mètodes de l'estudi així com el context dels recomanadors i els resums dels mètodes estudiats.

2.1 Taxonomia dels sistemes de recomanació

Com es veu a la Figura 3, els recomanadors els dividirem en cinc grups segons el seu objectiu:

1. Basats en preu;
2. Basats en el modelatge econòmic útil;
3. Basats en la consciencia dels beneficis;
4. Basats en la promoció;
5. Basats en la sostenibilitat a llarg termini.

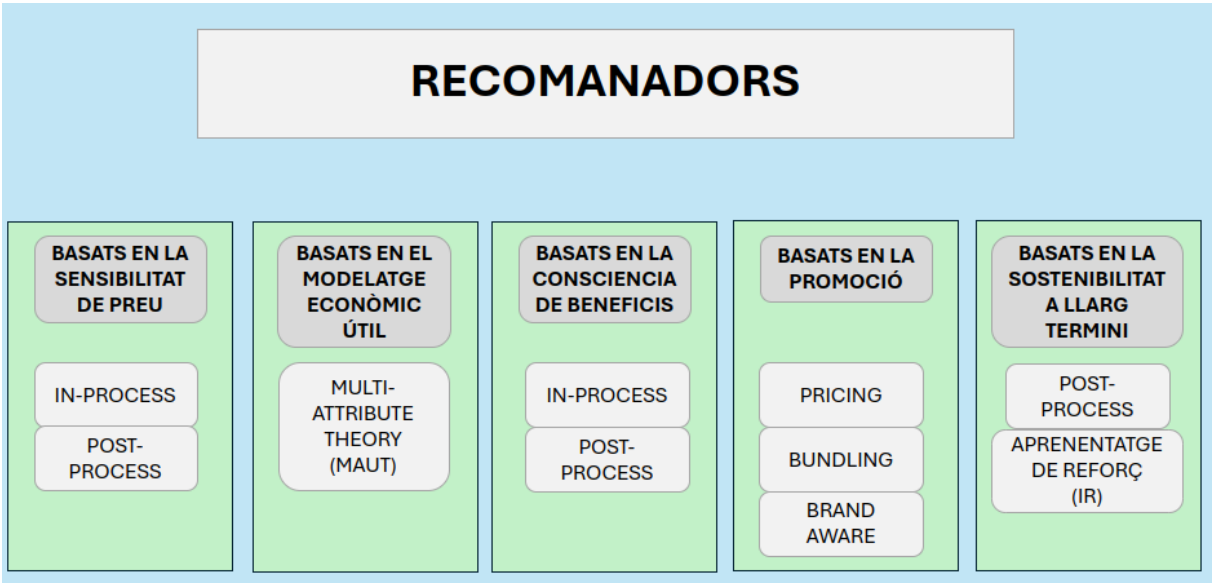

Figura 3: Taxonomia dels sistemes de recomanació

A continuació es detallen cadascun d'ells en la Secció 2.2.

2.2 Economic recommender systems

Un *Economic recomender system* (ECRS) [3] és un recomanador que aprofita la informació que té sobre el preu i els beneficis així com altres conceptes de màrqueting i economia per a maximitzar els beneficis de l'organització que els utilitza.

Els ECRSs es poden descomposar en diferents dimensions d'ànalisi (DAs). Concretament, s'ha identificat cinc tipus de propostes, les mostrades a la figura 3

Aquests 5 tipus de propostes es poden dividir en dos grans grups, orientats al consumidor i orientats a l'empresa. Els recomanadors orientats al consumidor busquen integrar els models dels RSs amb els mecanismes d'estudi del comportament de les compres per a generar recomanacions més rellevants i així augmentar el volum de vendes. Per la seva banda, els recomanadors orientats a l'empresa utilitzen estratègies especifiques de l'empresa per optimitzar els KPIs del negoci. La diferència principal és que els recomanadors orientats al consumidor busquen solucions a problemes des de la perspectiva del client i els orientats a l'empresa, busquen solucions a problemes des del punt de vista del negoci. Ara s'explica el raonament darrera de cada proposta.

- **DA1. La sensibilitat de preu** busca considerar explícitament les preferències de preu dels usuaris en la recomanació. Al considerar el preu als algorismes s'aconsegueixen recomanacions més precises i rellevants que comporten un augment en la probabilitat de compra i, per tant, un augment en els ingressos de l'empresa.
- **DA2. El modelatge econòmic útil** intenta considerar explícitament la utilitat de les recomanacions per a l'usuari en concordança d'un punt de vista econòmic. Generar recomanacions basades en el comportament utilitari d'un usuari pot augmentar el número de ventes.
- **DA3. La consciència dels beneficis** incorpora la informació dels beneficis als models de recomanació. Generar recomanacions amb un major benefici per a l'empresa comporta directament una optimització en els objectius econòmics d'aquesta.
- **DA4. La promoció** intenta generar recomanacions alhora que posa estratègica-ment els preus de certs productes centrant-se en l'antenció al client o certes promocions.
- **DA5. La sostenibilitat a llarg termini** intenta generar recomanacions tenint en compte la prespectiva econòmica a llarg termini. De fet, és un dels punts més importants per a una empresa.

Al final, tots els recomanadors estudiats busquen solucionar el problema de *top-k recommendation*. Per fer-ho consideren un set $\boldsymbol{U} = \{u_1, u_2, ..., u_m\}$ de m usuaris i $\boldsymbol{I} = \{i_1, i_2, ..., i_n\}$ de n ítems i una matriu d'interaccions usuari-ítem $\boldsymbol{X}$ on cada entrada $x_{u,i}$ representa la reacció de l'usuari u sobre l'ítem i. L'objectiu es trobar una funció de puntuació per a predir els valors de les entrades que manquen a la matriu $\boldsymbol{X}$. Com hem dit amb els cinc DAs, dividim els sistemes de recomanació en cinc tipus.

2.2.1 Recomanadors basats en la sensibilitat de preu

Una recomanació dins del rang de preu esperat per l'usuari augmenta les probabilitats de compra i per tant, més ventes i major benefici per a l'empresa. A la literatura s'han definit dues propostes, mètodes *In-process* i mètodes *Post-process*.

Mètodes *In-process*. Aquest tipus de mètodes estan basats en els algorismes *in-processing* on s'hi afegeix directament la sensibilitat del preu dins del model de recomanació. Particularment, s'ha trobat que és molt flexible en el model de la factorització de matrius (FM) la qual busca estimar l'interès d'un usuari cap a un cert producte usant el producte escalar de vectors de factors latents. El principal problema dels models clàssics és que generar recomanacions de productes en categories inexplorades pot causar una caiguda de rendiment fins a un 40%. En canvi, incorporar explícitament les preferències de preu d'un usuari pot comportar un augment de rendiment en aquestes recomanacions de categories desconegudes. Però, aquests models poden augmentar molt el seu rendiment alhora de recomanar productes amb un preu baix però no encertar alhora de recomanar productes amb un preu més alt, repercutint negativament a l'empresa. En aquesta categoria entren també els mètodes com el PUP [27], que combinen les preferències de preu amb les xarxes neuronals de gràfics (GNNs).

Mètodes *Post-process*. Aquesta idea consisteix en aplicar mètodes de ranqueig després d'un recomanador base. Es generen les recomanacions balancejant l'interès de l'usuari respecte al producte, així com la sensibilitat del preu.

2.2.2 Recomanadors basats en la utilitat econòmica.

Segons la teoria de la decisió racional, un usuari, quan se li presenten una sèrie d'opcions, escollira la que té una major utilitat per a ell o ella. En el camp dels sistemes de recomanació, s'assumeix que la utilitat $\rho_{u,i}$ d'un producte per a un usuari depèn de l'historial de compra d'aquest últim. En els actuals sistemes de recomanació, la utilitat d'una recomanació no és més que la suma de les puntuacions prevengudes $\rho_{i,u} = x_{i,u}(\Theta)$. També hi ha estudis que valoren l'ús de la *Multi-Attribute Utility Theory* (MAUT) [20] que consisteix en sospesar una sèrie de variables rellevants per tal d'analitzar cadascuna de les opcions.

Altres estudis es basen en el problema de recomanacions de productes ja comprats. En particular, s'estudia com una seqüencia de compres pot seguir la norma de *Law of Diminishing Marginal Utility* [1]. Aquesta teoria defensa que un tipus de producte disminueix la seva utilitat alhora que augmenta la quantitat de productes comprats, per exemple, els telèfons mòbils, però, n'hi ha d'altres que la utilitat augmenta quan més se'n compra, com podrien ser els bolquers de bebè.

2.2.3 Recomanadors basats en la consciencia dels beneficis

Aquest tipus de recomanador es basa en augmentar directament el benefici de l'empresa. En aquest tipus també s'han definit mètodes amb filosofia *In-process* i *Post-process*

Mètodes *In-process.* Alguns estudis proposen agafar els recomanadors per associació, *els usuaris que han comprat això també han comprat...* I optimitzar els beneficis de l'empresa, però aquesta idea no té en compte la personalització de les recomanacions i tots els usuaris rebran les mateixes recomanacions. Altres estudis proposen punts de vista basats en grafs, com les xarxes socials. També s'ha estudiat un recomanador basat en beneficis amb algorismes de filtratge col·laboratius [4], més concretament, ampliar el ja conegut criteri de selecció de veïns basat en usuaris del veí més proper. A més a més s'ha explorat solucions basant-se en el *Learn to Rate* (LTR) [16], una tècnica d'*Information Retrieval* (IR). En particular, l'algorisme proposa integrar el preu en la funció objectiu per a maximitzar els ingressos de l'empresa.

Mètodes Post-process. Tots els mètodes revisats en aquest apartat es basen en una suposició senzilla però important. Els ítems més rellevants per a l'usuari no sempre són els que més beneficis aporten a l'empresa. Per tant, prioritzar els productes amb un major marge de guanys millorarà els beneficis de l'empresa. Per aconseguir-ho, els interessos de les empreses i dels seus usuaris han d'estar balancejades, doncs hi ha el risc de perdre clients al no estar satisfets amb les recomanacions esbiaixades. Alguns estudis proposen aplicar un sistema de re-rànquing basat en el *Dice coefficient*, el qual genera recomanacions no massa diferents a les generades pel RS original basat en un límit η.

2.2.4 Recomanadors basats en mètodes promocionals

Aquests mètodes busquen augmentar les vendes promocionant productes i serveis als segments de clients més apropiats. S'identifiquen tres tipus, *Pricing*, *Bundling* i *Brand-awareness*.

Pricing Methods. Prèviament s'ha demostrat la importància del preu al moment de generar una recomanació. Però, fins ara havíem vist mètodes orientats al consumidor que integraven la sensibilitat del preu per a generar una recomanació més agradable per a l'usuari. En aquesta secció, en canvi, es tractaran una sèrie d'estratègies promocionals que una empresa pot aplicar per incentivar la compra de certs productes canviant-li els preus. Trobarem dues estratègies, *ocasional discounting* i *personalized dynamic pricing*.

Una de les estratègies més comunes es aplicar descomptes ocasionalment, com podria ser el *Black Friday* o les rebaixes de nadal. En els camps de RSs, s'intenta buscar la forma de recomanar productes tenint en compte el seu descompte a cada moment. Alguns estudis defensen l'ús de mètodes de re-rànquing per a promocionar productes en descompte, mentre d'altres, busquen considerar explícitament la sensibilitat als descomptes d'un usuari en mètodes basats en *Matrix Factoritzation* (MF) [13]. Dos últims estudis es centren en els efectes que té el descompte d'un producte en altres de la mateixa categoria a l'igual que d'altres de diferents categories. Un usuari comprarà productes en descompte i també, productes relacionats amb aquest però a preu sencer, per exemple, una càmera amb descompte però uns objectius sense.

Tot i que els descomptes ocasionals són iguals per a tots els usuaris, alguns mètodes proposen generar preus dinàmics per a cada usuari per així poder promocionar certs productes específics i augmentar els beneficis. Alguns estudis inicials proposen l'ús de *conjoint analysis* per estimar el *Willing-To-Pay* (WTP) de cada usuari per filtrar els productes superiors a aquest *WTP* i oferir-hi descomptes. Tot i haver-hi més mètodes, mai s'ha pogut provar l'efectivitat amb un gran número d'usuaris, per tant, aquests mètodes de *personalized dynamic pricing* són teòrics.

Bundling Methods. Una estratègia promocional bastant utilitzada és oferir descomptes en productes si es compren en lot. Per exemple, productes relacionats, productes completament diferents per així buidar estoc, o dos o més productes iguals, un 2x1 o un 3x2. Per estudiar aquests mètodes aplicats als RSs, ens centrarem en aquests que busquen explícitament optimitzar els KPIs del negoci.

Uns primers estudis [7] es centren en el desenvolupament d'un bot capaç d'oferir lots a preu rebaixat, però els experiments realitzats s'han fet amb una baixa quantitat de dades i sense tenir en compte els KPIs. Més estudis han proposat mètodes de programació per a recomanar lots que ajudin a assolir els objectius d'una empresa. El primer d'ells [12] proposa un sistema que, a mesura que un usuari afegeix productes al lot, calcula en temps real el descompte aplicable per a maximitzar els beneficis de l'empresa. El segon treball [28] busca estudiar la compatibilitat entre productes per tal de generar recomanacions que, un cop aplicat el descompte, maximitzin els beneficis de l'organització.

Brand-Awareness Methods. Alguns mètodes es poden usar per a promocionar els serveis i productes d'una empresa i així augmentar els beneficis a llarg termini. Aquests mètodes es basen en el *sales funnel*, un model teòric que descriu el camí d'un usuari en diferents moments. Amb això, podem suposar que es recomanable dissenyar un sistema de recomanació amb diferents propòsits segons l'estat. Si el client encara no ha fet cap compra, pot ser prometedor augmentar el ràting de conversió completant la primera compra el més ràpid possible. En aquest estat, no és idoni recomanar els productes més populars, ja que tard o d'hora, el client els acabaria trobant, però, seria millor recomanar productes, que si bé són populars, no estan tan relacionats per així ampliar el ventall de possibilitats que té el nostre client. Un cop el client ha fet la seva primera compra, l'empresa pot aplicar mecanismes per a maximitzar els beneficis a llarg termini. En aquest cas, també és important recomanar productes que, si bé han obtingut una bona puntuació en els rànquings, no són els més populars i segurament li siguin d'interès a l'usuari.

2.2.5 Métodes de sostenibilitat a llarg termini.

És molt important per a les empreses créixer de forma sostenible a llarg termini. Els algorismes de recomanació que consideren les dinàmiques temporals són els que més optimitzen

el valor del negoci a llarg termini. Molts d'ells es basen en el *Customer Lifetime Value* (CLV) el qual representa el valor del negoci esperat per tot el flux de diners per un sol usuari.

Mètodes de post-processament. Alguns estudis proposen aquests tipus d'algorismes per a maximitzar els beneficis explotant una heurística. Un estudi [10] proposa que, quan un client confia en un recomanador, aquest s'ha d'esbiaixar per tal de maximitzar els beneficis de l'organització, en canvi, quan no hi confia, s'ha de recomanar productes més populars per tal de recuperar aquesta confiança. Un altre punt de vista [11] es basa en els algorismes de *machine learning* per a optimitzar els beneficis a llarg termini. En particular, s'estudia el comportament de compres dels usuaris amb un CLV més alt i després, les recomanacions es donen intentant seguir aquests patrons de compra de la forma més fidel possible. Però hi ha un problema i és que de moment, només s'ha pogut testejar en simulacions.

Mètodes d'aprenentatge de reforç (RL). El RL és un enfocament d'aprenentatge que busca assolir una política òptima basada en una seqüència d'interaccions entre un usuari i el seu entorn per tal de maximitzar les recompenses. Un estudi [18] es basa en les transaccions de l'usuari, concretament, cada element de benefici es pot relacionar amb una acció de l'usuari, per tant, el benefici final es pot maximitzar optimitzant la suma d'aquestes accions considerant la probabilitat de que ocorrin segons les recomanacions.

A continuació es detallen els articles més importants de la literatura i que han estat d'estudi en profunditat en aquest treball de fi de grau.

2.3 Estudi sobre PASVD

En el camp del comerç electrònic, l'objectiu principal de les recomanacions és aconseguir que l'usuari compri el producte recomanat. En la majoria de casos, el preu és el factor determinant en la decisió de l'usuari sobre comprar o no el producte.

De forma general, els recomanadors tendeixen a proposar productes similars a les preferències del consumidor, però si no es considera el preu, és difícil que s'acabi venent l'article. Cada usuari té un rang de preu preferit a l'hora de valorar un producte, així com una sensibilitat dins d'aquest rang, per tant, per tal de millorar les recomanacions és necessari modelar tant la preferència com la sensibilitat del preu de forma personalitzada per a cada usuari.

Aquest primer article [23] proposa un recomanador basat en un model de factorització de matrius tenint el compte les preferències dels usuaris respecte al preu així com la resta de factors que afectin un producte, afegint el terme *user-price* al framework. A més a més, es tindrà en compte dos tipus de sensibilitats. La sensibilitat global, la qual mesurara la diferència en el preu preferit d'un usuari comparada amb la resta de consumidors, i la sensibilitat local, que buscarà reflexar la diferència en la preferència de preu en un cert rang de preus respecte a un altre rang diferent.

2.3.0.1 El model PASVD

El model *PASV*, de l'anglès, *Price aware singular value decomposition* consisteix en afegir preferències de preu sobre el ja existent *SVD*, que prediu els ràtings $\hat{r}_u i$ de la següent forma:

$$\hat{r}_u i = \mu + b_u + b_i + \mathbf{p}_u^T \cdot \mathbf{q}_i \tag{1}$$

El paràmetre μ indica la mitjana dels ràtings, els paràmetres b_u i b_i denoten la desviació observada de l'usuari u i l'ítem i sobre la mitjana, p_u és el vector d'usuaris i factors, i q_i és el vector d'ítems i factors.

A aquesta fórmula s'hi afegeix un terme b_{up} tal i com es veu a l'equació 2, que és la preferència de preu de l'usuari u. A més a més, afegim el pes w per a mesurar la sensibilitat de cada usuari i ajustar la preferència de forma personalitzada. Si w és gran, b_{up} tindrà més pes i viceversa. Per tant, la funció final per a predir el preu queda així:

$$\hat{r}_u i = \mu + b_u + b_i + w_{up} \cdot b_{up} + \mathbf{p}_u^T \cdot \mathbf{q}_i \tag{2}$$

El pes w conté dues parts, el pes global OS i el pes local LS per a cada usuari i es calcula de la següent forma:

$$w_{up} = 1 + \gamma \cdot OS_u + (1 - \gamma) \cdot LS_u \tag{3}$$

On

$$OS_u = \sqrt{\frac{1}{t} \sum_{p=1}^{t} (b_{up} - \bar{b_{up}})^2} \tag{4}$$

$$LS_u = b_{up} - b_{u,p-1} \tag{5}$$

La variable t és el número d'intervals en els que s'ha dividit els rangs de preu.

Finalment, la recomanació es pot obtenir ordenant els productes per la seva puntuació final i retornant els productes amb el valor més alt.

2.4 Estudi sobre PEDR

Diversos estudis com el realitzat per Kyung Young Lee [14] han trobat que, en el món del comerç electrònic, les preferències de preu d'un usuari a l'hora de comprar un producte canvien amb el temps, a més a més, la sensibilitat canvia en funció de l'últim element seleccionat. Els models tradicionals com el SVD, així com la resta de mètodes tradicionals que tenen en compte el preu, no tenen en compte el registre de compres per a analitzar i actualitzar les preferències dels usuaris. Per altra banda, els models basats en Deep-learning, tals com GRU4Rec [9] i BERT4Rec [22] aprenen de l'historial de compra dels usuaris, però no tenen en compte el preu dels productes a l'hora de realitzar les recomanacions.

En aquest article [8] es proposa un nou model basat en deep-learning anomenat *PEDR* de l'anglès *Price-aware Enhanced Dynamic Recommendation*, que recomana articles basat en algorismes de Deep-learning i el preu d'aquests.

2.4.0.1 El model PEDR

Suposem que tenim el conjunt de clients C = $\{c_1, c_2, ..., c_c\}$ i el conjunt de productes P = $\{p_1, p_2, ..., p_p\}$, el preu dels productes és L = $\{l_{p_1}, l_{p_2}, ..., l_{p_p}\}$, la seqüència de compres anteriors $S_C = \{p_1^c, p_2^c, ..., p_t^c\}$ on p_t^c és el producte que l'usuari c ha comprat en el temps t. El ràting que un client c li atorga a un producte p_t^c és $r_{p_t}^c$ i $\mathbf{R}_c$ és la seqüència de ràtings en el temps de c. Les reviews es representen com a $\mathbf{Re}_{p_t}^c$. El problema que es pretén solucionar és predir el producte p_{t+1}^c que el client c comprarà en el temps $(t+1)$.

Tal i com es mostra a la Figura 4, el *PEDR* es desenvolupa comprimint quatre capes. La capa de *generació* del producte, que genera una nova seqüencia considerant les preferències de preu dels usuaris expressades a les reviews. La capa de *codificació*, la qual converteix el producte, el preu i més informació en la forma correcta. La capa de *transformació*, s'encarrega d'explorar les correlacions dels usuaris amb els productes i els preus de forma separada. Per últim tenim la capa de *predicció*, que com bé diu el seu nom, s'ocupa de predir quin producte serà el següent en ser comprat.

La capa de generació rep com a paràmetres les reviews del consumidor així com els ratings d'una sèrie de productes i obtenim com a resultat una seqüència de productes amb el preu desitjat per l'usuari. S'utilitza un *price score extractor* per a extreure el preu de les reviews.

La puntuació del preu es calcula mitjançant tècniques de processament de llenguatge natural. Primer de tot s'usa el $Word2Vec$ [17] per aprendre dels texts dels usuaris. En segon lloc, es crea un diccionari amb frases i paraules que fan referència al preu. Es basen en el diccionari *Sentiwordnet* 3.0 i afegint-hi les frases i paraules relacionades amb el preu. Tercer, es decideix

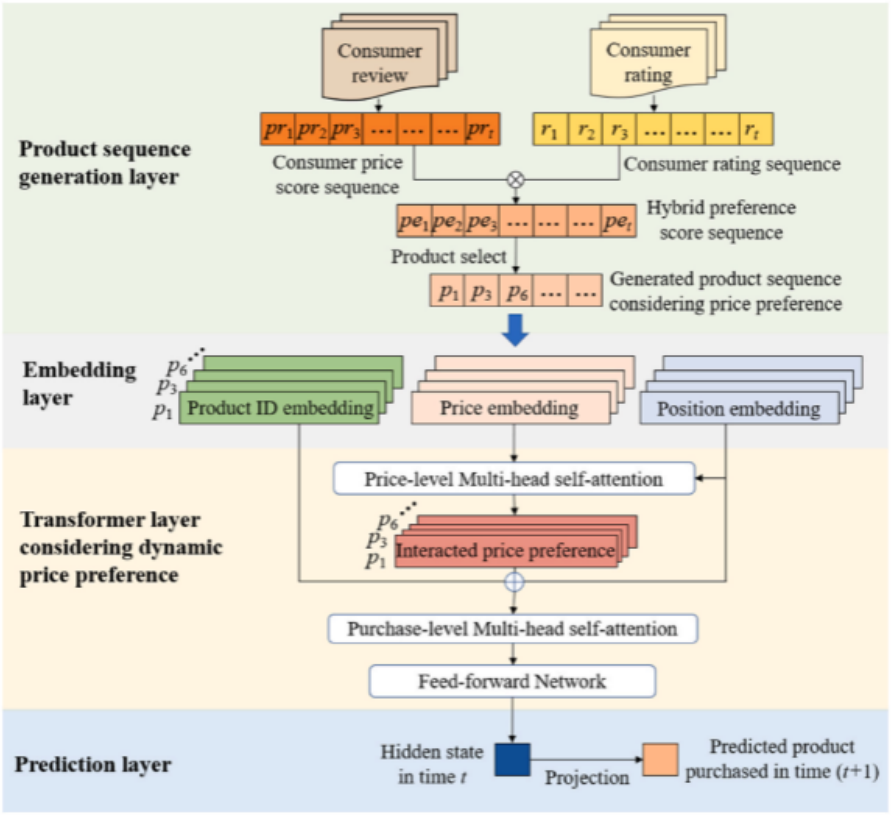

Figura 4: Arquitectura del model PEDR

si les frases d'una review tenen relació amb el preu usant el diccionari. Per acabar, si una frase té relació amb el preu, es calcula la seva polaritat mitjançant el mètode *NLTK* [15]. La puntuació va de 0 a 1, sent 0 el més baix i 1 el més alt.

Generador de seqüències basat en reviews i ràting. Es selecciona una seqüència basant-se en la puntuació del preu i les reviews. Primer es multiplica el ràting del consumidor y la puntuació del preu de l'usuari.

$$\mathbf{PE}_c = \mathbf{R}_c \cdot \mathbf{PR}_c \tag{6}$$

Després s'utilitza el *gating* per a filtrar les preferències i es mapejen per generar la seqüència com:

$$\mathbf{S}_c^G = mapping(\mathbf{S}_c, gating(\mathbf{PE}_c)) \tag{7}$$

La capa de conversió. En aquesta capa es divideix tot el rang de preu en diferents subrangs i es determina el seus preus. Posteriorment es fa l'embedding com:

$$\mathbf{ep}_t^c = emb(p_t^c) \tag{8}$$

$$\mathbf{el}_t^c = emb(l_t^c) \tag{9}$$

$$\mathbf{epos}_t^c = emb(POS_t^c) \tag{10}$$

La capa de transformació. En aquesta capa es desenvolupen dos mecanismes d'atenció multi nivell.
En el price-level multi-head self-attention primer ajuntem $\mathbf{el}_t^c$ per dimensió de la compra dins de $\mathbf{EL}c$ i després ajuntem $\mathbf{EL}c$ per dimensió de clients.

$$EL = [EL^1, EL^2, ..., EL^C]$$

Després s'aplica el mecanisme *price-level multi-head self-attention*:

$$\mathbf{Q}_n^1 = \mathbf{W}_{q_n}^1(\mathbf{EL} + \mathbf{EPOS}) \tag{11}$$

$$\mathbf{K}_n^1 = \mathbf{W}_{kn}^1(\mathbf{EL} + \mathbf{EPOS}) \tag{12}$$

$$\mathbf{V}_n^1 = \mathbf{W}_{vn}^1\mathbf{EL} \tag{13}$$

$$\mathbf{head}_n^1 = \sigma\left(\frac{(\mathbf{Q}_n^1)^T\mathbf{K}_n^1}{\sqrt{\frac{d}{nh_1}}}\right)\mathbf{V}_n^1 \tag{14}$$

$$\mathbf{PP} = concate(\mathbf{head}_1^1, \mathbf{head}_2^1, ..., \mathbf{head}_{nh1}^1) \tag{15}$$

En el *purchase-level multi-head self-attention* també ajuntem l'embedding dels consumidors i els concatenem per obtenir la informació del producte

$$\mathbf{PI} = concate(\mathbf{EP}, \mathbf{PP}, \mathbf{EPOS}) \tag{16}$$

Després estudiem les preferències dinàmiques com:

$$\mathbf{Q}_n^2 = \mathbf{W}_{q_n}^2\mathbf{PI} \tag{17}$$

$$\mathbf{K}_n^2 = \mathbf{W}_{kn}^2\mathbf{PI} \tag{18}$$

$$\mathbf{V}_n^1 = \mathbf{W}_{vn}^2\mathbf{PI} \tag{19}$$

$$\mathbf{head}_n^2 = \sigma\left(\frac{(\mathbf{Q}_n^2)^T\mathbf{K}_n^2}{\sqrt{\frac{d}{nh_2}}}\right)\mathbf{V}_n^2 \tag{20}$$

$$\mathbf{PUP} = concate(\mathbf{head}_1^2, \mathbf{head}_2^2, ..., \mathbf{head}_{nh1}^2 \tag{21}$$

Finalment apliquem una *fast-forward network* (FFN) per dotar el model de nolinearitat. Es calcula com:

$$\mathbf{A}^1 = LN(\mathbf{PUP} + Drop(Dense(\mathbf{PUP}))) \tag{22}$$

$$\mathbf{A}^2 = GELU(Dense(\mathbf{A}^2)) \tag{23}$$

$$\mathbf{H} = LN(\mathbf{A}^1 + Drop(Dense(\mathbf{A}^2)) \tag{24}$$

on $\mathbf{A}^1$ i $\mathbf{A}^2$ són estats intermedis i H és l'estat ocult que representa les preferències dels consumidors sobre els productes.

La capa de predicció. Aquí es preveu que comprarà l'usuari en temps (t+1) i es guarda en $\mathbf{PRO}_c^{(t+1)}$ que és la probabilitat que hi ha de que el consumidor compri un producte en temps (t+1).

$$\mathbf{PRO}_c^{(t+1)} = softmax(\mathbf{H}_c^t(\mathbf{E}^T) + \mathbf{b}) \tag{25}$$

Entrenament y tests Per a entrenar el model, s'ha decidit emmascarar uns certs productes de la seqüència amb l'objectiu de predir els productes emmascarats amb el *PEDR*. En el testing també es prediu els productes i s'utilitza el clàssic optimitzador d'Adam i la fórmula de pèrdua *cross-entropy*.

$$l^c = -\sum_{j=1}^{P} z_j^c \log(y_c^j) + (1 - z_j^c) \log(1 - y_c^j) \tag{26}$$

2.5 Estudi sobre PUP

En la majoria de casos del comerç electrònic un usuari només comprarà un producte si li és d'interès i el preu és acceptable. Generalment hi ha dues dificultats a l'hora d'afegir el preu a un sistema de recomanació:

- **Consciència de preu no escrita.** Un usuari defineix la seva preferència i sensibilitat de preu de forma única per a cada producte. Per tant s'ha d'obtenir de forma personalitzada aquests valors basant-se en l'historial de cada consumidor. Encara més difícil, s'ha de considerar l'efecte del filtres col·laboratius als historials d'usuaris similars.
- **Dependència de la categoria.** Cada usuari tindrà una diferent sensibilitat i preferència de preu segons la categoria del producte. Un esportista, per exemple, tindrà més tolerància en el preu a l'hora de comprar material esportiu que al comprar productes de neteja.

En aquest article [27] es proposa una solució per aquests dos problemes, el **P**rice-aware **U**ser **P**reference-modeling (PUP) que utilitza les xarxes de grafs convolucionals per aprendre.

2.5.0.1 El problema

Per a entendre la inconsistència de la sensibilitat entre categories, s'estén el ja conegut preu que accepta pagar un usuari, *willing to pay* (WTP) per el preu que accepta pagar un usuari per cada categoria *category willing to pay* (CWTP).

Per aquest problema tenim U i I que són la llista d'usuaris i ítems respectivament, la matriu d'utilitat R_{MxN} on M i N són el nombre d'usuaris i productes. $R_{ui} = 1$ on R implica que l'usuari u ha comprat una vegada l'article i. S'utilitza $\mathbf{p} = \{p_1, p_2, ...p_N\}$ i $\mathbf{c} = \{c_1, c_2, ...c_N\}$ per a definir el preu i la categoria dels ítems. Es divideix el preu de cada categoria de productes en diferents rangs.

D'entrada el model rep R i els preus $\mathbf{p}$ i les categories $\mathbf{c}$. De sortida retorna la probabilitat estimada de comprar per una parella usuari-ítem (u,i).

2.5.0.2 El model PUP

S'ha dissenyat el model *PUP* amb un graf heterogeni unificat, un graf codificador convolucional i un descodificador basat en parelles.

Graf heterogeni unificat. Les dades interactives d'entrada i els atributs, preu i categoria es poden representar amb un graf no dirigit, *G=(V,E)*. Els nodes de V són els nodes d'usuaris $u \in U$, els nodes d'ítems $i \in I$, els vèrtex de categories, $c \in \mathbf{c}$ i els vèrtex de preus $p \in \mathbf{p}$.

Graf codificador convolucional. En aquesta part s'estén el tradicional LFM per aprendre les representacions dels quatre tipus d'entitats. S'utilitzen tres capes.

Embedding Layer. En aquesta capa es comprimeixen les IDs dels nodes en un vector de valors reals més dens. $e' = \mathbf{R}^d$ on d és la mida del embedding.

Embedding Propagation Layer. En aquest codificador, aquesta capa captura el missatge transferit entre dos nodes connectats directament.

$$\mathbf{t}_{ji} = \frac{1}{|N_i|} e'_j, \tag{27}$$

on N_i és la llista de nodes veïns.

Neighbour Aggregation Layer. En aquest codificador s'usa l'agrupació per mitjana i una funció d'activació no lineal per a passar el missatge. La norma de modificació s'expressa

com:

$$
\begin{aligned}
o_u &= \sum_{j \in \{i \text{ with } R_{ui}=1\} \cup \{u\}} \mathbf{t}_{ju}, \\
o_i &= \sum_{u \in \{i \text{ with } R_{ui}=1\} \cup \{i, \mathbf{c}_i, \mathbf{p}_i\}} \mathbf{t}_{ji}, \\
o_c &= \sum_{j \in \{i \text{ with } \mathbf{c}_i=c\} \cup \{c\}} \mathbf{t}_{jc}, \\
o_p &= \sum_{j \in \{i \text{ with } \mathbf{p}_i=p\} \cup \{c\}} \mathbf{t}_{jp}, \\
e_f &= tanh(o_f), f \in \{u, i, c, p\}.
\end{aligned}
\tag{28}
$$

Des d'un punt de vista de recomanació, aquest codificador pot capturar la similitud entre dos nodes si hi ha un camí entre ells.

Pairwise-interaction Based Decoder. Es fa servir un descodificador seguint l'estil de les màquines de factorització. Seguint la mateixa nomenclatura del codificador anterior, les prediccions de la probabilitat de compra s'expressen com:

$$
\begin{aligned}
s &= s_{global} + \alpha s_{category} \\
s_{global} &= e_u^T e_i + e_u^T e_p + e_i^T e_p \\
s_{category} &= e_u^T e_c + e_u^T e_p + e_c^T e_p
\end{aligned}
\tag{29}
$$

Entrenament. S'usa un auto-codificador semi-supervisat de grafs i una fórmula de pèrdua basada en el Ranking Personalitat Bayesià (BPR).

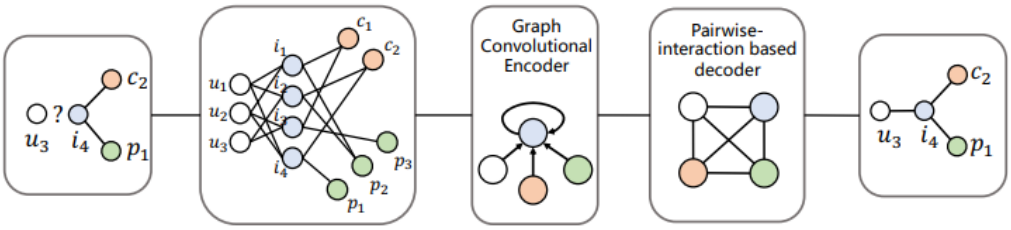

Figura 5: Arquitectura del model PUP

A la Figura 5 es pot observar un gràfic amb l'arquitectura del model *PUP*, des de la creació del graf, pasant pels codificadors fins a la predicció.

2.6 Estudi sobre CoHHN

Els recomanadors actuals es centren en modelar els interessos de l'usuari, per exemple, que tant li interessa una categoria en concret utilitzant xarxes neuronals recurrents (RNN), xarxes d'atenció i xarxes de grafs neuronals (GNN). Cap d'aquests casos inclou un factor important, les preferències de preu dels usuaris, que ens indica fins a quin preu esta disposat a pagar un usuari per un ítem. Nous estudis demostren que el comportament de compra d'un usuari està molt afectat pel preu d'aquest.

2.6.0.1 El problema

L'objectiu és modelar les preferències de preus i interessos per a millorar els recomanadors basats en sessions. Considerem I la llista d'ítems disponibles i $|I| = \mathbf{n}$ el nombre total d'ítems. Un ítem $x_i \in I$ conté el ID, el preu i la categoria. $\boldsymbol{S} = [x_1, x_2, ..., x_m]$ és una sessió generada per un usuari aleatori on m és la longitud de $\boldsymbol{S}$. La funció proposada rep com a entrada és l'actual sessió $\boldsymbol{S}$ i la sortida es $\boldsymbol{y} = \int(\boldsymbol{S}) = [y_1, y_2, ..., y_n]$ on $y_j (j \in [1, ..., n]$ és la probabilitat de que l'ítem x_j sigui el proper a ser comprat per l'usuari.

El preu es discretitza en ρ nivells on la probabilitat corresponent a cada interval es igual. Es calcula com:

$$\rho_i = \frac{\phi(x_\rho) - \phi(min)}{\phi(max) - \phi(min)} \otimes \rho \tag{30}$$

on $\phi(x)$ és la distribució acumulada de la funció de distribució logística, definida com:

$$\phi(x) = P(X \leq x) = \frac{1}{1 + e^{-\pi \frac{x-\mu}{\sqrt{3}\delta}}} \tag{31}$$

on μ i δ són el valor esperat i la desviació estàndard respectivament.

2.6.0.2 El Model CoHHN.

El model proposat CoHHN [26] està basat en els hypergrafs heterogenis. S'usa un canal doble d'agregació per a propagar la informació pels nodes en dos canals, per dins i entre ells. S'implementa una capa d'atenció per a extreure les preferències de preus i interessos de l'usuari. Un sistema d'aprenentatge co-guiat s'aplica per a modelar les relacions entre les dades obtingudes. Finalment es realitza la predicció.

Mecanisme d'agregació de dos canals. Un node pot tenir molts tipus de nodes adjacents diferents. S'utilitzen dos canals per dins i entre nodes per a donar diferents nivells d'importància als nodes adjacents.

Intra-type Aggregation. $\mathbf{v}_t \in R^d$ és la incrustació d'un node amb tipus t. Els seus nodes adjacents formen un set N_t^τ. L'objectiu d'aquest mecanisme d'agregació és diferenciar la importància de diferents nodes per a representar el tipus (τ). La representació és:

$$\mathbf{e}_t^\tau = \sum_i a_i \mathbf{v}_t^\tau \tag{32}$$

$$a_i = \frac{exp(\mathbf{u}_\tau \mathbf{v} i^\tau)}{\sum_{\mathbf{v}_t^\tau \in N_t^\tau} exp(\mathbf{u}_\tau \mathbf{v} i^\tau)} \tag{33}$$

on $\mathbf{u}_\tau^T \in R^d$ es el vector d'atenció que representa la importància d'un node. Podem abstreure el proces per definir-lo com a la funció:

$$\mathbf{e}_t^\tau = f_a(N_t^\tau) \tag{34}$$

Inter-type Aggregation. Com que els diferents tipus provenen informació heterogènia per al node objectiu. Aquest mètode junta la informació a nivell de característiques de la següent forma:

$$\begin{aligned} g_t &= \sigma(\mathbf{W}_t[\mathbf{v}^t; \mathbf{e}_t^{\tau 1}; \mathbf{e}_t^{\tau 2}] + \mathbf{W}_t^{\tau 1}\mathbf{e}_t^{\tau 1} + \mathbf{W}_t^{\tau 2}\mathbf{e}_t^{\tau 2}) \\ \mathbf{h}^t &= \mathbf{v}^t + g_t * \mathbf{e}_t^{\tau 1} + (1 - g_t) * \mathbf{e}_t^{\tau 2}) \end{aligned} \tag{35}$$

El procés aquest es pot definir de la següent forma:

$$\mathbf{h}^{id} = f_b(\mathbf{v}^{id}, f_a(N_{id}^p f_a(N_{id}^c)) + avg(N_{id}^{id}) \tag{36}$$

$$\mathbf{h}^p = f_b(\mathbf{v}^p, f_a(N_p^{id}, f_a(N_p^c)) \tag{37}$$

$$\mathbf{h}^c = f_b(\mathbf{v}^c, f_a(N_c^p, f_a(N_c^{id})) \tag{38}$$

Extracció de preferències Consisteix en extreure les preferències de preu i interessos.

Preferències de preu. Aquestes es poden obtenir suposant que els preus preferits de l'usuari són els que hi ha al seu historial de compres. Utilitzen un sistema *self-attention* amb un sistema *multi-head* per a modelar les preferències de preu de la següent forma:

$$\begin{aligned} \mathbf{E}_p &= [h_1^p, h_2^p, ..., h_m^p] \\ head_i &= Attention(W_i^Q \mathbf{E}p, W_i^K \mathbf{E}_p, W_i^V \mathbf{E}_p) \\ \mathbf{S}_p &= [head_1; head_2; ...; head_h] \end{aligned} \tag{39}$$

Preferències d'interessos. Aquestes preferències varien amb el temps, per tant, s'ha de tenir en compte la posició d'aquestes. Es calculen de la següent manera:

$$\begin{aligned} \mathbf{v}_i^* &= tanh(\mathbf{W}_f[\mathbf{h}_i^{id}; \mathbf{pos}_i] + \mathbf{b}_f) \\ \hat{\mathbf{I}}_p &= \sum_{i=1}^{t} \beta_i \mathbf{h}_i^{id} \\ \beta_i &= \mathbf{u}\sigma(\mathbf{A}_1 \mathbf{v}_i^* + \mathbf{A}_2 \overline{\mathbf{v}}^* + \mathbf{b}) \end{aligned} \tag{40}$$

2.7 Estudi sobre PASBR

Els mètodes tradicionals de *SBR* cauen en el patró dels mètodes basats en extracció i en la cadena de Markov, es a dir, basen les prediccions en l'ultima acció de l'usuari enlloc de tenir en compte l'ordre dels ítems dins de cada sessió. Un altre problema dels mètodes basats en la cadena de Markov és la forta suposició de la dependència condicional entre dos termes adjacents. A més a més, els models tradicionals basats en les xarxes de grafs neuronals també tenen pèrdua d'informació a l'hora de crear els grafs. El model *PASBR* [5], de l'anglès, *Price-aware Session-based Recommendation* es proposa solucionar aquests problemes.

2.7.0.1 El Mètode PASBR

Tenim $I = \{i_1, i_2, ..., i_m\}$ sent m el nombre d'ítems que tenim i $|I| = m$ es el total d'ítems. Cada ítem $i_j \in Ij \in [1, 2, ..., m]$ té la corresponent categoria $c_j^{cg} \in C^{cg}$ i un preu $e_j^{pr} \in E^{pr}$ on $C^{cg} = \{c_1^{cg}, c_2^{cg}, ..., c_{|c^{cg}|}^{cg}\}$ és el conjunt de categories i $E^{pr} = \{e_1^{pr}, e_2^{pr}, ..., e_{|e^{pr}|}^{pr}\}$ és el conjunt de preus. *PASBR* té com a objectiu predir el següent ítem de la seqüència de compra calculant la probabilitat de cada ítem.

El PASBR consisteix en tres mòduls, entrada, extracció de característiques i predicció. Com a entrades, el PASBR inclou les categories, els preus i els ítems d'una sessió, així com la pròpia sessió. El mòdul d'extracció de característiques es divideix en sis capes, la capa de representació gràfica de la sessió (SGRL), la capa de preu (PAL), la capa de representació de les intencions de l'usuari (UIRL), la capa d'interès a curt termini (STIRL), la capa d'interès a llarg termini (LTIRL) i la capa de representació de l'interès dinàmic (DIRL). El mòdul de predicció ajunta tot el que s'ha aprés i genera una predicció.

Input Module. Aquest mòdul busca aprendre la configuració inicial de la sessió. S'obte mitjançant una *2-layer Faceforward Neural Network* (FNN).

Feature Extraction Module. Com ja s'ha dit prèviament, aquest mòdul consta de sis capes, SGRL, PAL, UIRL, STIRL, LTIRL i DIRL.

SGRL. Aquesta capa genera els grafs de sessions d'acord a les posicions dels ítems a les sessions i aprèn la representació dels nodes. Quan es construeixen grafs de sessions, la majoria de mètodes tenen una pèrdua de dades important i, restaurar-les un cop generats els grafs, és impossible. Per evitar-ho es construeixen dos tipus de grafs sense pèrdua, ***S**ession to **M**ulti-**G**raph* (S2MG) i ***S**ession to **S**hortcut **G**raph* (S2SG). La representació dels nodes es realitza a les capes EOPA i SGAT de forma independent. Les capes *EOPA* i *SGAT* són apilades i el resultat de l'última capa es representa com $X^V \in R^{(2L+1)d}$.

PAL. Aquesta capa busca aprendre com influencia el preu als diferents ítems de les diferents categories. Per a saber-ho, primer s'ajunta la categoria i el preu d'un ítem i després la funció resultat es codifica mitjançant el ***M****ultilayer* ***P****erceptron* (MLP). La forma de realitzar aquest procès és mitjançant la suma (add).

$$\rho_i = \sigma(W_{pr}(q_i^{cg} + p_i^{pr})) \tag{41}$$

UIRL. Aquesta capa busca modelar les preferències dels usuaris per a cada categoria. La seqüència de transferència del patró entre nodes es captura mitjançant GRU i es calcula com:

$$h_i^{gc} = GRU(q_i^{cg}, h_{i-1}^{cg}) \tag{42}$$

STIRL. La representació de l'últim ítem amb el que ha interactuat l'usuari es guarda com a l'interés a curt termini de l'usuari $s_{[}short]$).

LTIRL. Aquesta capa s'encarrega de modelar els gustos de l'usuari a llarg termini. En la literatura tradicional dels recomanadors, s'agafa amb més pes l'últim ítem de la seqüència i amb això es calcula la suma de pesos per a trobar els interessos d'un usuari. Aquesta forma de generar els interessos pot portar problemes, doncs si un usuari selecciona un ítem que no li interessa per error, aquest ítem introduiria molt soroll al càlcul. Per evitar aquest problema *PASBR* té en compte la intenció de cada ítem s_{intent} per a calcular els interessos de la següent forma:

$$\begin{aligned} \eta_{in} &= q_{agg}^T \sigma(W_1 X_i^V + W_2 X_n^V + W_3 s_{intent} + b_{agg}) \\ a = softmax(\eta) &= \frac{\eta_{jn}}{\sum_{i\in[1,2,\ldots,n]} \eta_{in}} \\ s_{long} &= \sum_{i=1}^{n} a_{in} X_i^V \end{aligned} \tag{43}$$

DIRL. Aquesta capa s'encarrega de modelar els interessos dinàmics d'un usuari. Es important tenir en compte que els interessos dels usuaris varien amb al temps, i tot haver vist en la capa anterior que no sempre és millor adjudicar un pes major als últims elements amb els que ha interactuat l'usuari, en aquesta capa i seguint els estudis sobre *SBR* si que es farà així.

$$\begin{aligned} \phi_{in} &= exp(|t_i - t_n| + log_2(t_n + 1)) \\ \gamma &= softmax(\phi) \\ S_{interest} &= \sum_{i=1}^{n} \gamma_{in} X_i^V \end{aligned} \tag{44}$$

Prediction Module. Un cop passat pel mòdul d'extracció de característiques toca realitzar les prediccions tenint en compte totes les característiques extretes.
La representació global de l'usuari es calcula com:

$$s_{global} = W_{global}([||s_{short}||s_{long}||s_{intent}||s_{interest}|]) \tag{45}$$

on W_{global} és la matriu de paràmetres d'aprenentatge. Posteriorment calculem el següent vector que usarem per a calcular les prediccions finals:

$$\hat{\gamma}_i = softmax(s_{global}^T x_j \tag{46}$$

I la predicció serà el resultat de:

$$L(\gamma, \hat{\gamma} = \sum_{j=1}^{m} \gamma_j log(\hat{\gamma}_j) \tag{47}$$

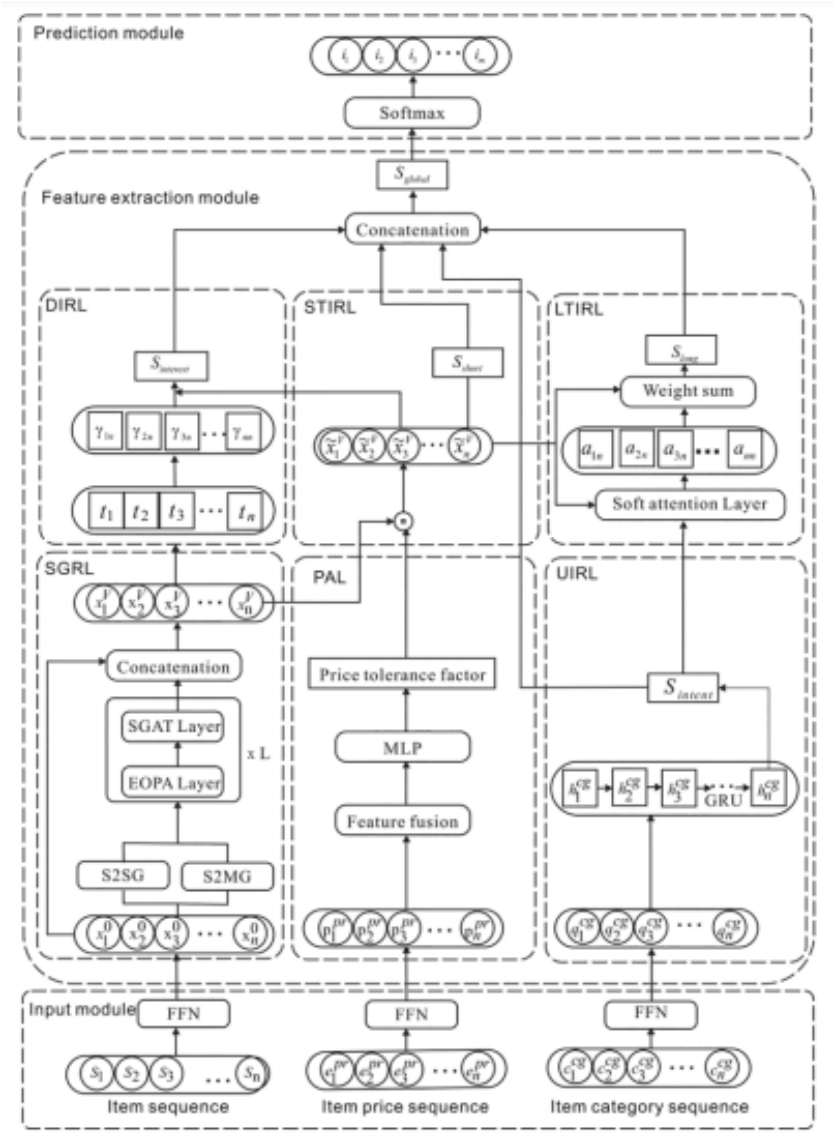

Figura 6: Arquitectura del model PASBR

La Figura 6, que ocupa la següent pàgina, mostra com funciona l'arquitectura del *PASBR*. Podem veure com el mòdul d'entrada, com separa el preu, la categoria i la seqüència d'ítems i les redirigeix a les diferents capes del mòdul d'extracció de característiques. Les característiques extretes posteriorment es concatenen i s'extrau S_{global} i s'envia al mòdul de predicció per a generar les prediccions.

3 Anàlisi, disseny i implementació

En aquesta secció es mostraran els codis seleccionats per al seu estudi així com els canvis realitzats a dits codis per a poder executar-los en un entorn de Google Collaborate[1] així com Paperspace[2]. Tots els codis estan en llenguatge Python.

3.1 Llibreries

A continuació s'expliquen les cinc llibreries més importants a tenir en compte, doncs són necessàries per a la implementació de tots els codis estudiats.

- ***NumPy*:** Llibreria de *Python* dedicada al càlcul numèric i anàlisi de dades, doncs ens permet crear vectors i matrius de gran dimensió. A més a més, incorpora funcions matemàtiques d'alt nivell per a realitzar càlculs sobre les estructures creades. Gràcies a aquestes operacions i noves estructures, es poden realitzar operacions de forma molt més ràpida que amb *Python* base.
- ***Pandas*:** És una llibreria de codi obert que busca ser l'eina fonamental de creació de blocs d'alt nivell per a l'anàlisi de dades en *Python*. Ens permet carregar dades, manipular-les i analitzar-les tot en una sola llibreria.
- ***PyTorch*:** Es tracta d'una llibreria de codi obert basada en *Torch* i molt usada en aplicacions de visió artificial i processament de llenguatge natural. Ens permet accelerar la computació de tensors (com NumPy) mitjançant l'ús de les unitats de processament gràfic (GPU).
- ***TensorFlow*:** És una aplicació de codi obert desenvolupada per Google que permet construir i entrenar xarxes neuronals. Tot i poder ser usat en *Python*, l'aplicació s'executa en *C++* d'alt nivell.
- ***CUDA*:** Són les sigles de *(Compute Unified Device Architecture)* desenvolupada per *NVIDIA* que ens permet explotar les avantatges de les seves GPU utilitzant el paral·lelisme que ofereixen els seus múltiples nuclis, permetent el llançament d'un alt número de fils simultanis.

3.2 Algorismes de recomanació

Per a fer l'estudi s'ha seleccionat una sèrie d'algorismes de recomanació. Tot seguit s'introduiran els mètodes escollits, explicant, tant a nivell teòric com pràctic, com funcionen els algorismes i quin problema intenten solucionar. L'ordre serà el següent; *CoHHN [26], GCSAN [6], KNN-based [21], GRU4Rec [9] LESSR [2]* i *PASBR* [5]. Cal recordar que *CoHHN* i *PASBR* son mètodes basats en preu mentre que la resta son mètodes més tradicionals.

1. Google Colab
2. Paperspace

3.2.1 CoHHN

Com ja s'ha explicat al resum de l'estudi sobre aquest mètode, l'objectiu és modelar les preferències de preus i interessos per a millorar els recomanadors basats en sessions. La funció proposada rep com a entrada és l'actual sessió i la sortida és la probabilitat de que un ítem sigui el proper a ser comprat per l'usuari. A més a més les preferències dels usuaris varien amb el temps i utilitzen les posicions integrades que han aprés per a solucionar aquest detall i així poder obtenir millors prediccions.

El model proposat CoHHN està basat en els hypergrafs heterogenis. S'usa un canal doble d'agregació per a propagar la informació pels nodes en dos canals, *Inter-type* i *Intra-type*.

3.2.1.1 Construcció del model

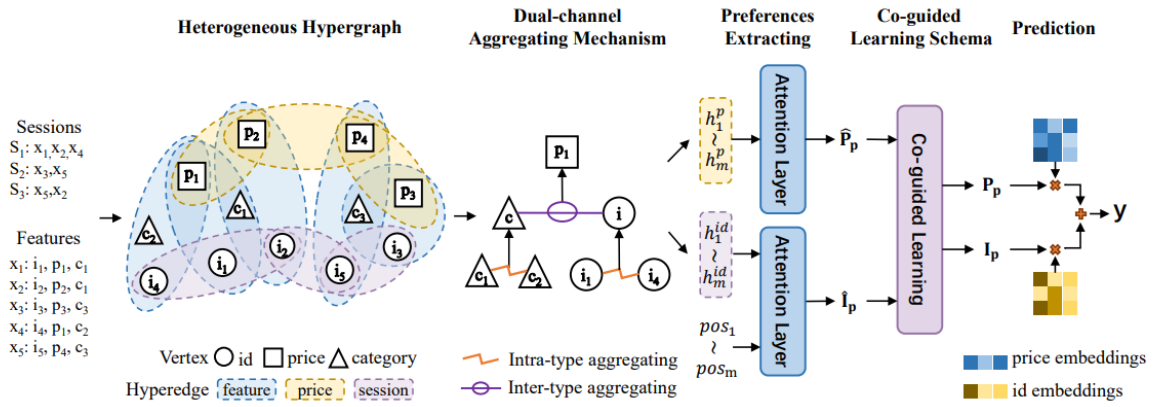

Figura 7: Flux del model CoHHN

En la Figura 7 s'explica el flux de treball del CoHHN. Primer es genera un hipergraf heterogeni basat en totes les sessions. A continuació, s'utilitza el sistema d'agregació de dos canals per a compartir la informació entre els diferents nodes. Posteriorment s'extreuen les preferències dels usuaris basades en preu i les hiperarestes dels nodes. Després es passa per un esquema d'aprenentatge co-guiat per a modelar les relacions entre les preferències extretes i, finalment, es prediuen els resultats.

A la Figura 8 observem el procés de dividir les dades en dos conjunts per entrenar i testejar el model, així com la creació del model. També es pot observar com es guarda el model mitjançant la funció *save_model()* i com es carrega amb *load_model()*.

```
train_data = Data(train_data, shuffle=True, n_node=n_node, n_price=n_price, n_category=n_category)
test_data = Data(test_data, shuffle=True, n_node=n_node, n_price=n_price, n_category=n_category)
model = trans_to_cuda(CoHHN(adjacency=train_data.adjacency,adjacency_pv=train_data.adjacency_pv,adjacency_vp=train_data.adjacency_vp) ,
optimizer = torch.optim.SGD(model.parameters(), lr=0.001)
model, optimizer, epoch1 = load_model(model, optimizer, save_path)

top_K = [1, 5, 10, 20]
best_results = {}
for K in top_K:
    best_results['epoch%d' % K] = [0, 0, 0]
    best_results['metric%d' % K] = [0, 0, 0]

for epoch in range(opt.epoch):
    print('-------------------------------------------------------')
    print('epoch: ', epoch1)
    metrics, total_loss = train_test(model, train_data, test_data)
    save_model(model, optimizer, epoch1, save_path)
```

Figura 8: Creació del model amb train i test

3.2.2 GC-SAN

Estudis recents sobre la recomanació basada en sessions han trobat que els mètodes que utilitzen ***S**elf-**A**ttention **N**etwork* (SAN) han obtingut un èxit considerable en les prediccions sense usar xarxes convolucionals o recurrents. *SAN* però no té en compte les relacions que existeixen entre ítems adjacents i per tant limita la capacitat d'aprenentatge. Per això es proposa un mètode anomenat ***G**raph **C**ontextualized **S**elf-**A**ttention **N**etwork* (GC-SAN).

En aquest model [6], com es mostra a la Figura 9 es construeix un estructura dinàmica per cada seqüència de sessions i s'agafen les dependències locals importants usant una *Graph Neural Network* (GNN). Després, cada sessió aprèn aquestes dependències amb un mecanisme d'autoaprenentatge. Finalment les sessions es representen amb una combinació lineal de les preferències globals i els interessos actuals de la sessió.

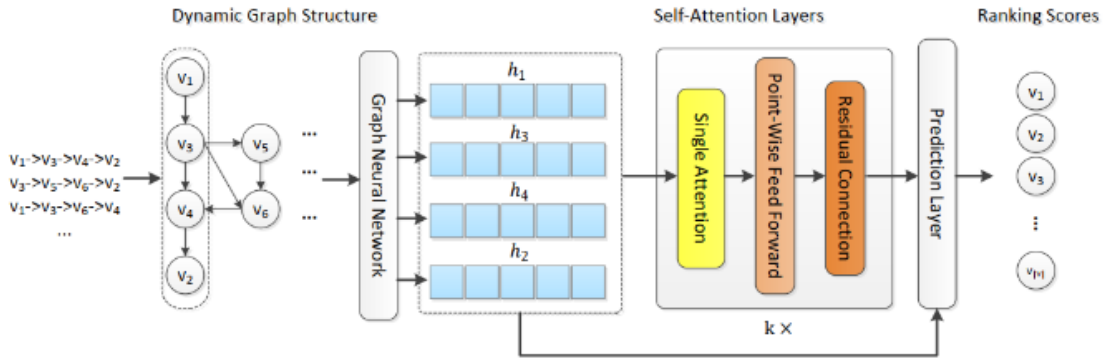

Figura 9: Flux del model GC-SAN

En el flux es pot observar com primer es crea una estructura dinàmica per a representar totes les sessions. Posteriorment les sessions passen per una *GNN* per tal d'obtenir els vectors de cada node de les sessions. Posteriorment passa per les *Self-Attention Layers* on es captura les dependències entre els ítems de la sessió i es realitza l'autoaprenentatge. Per últim a la capa de predicció es realitzen les prediccions i finalment es computen els rànquings de cada ítem.

3.2.2.1 El model

El model, com ja hem vist al flux, consta de tres parts ben diferenciades. La creació de l'estructura dinàmica del graf, les capes d'auto-atenció i la capa de predicció.

En la creació dinàmica del graf, primer cal construir una *GNN* de totes les sessions. Cada ítem de la sessió serà un node i cada interacció d'un usuari amb un ítem des d'un altre serà un vèrtex. Si tenim la sessió $S = [s_1, s_2, ..., s_n]$, s_i serà un ítem i $(s_{i-1}, s_i$ serà un vèrtex. El pes de cada vèrtex, serà el nombre de vegades que apareix l'ítem en la sessió. Posteriorment, s'han d'anar actualitzant els vectors dels nodes. És fa de la següent forma:

$$\begin{aligned} textbf{z}_t &= \sigma(\mathbf{W}_z \mathbf{a}_t + \mathbf{P}_z \mathbf{s}_{t-1}), \\ \mathbf{r}_t &= \sigma(\mathbf{W}_r \mathbf{a}_t + \mathbf{P}_r \mathbf{s}_{t-1}), \\ \hat{\mathbf{h}}_t &= tanh(\mathbf{W}_h \mathbf{a}_t + \mathbf{P}_h(\mathbf{r}_t \odot \mathbf{s}_{t-1})), \\ textbf{h}_t &= (1 - \mathbf{z}_t) \odot \mathbf{s}_{t-1} + \mathbf{z}_t \odot \hat{\mathbf{h}}_t. \end{aligned} \tag{48}$$

La segona part són les capes d'auto-atenció. Aquesta part consta de tres seccions, la capa d'atenció, la xarxa *Point-Wise Feed-Forward* i més capes d'atenció.

La primera capa d'atenció s'encarrega d'obtenir els vectors latents de cada node i extreure les preferències globals. En la xarxa *Point-Wise Feed-Forward* s'apliquen dos transformacions lineals amb la funció d'activació *ReLU* per considerar les relacions entre diferents dimensions latents. Per últim s'apliquen dues o més capes d'auto-atenció que, al igual que la primera, extreuen les preferències globals.

La tercera part és la capa predictora. Aquesta utilitza una barreja dels interessos a llarg termini que han extret les capes anteriors amb l'interès immediat de la sessió. D'aquesta manera obtenen millors resultats.

3.2.3 GRU4Rec

Aquest algorisme [9] utilitza *GRU-based RNN* per als recomanadors basats en sessions. La principal diferència entre els mètodes convencionals i els que usen *RNN* és l'existència d'un estat intern amagat dins de les unitats que formen la xarxa. Aquests mètodes però tenen un problema, la desaparició del gradient. Aquest problema es crea quan els gradients usats per a actualitzar la xarxa neuronal acaben reduint-se molt o "desapareixent"al quedar-se només a les primeres capes enlloc de les últimes i per tant, les que afecten al resultat final.

Un ***G**ated **R**ecurrent **U**nit* (GRU) és un variació més elaborada del model *RNN* amb l'objectiu de solucionar aquest problema del gradient. Per solucionar-ho, el que es fa és actualitzar aquest estat intern amagat cada cert temps.

Tal com es mostra a la Figura 10, en el centre de la xarxa és on es troben les capes *GRU*, just després de les capes necessàries per a convertir els vectors. A més a més, després de les capes de *GRU* es poden afegir diverses capes de retroacció abans de la capa final del resultat. El resultat obtingut serà una llista d'ítems amb les probabilitats de que siguin el següent ítem de la sessió. Si s'utilitzen múltiples capes de *GRU*, l'entrada d'una serà l'estat intern amagat de l'anterior. A més a més l'entrada de la primera capa *GRU*, els vectors matemàtics, pot ser enviada també a la resta de capes *GRU* per a millorar els resultats.

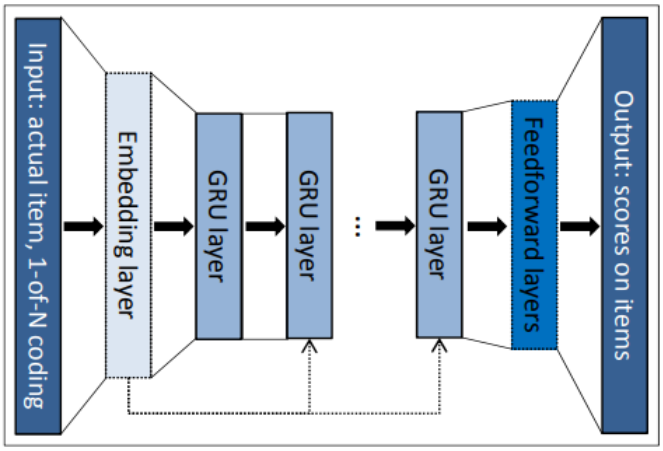

Figura 10: Flux del model GRU4Rec amb múltiples capes

3.2.3.1 Mostreig del resultat

Calcular les puntuacions de tots els ítems d'una web pot ser una tasca molt costosa, pràcticament insostenible. Per tant, aquest model només calcula les puntuacions d'un petit subconjunt de dades i per tant, només una part dels pesos dels ítems seran actualitzats. A més a més, també s'ha de tenir en compte els ítems amb els que l'usuari no ha interaccionat alhora de calcular les puntuacions, doncs si un usuari ha vist un producte i no ha interactuat amb aquest, implica que no li interessa. Quan més popular és un producte, més probable es que l'usuari l'hagi vist, per tant, a part d'actualitzar les puntuacions i els pesos dels productes amb els que l'usuari ha interactuat, també és calculen les puntuacions d'alguns d'aquests productes més populars sense interacció per part de l'usuari.

3.2.4 LESSR

Com a tots els recomanadors, l'objectiu és trobar quin ítem és el més probable de ser seleccionat per un usuari, en aquest cas, donat una seqüència d'ítems anteriors. En concret,

LESSR [2] utilitza ***G**raph **N**eural **N**etworks* (GNN) per aconseguir-ho. Això comporta un problema, doncs les *GNN* tenen pèrdua d'informació i per tant, el model proposat ha d'aconseguir utilitzar les *GNN* però sense perdre informació.

3.2.4.1 El model

Com es veu a la Figura 11 el primer que s'ha de fer, abans d'aplicar el model sobre les sessions, és convertir-les a grafs, per així poder processar-les amb *GNN*. En concret, s'utilitzaran dues capes, la primera, anomenada ***E**dge **O**rder **P**reserving Agregation* (EOPa) *layer* aplica el mètode *S2MG* per a convertir les sessions en un multi-graf que preserva l'orde dels vèrtexs. Aquest utilitza *GRU* per agregar la informació entre nodes.

La següent capa també s'utilitza per a preparar els grafs abans d'entrar model en sí. La capa ***S**horcut **G**raph **AT**tention* (SGAT) ens permet capturar les relacions entre nodes que no són veïns, és a dir, que la distància entre ells es major a 1. D'aquesta manera es pot tenir en compte més d'un ítem a l'hora de realitzar l'aprenentatge.

A partir d'aquí hi ha una seqüència de capes *EOP* i *SGAT* intercalades doncs les capes *EOP* són molt precises per a recaptar informació local i les capes *SGAT* ho són per a informació global. A més a més, *SGAT* és una conversió amb pèrdua, però **EOP** no, i si hi haguessin moltes capes *SGAT* consecutives, el model perdria molta informació.

Un cop s'han acabat les capes anteriors, toca convertir els nodes de les sessions a vectors matemàtics per a poder realitzar les prediccions. Finalment es passa per la capa de predicció i es recomana els ítems amb la probablitat *top-K*.

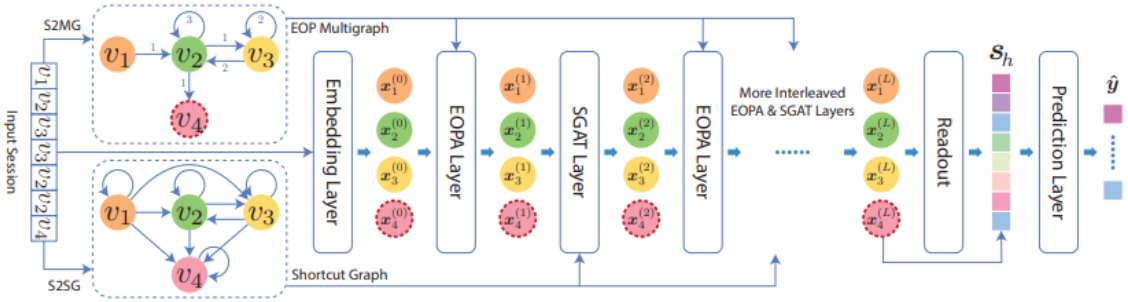

Figura 11: Flux del model LESSR

3.2.4.2 Mètodes de conversió a grafs

Com bé s'ha comentat prèviament, per a processar les sessions, aquestes han de ser convertides a grafs. En aquesta secció veurem més profundament com funcionen *S2MG* [24] i *S2SG* [19] que el model *LESSR* [2] utilitza per a fer les conversions.

En la literatura de recomanació tenim el tradicional mètode *S2WG*, veure Figura 12 (a), el qual converteix les sessions en grafs amb pesos. Aquest mètode té un problema alhora de convertir el tipus de sessions que utilitzem com a entrada per a aquest mètode, doncs tots els pesos seran 1 ja que la distància entre les sessions és 1.

Per a solucionar aquest problema tenim els mètodes *S2MG* i *S2SG* que s'apliquen a les capes *EOP* i *SGAT*.

- **EOP:** Aquest mètode converteix les sessions en un graf amb pesos, però hi ha una diferència respecte al mètode *S2WG*. En aquest procés no s'utilitza el pes segons la distància de les sessions, sinó que el pes utilitzat per a cada aresta és el nombre de vegades que es repeteix la sessió a l'entrada.
- **SGAT:** A la capa *SGAT* s'utilitza el *S2SG* per a crear un nou graf on cada ítem de la sessió és un node i cada aresta són les transicions recíproques entre nodes. A més a més, aquest mètode crea un graf sense ítems intermediaris i amb bucles a cada node, de tal forma que després el *SGAT* pot passar la informació entre tots els nodes.

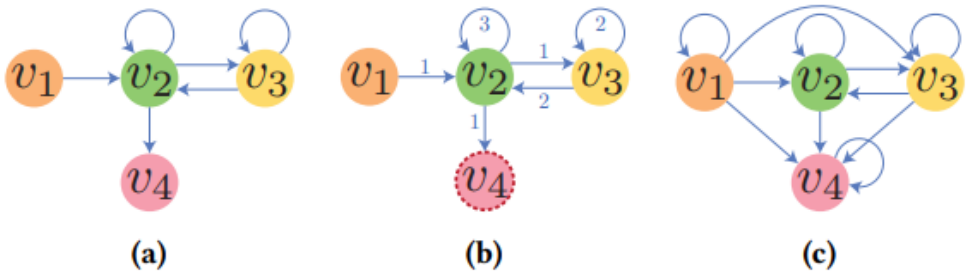

Figura 12: En la imatge (a) veiem un graf convertit amb S2WG, en la (b) esta convertit amb S2MG i (c) amb S2SG

En la Figura 12 podem veure els resultats de convertir sessions amb els diferents mètodes. Cal observar que els pesos del primer graf, el convertit per *S2WG* no apareixen doncs en tots els casos són 1 i, per tant, s'han suprimit.

3.2.4.3 Revisió de codi

Ara veurem els detalls més importants del codi. A diferència d'altres mètodes, aquest codi no només necessita els fitxers amb les dades per a l'entrenament i test, sinó que necessita un fitxer amb el número d'ítems.

A més a més el codi requereix de la llibreria *DGL* de python, la qual ens ofereix un sistema de control versàtil de missatges entre nodes així com una optimització de velocitat i un entrenament multi-GPU/CPU per a escalar grafs més grans. Aquesta llibreria no s'ha mencionat a l'apartat 3.1 doncs no s'utilitza en la resta de models.

A la Figura 13 podem observar com es creen les capes *EOP* i *SGAT* al codi. Es veu com s'intercalen les capes mentre no s'hagi arribat al màxim de capes especificades a l'inici del mètode, doncs anem iterant fins a aquell número i si toca parell es crea una capa *EOP* i si toca senar es crea una capa *SGAT*. Un cop aplicades les capes ja passem a la part de predicció.

```
for i in range(num_layers):
    if i % 2 == 0:
        layer = EOPA(
            input_dim,
            embedding_dim,
            batch_norm=batch_norm,
            feat_drop=feat_drop,
            activation=nn.PReLU(embedding_dim),
        )
    else:
        layer = SGAT(
            input_dim,
            embedding_dim,
            embedding_dim,
            batch_norm=batch_norm,
            feat_drop=feat_drop,
            activation=nn.PReLU(embedding_dim),
        )
    input_dim += embedding_dim
    self.layers.append(layer)
```

Figura 13: Creació de capes LESSR

3.2.5 PASBR

Els mètodes tradicionals de *SBR* cauen en el patró dels mètodes basats en extracció i en la cadena de Markov, és a dir, basen les prediccions en l'última acció de l'usuari enlloc de tenir en compte l'ordre dels ítems dins de cada sessió. Un altre problema dels mètodes basats en la cadena de Markov és la forta suposició de la dependència condicional entre dos termes adjacents.

PASBR [5] també intenta solucionar els problemes que sorgeixen a l'aplicar mètodes basats en *GNN*, com pot ser la pèrdua d'informació a l'hora de crear els grafs, o no tenir en compte les categories i preus dels productes, o no tenir en compte les preferències i els interessos dinàmics dels usuaris, així com les seves intencions.

3.2.5.1 El model

Com ja s'ha explicat en la secció anteriorment en la secció de l'estat de l'art, *PASBR* consta de tres mòduls, mòdul d'entrada, mòdul d'extracció de característiques i el mòdul de predicció, veure la Figura 5.

A continuació veurem més detalladament les capes de l'algorisme així com la forma en la que *PASBR* utilitza les capes *EOP* i *SGAT* que hem vist anteriorment en l'algorisme *LESSR*.

Primer parlarem del mòdul d'entrada que simplement busca aprendre la codificació inicial de les sessions i els seus preus i categories. S'obtè mitjançant una *2-layer* ***F****eedforward* ***N****eural* ***N****etwork* (FFN). En el mòdul d'extracció de característiques tenim 6 capes diferents, *SGRL, PAL, UIRL, STIRL, LTIRL* i *DIRL*. La capa *SGRL* s'encarrega de construir els grafs de sessions. Per a construir els grafs sense pèrdua, utilitza, com ja hem dit, els mètodes ***S****ession to* ***M****ulti****G****raph* (S2MG) i ***S****ession to Shortcut* ***G****raph* (S2SG), o el que es el mateix, les capes *EOP* i *SGAT* de LESSR però amb mètodes diferents per a aprendre les característiques.

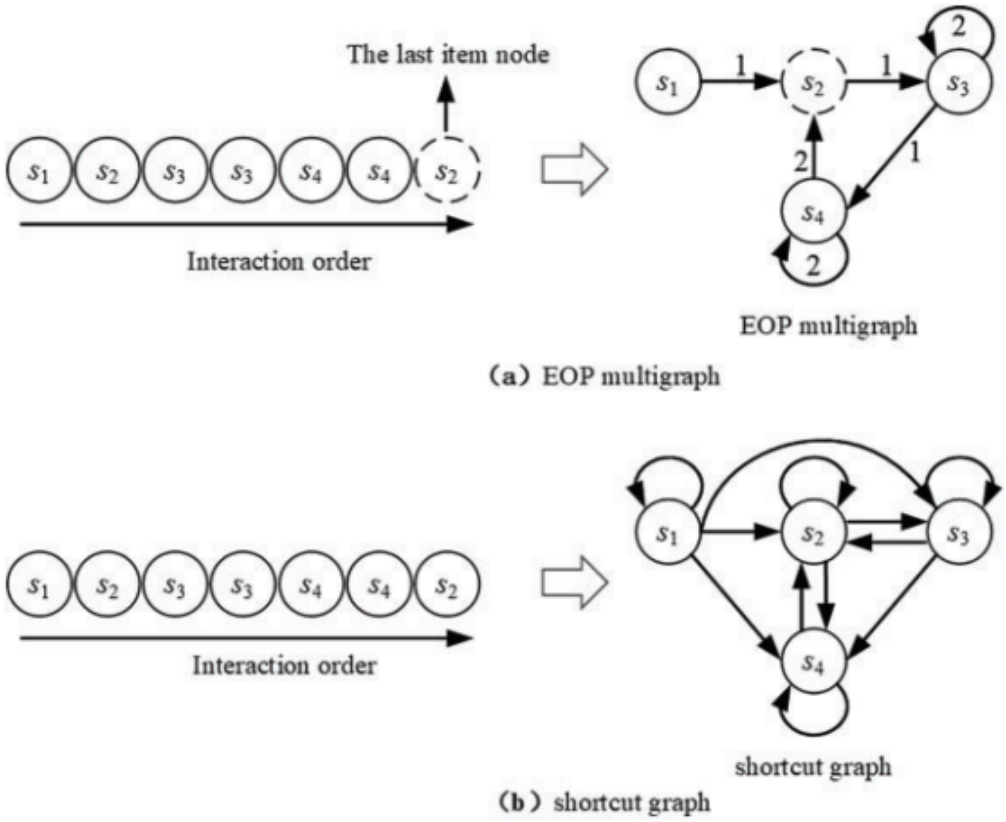

Figura 14: Grafs EOP i SGAT de PASBR

La capa *PAL* busca aprendre el factor preu a les accions dels usuaris. Ho fa fusionant la codificació del preu amb la codificació de les categories utilitzant el ***M**ulti**l**ayer **P**erceptron* (MLP) i la funció *add*.

La capa *UIRL* busca modelar les intencions de l'usuari a nivell categòric. El patró entre categories d'una sessió es captura mitjançant *GRU*, tal i com es pot veure al mètode *GRU4Rec* on l'estat amagat inicial es un vector de zeros.

La capa *STIRL* simplement s'encarrega de capturar la representació de l'últim ítem de la sessió amb el que ha interactuat l'usuari.

La capa *LTIRL* s'encarrega de capturar els interessos a llarg termini de l'usuari a l'hora que considera la intenció de l'usuari a l'hora de seleccionar un ítem i així evitar que un possible error quan es selecciona un ítem pugui desvirtuar la predicció.

L'última capa abans de la predicció és la *DIRL* la qual s'encarrega de capturar els interessos dinàmics de l'usuari.

Finalment la capa de predicció s'encarrega de retornar un vector *One-Hot* amb la probabilitat distribuïda del següent ítem.

3.2.6 KNN-Based

Per aquest mètode hem seleccionat una de les múltiples aplicacions existents que utilitzen la tècnica dels *K-Nearest-Neighbours* (KNN) [21] per a establir les relacions entre elements i poder predir quin serà el següent ítem a ser seleccionat.En concret, el mètode escollit utilitza la similitud del cosinus per a atorgar les puntuacions.

3.2.6.1 KNN

KNN és un mètode d'aprenentatge supervisat que estima el valor de la distància per decidir si un ítem pertany a un conjunt. En aquest cas, que un ítem sigui escollit per l'usuari i entri al conjunt de sessions. k és el número de veïns a estudiar. Com es pot observar, és un algorisme molt senzill, i a més a més, donarà molt pes als últims elements de la sessió.

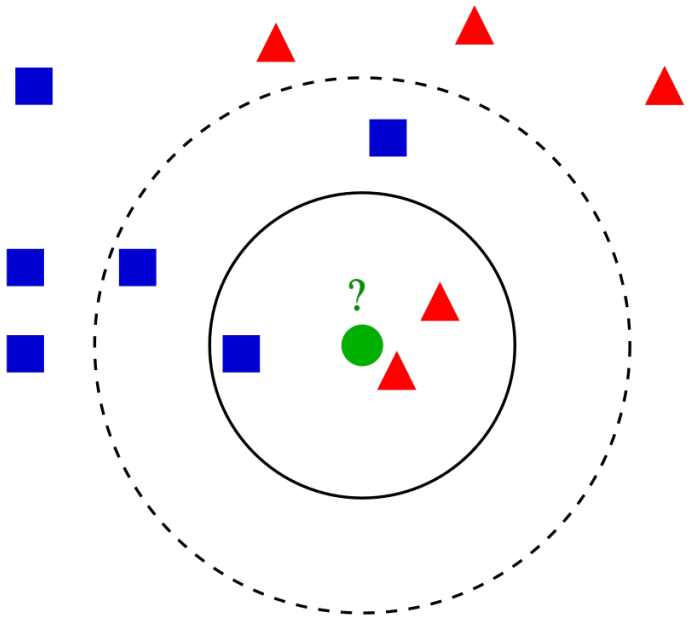

Figura 15: Exemple de KNN

En la Figura 15 es mostra com funciona l'algorisme *KNN* per a classificar el punt verd. Per a k=3 buscarem tres veïns més propers segons la distancia i, per tant, classificarà al punt verd com a triangle vermell, en canvi, per a k=5 sortirà un quadrat blau.

3.3 Escenari experimental

En aquest apartat es detalla l'escenari experimental, indicant les preguntes de recerca, com s'ha executat el codi i quines dades s'usaran en els experiments.

3.3.1 Preguntes de recerca

L'objectiu dels diferents experiments serà respondre a una sèrie de preguntes:

- **P1:** Quin model obté millors resultats?
- **P2:** Com afecten els paràmetres dels models als resultats?

3.3.2 Execució del codi

L'execució d'aquests codis s'ha realitzat en tres plataformes diferents, per una banda tenim el *Paperspace*[2], que ens permet executar proves utilitzant diferents màquines GPU. Per altra

1. Google Colab
2. Paperspace

banda, *Google Colab*[1], plataforma la qual ens permet crear notebooks de *Python* i executar-los amb GPU. Les dues plataformes tenen recursos limitats i, si bé hi ha alguns mètodes que s'han pogut executar sense cap problema, en alguns no es podia córrer més d'una iteració. Per solucionar aquest problema s'ha modificat els mètodes de tal manera que, un cop acabada una iteració, es guardi el model resultant com un *checkpoint* que poguem carregar i executar una nova prova a partir d'aquell punt, tal i com es mostra a la Figura 16.

```
def save_model(model, optimizer, epoch, save_path):
    state = {
        'epoch': epoch,
        'model_state_dict': model.state_dict(),
        'optimizer_state_dict': optimizer.state_dict(),
    }
    torch.save(state, save_path)

def load_model(model, optimizer, checkpoint_path):
    checkpoint = torch.load(checkpoint_path)
    model.load_state_dict(checkpoint['model_state_dict'])
    optimizer.load_state_dict(checkpoint['optimizer_state_dict'])
    epoch = checkpoint['epoch']
    return model, optimizer, epoch
```

Figura 16: Mètodes per a guardar i carregar el model

La tercera era un servidor situat a la universitat Arturo Prat de Xile, que ens han deixat una màquina amb una GPU senzilla però que ens permetia fer execucions durant més temps.

Com ja s'ha mencionat anteriorment, tots els codis estudiats es troben en llenguatge *Python*. Els codis es troben en arxius *.py* de *Python* i emmagatzemats en un paquet tots junts, és a dir, no formen part d'una llibreria.

A continuació es detallen els mètodes comparats en el projecte, els quals es dividiran en dos grups:
Grup I: Mètodes basats en la sensibilitat de preu:

- **CoHHN:** Utilitza hipergrafs heterogènis co-guiats per a codificar les dades. Incorpora la sensibilitat de preu per a realitzar les prediccions.
- **PASBR:** Soluciona el problema de la pèrdua d'informació de les *GNN* mitjançant les capes *SGAT* i *EOP*. Incorpora la sensibilitat de preu i els efectes dinàmics d'aquesta a curt i llarg termini

Grup II: Mètodes de la literatura tradicional:

- **KNN-Based:** Recomana ítems similars als últims de la sessió.

- **GC_SAN:** Combina *GNN* amb mecanismes d'auto-atenció per capturar les dependències globals i locals entre ítems
- **GRU4Rec:** Utilitza *GRU* per a trobar patrons en les sessions.
- **LESSR:** Soluciona el problema de la pèrdua d'informació de les *GNN* mitjançant les capes *SGAT* i *EOP*.

3.4 Detall de les dades

Per a realitzar les proves s'han utilitzat tres conjunts de dades, *Amazon*, *Diginetica* i *Cosmetics*.

Amazon[1] és un conjunt de dades molt utilitzat per a realitzar proves en el món de la recomanació. Conté diferents comportaments de compra d'una sèrie d'usuaris de la plataforma *Amazon*. Concretament agafem un subset anomenat *Grocery and Gourmet Food*.

Diginetica[2] és un conjunt de dades de comerç electrònic que conté el comportament de compra de diferents usuaris en diferents llocs web.

Cosmetics[3] és un conjunt de dades extret de *Kaggle* el qual conté les compres de múltiples usuaris a diferents webs de productes cosmètics. Concretament, agafarem el mes d'octubre de l'any 2019 i les accions de guardar un producte a la cistella i de comprar.

Datasets	Cosmetics	Diginetica	Amazon
#item	23,194	24,889	9,114
#price level	10	100	50
#category	301	721	613
#session	156,922	187,540	204,036
#interaction	1,058,263	855,070	487,701

Taula 2: Característiques dels conjunts de dades

D'aquests conjunts de dades ens quedem el primer 70% de les dades en ordre cronològic per a entrenar el model, un 10% per a fer la validació i finalment l'últim 10% per a fer les proves de l'article final, és a dir, el test final i així evitem que el model pugui aprendre les dades del test final i acabi obtinguent resultats molt elevats a causa del *overfitting*.

1. Amazon
2. Diginetica
3. Cosmetics

4 Anàlisi de resultats

Aquest capítol inclou l'experimentació i es mostren els resultats obtinguts pels mètodes seleccionats per l'estudi amb tres conjunts de dades ja explicats anteriorment. S'analitzen aquests resultats mitjançant una sèrie de gràfics.

4.1 Mètriques d'avaluació

Per a avaluar els mètodes s'han usat dues mètriques:

- Prec@k: La precisió mesura el nombre de vegades que un ítem correcte esta dins de la previsió.
- MRR@k: L'anomenat **M**ean **R**eciprocal **R**ank és la mitjana dels rànquings reciprocs dels ítems correctes dins de les llistes de recomanació.

L'avaluació es fa amb *k=20* per a les dues mètriques. Aquest és el valor de k que s'utilitza habitualment a la literatura.

Durant un temps es va valorar l'ús de **N**ormalized **D**iscounted **C**umulative **G**ain (NDCG) com a mètrica, la qual compara la llista resultant amb la llista idònia on els millors elements estan al principi. Aquesta mètrica va ser descartada, doncs el MRR ja ens proporciona una valoració segons el primer ítem rellevant de la llista, que és el que més ens interessa.

4.2 Anàlisi comparatiu d'algorismes

El primer que mostrarem en aquest apartat serà la taula dels resultats obtinguts per cada un dels mètodes proposats amb els conjunts de dades seleccionats. En negreta es veurà el millor resultat obtingut, per així facilitar l'enteniment de la taula.

Mètode	Diginetica		Cosmetics		Amazon	
	M@20	P@20	M@20	P@20	M@20	P@20
CoHHN	**25.25**	**62.05**	**58.79**	60.35		**67.75**
GCSAN	17.26	51.78	34.42	46.29	14.02	39.18
KNN-Based	16.49	49.35	30.80	47.68	47.56	60.47
LESSR	18.27	52.58	25.36	47.85	55.71	64.33
PASBR	17.55	52.32	33.56	**68.17**	5.38	15.37
GRU4Rec	11.84	28.15	14.68	21.94	52.24	56.18

Taula 3: Resultats dels diferents algorismes amb tots els datasets

Ara analitzarem els resultats obtinguts pels diferents mètodes estudiats, també veurem com canvien els resultats amb el temps i com es comparen els mètodes entre ells. Tot això ho farem amb l'ajuda de diferents gràfiques.

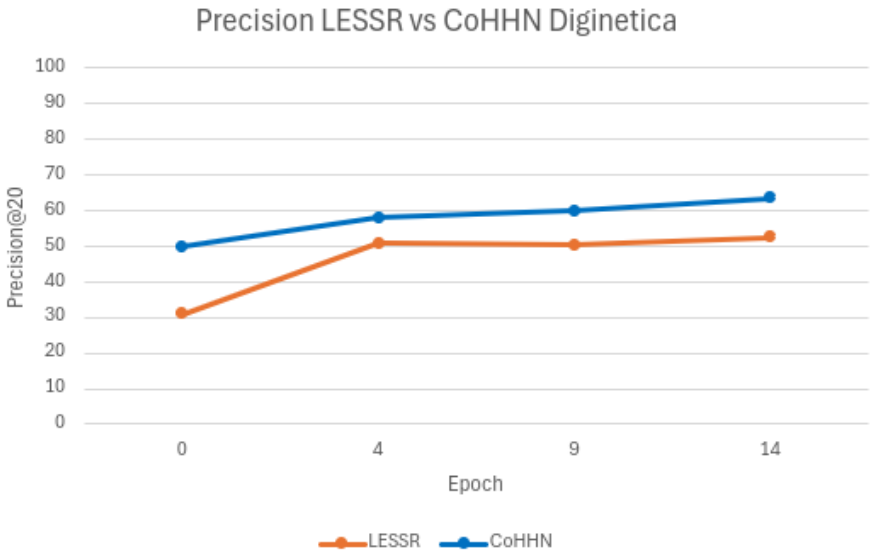

Figura 17: Precision LESSR vs CoHHN amb Diginetica

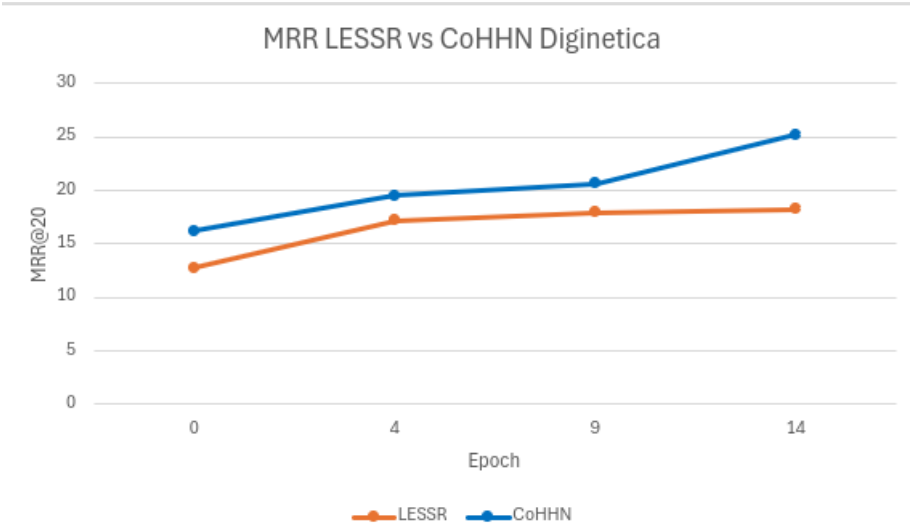

Figura 18: MMR LESSR vs CoHHN amb Diginetica

A la Figura 17 i la Figura 18 podem observar com es comporta un recomanador basat en sessions com és el *LESSR*, sense tenir present la sensibilitat de preu, respecte a un mètode de recomanació basat en sessions que es basa en la sensibilitat de preu com és el *CoHHN*. Es pot observar que tant a nivell de precisió com de *MRR* el mètode amb millors resultats es el *CoHHN*, doncs com ja s'ha explicat al llarg de la memòria, el preu és un element a tenir en compte durant la recomanació.

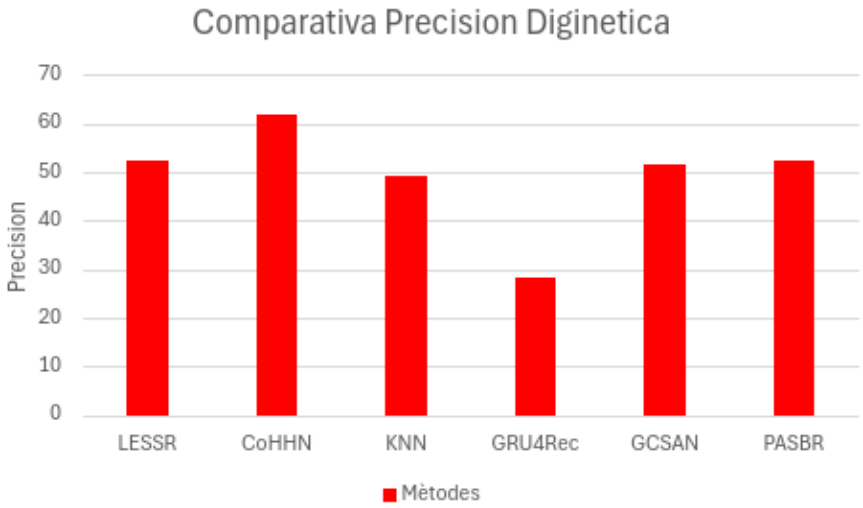

Figura 19: Comparativa de la precision dels mètodes

A la Figura 19 tenim la comparativa de la precisió obtinguda per tots els mètodes i es pot observar com el mètode més precís amb el conjunt de dades de diginetica és el *CoHHN* fet que es pot atribuir al seu sistema d'agregació de dos canals a partir del hiper-graf heterogeni que utilitza.

És certament sorprenent veure com a la Figura 19 el model *PASBR* no destaca per sobre dels seus competidors, especialment del *LESSR*, el qual obté millors resultats. Sembla que en aquest dataset en concret, l'extracció i consideració de la sensibilitat del preu no comporta una millora substancial respecte a la competència, sent que *LESSR* usa un sistema molt similar per a crear els grafs.

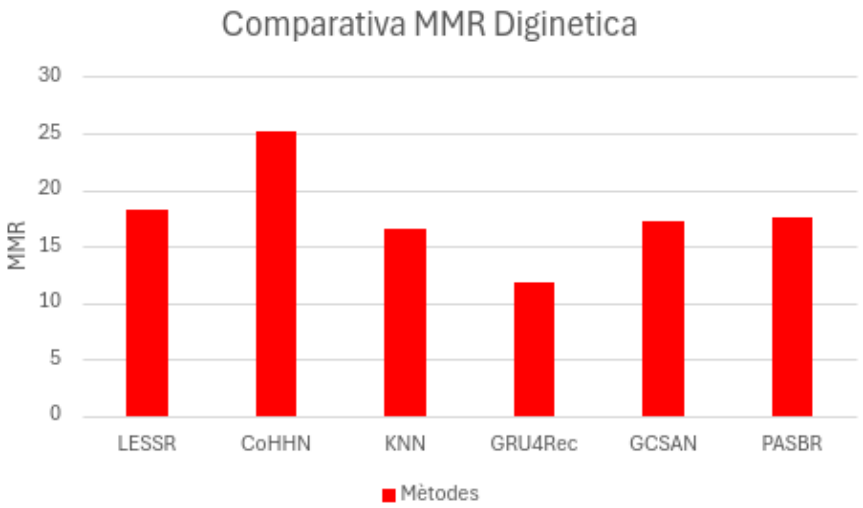

Figura 20: Comparativa del MRR dels mètodes

D'altra banda a la Figura 20 es pot observar la comparativa del *MRR* entre els mètodes estudiats. S'observa un comportament similar amb la precisió, *CoHHN* continua sent el millor i *PASBR* continua sense superar a *LESSR*.

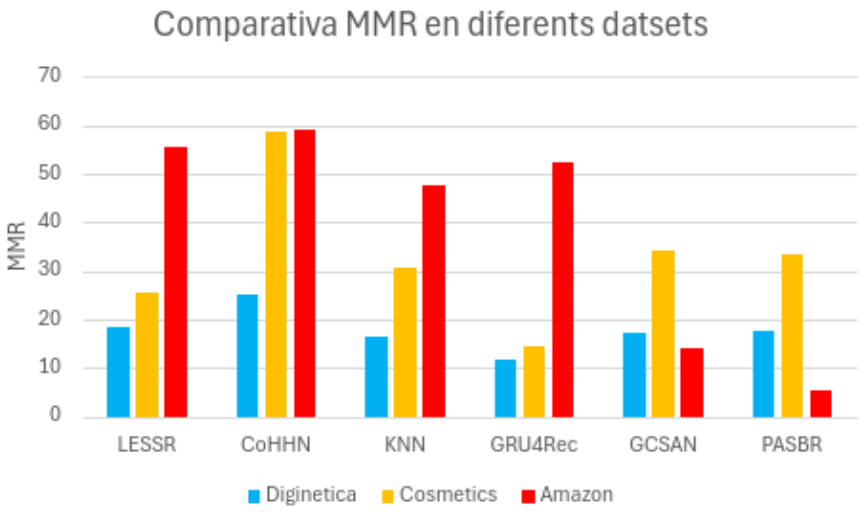

Figura 21: Comparativa del MRR dels mètodes amb tots els sets de dades

A la Figura 21 es pot veure la comparativa de resultats entre tots els mètodes amb els diferents conjunts de dades. Hi ha dos elements que destaquen en aquest gràfic. El primer és el valor que *CoHHN* obté sobre el conjunt de dades de *Cosmetics*, doncs és un valor proper a 60 mentre que la competència no arriba als 35. L'altre detall que observem és el baix *MMR* que obté *PASBR* amb *Amazon*. Si ve en els altres conjunts de dades *PASBR* obté resultats similars a la competència, en *Amazon* és queda molt lluny de l'esperable. Si ve pot ser que el propi mètode és dolent en aquest conjunt de dades, la causa d'aquest pobre resultat pot venir donada per la forma en la que guarda les categories i els preus, doncs, exceptuant *CoHHN*, la resta de mètodes no fan servir el preu o la intenció.

A la figura 22, contràriament al que es podria pensar veient el gràfic comparatiu del *MRR* de la Figura 21, *PASBR* es el mètode que obté una precisió més alta usant el conjunt de dades de *Cosmetics*. És també interessant mencionar que, tot i que en termes de *MRR* els mètodes obtenien millors resultats en el conjunt de dades de *Cosmetics* respecte *Diginetica*, és l'invers en termes de precisió.

A les Figures 21 i 22, els gràfics precisió i *MRR*, s'observa que *GRU4Rec* funciona molt bé amb el conjunt de dades d'*Amazon*. Amb aquestes dades podem afirmar que utilitzar una *GRU* no garanteix millors resultats ja que per si sol no obté grans resultats en els conjunts de *Diginetica* i *Cosmetics*. Les *GRU* poden ajudar a millorar en combinació amb altres capes, doncs altres mètodes com *PASBR* també les utilitzen.

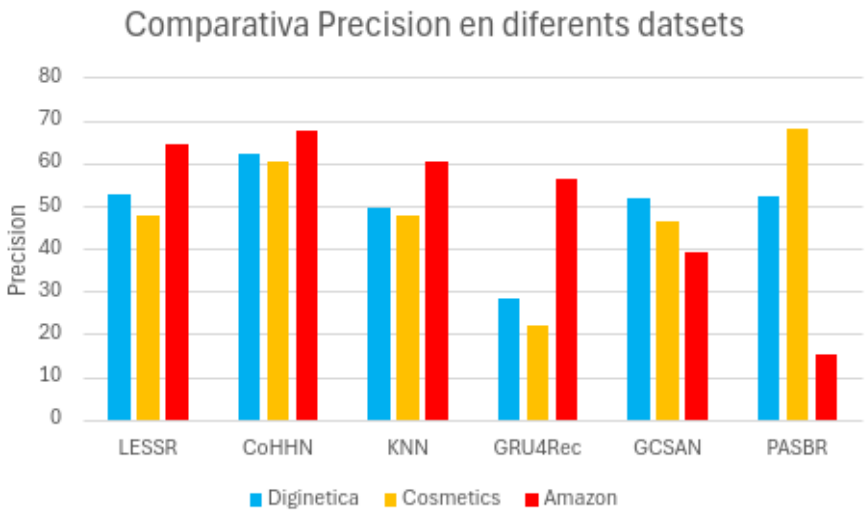

Figura 22: Comparativa de la Precision dels mètodes amb tots els sets de dades

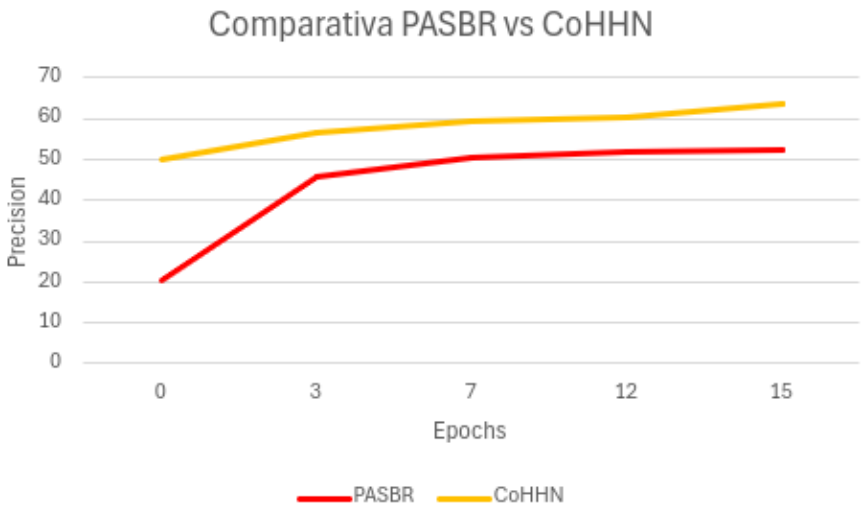

Figura 23: Comparativa PASBR vs CoHHN

La Figura 23 mostra una comparativa entre la precisió obtinguda per *PASBR* i *CoHHN*, els dos mètodes basats en la sensibilitat de preu de la selecció. Es tracta d'una comparativa força interessant, doncs es poden treure dues observacions. La primera és que *PASBR* comença amb pitjors resultats i en cap moment arriba a superar a *CoHHN*. A més a més, sembla que ja s'estanca i no milloraria a l'augmentar les iteracions. La segona observació és que *CoHHN* continua amb una tendència ascendent al passar de la última iteració obtinguda.

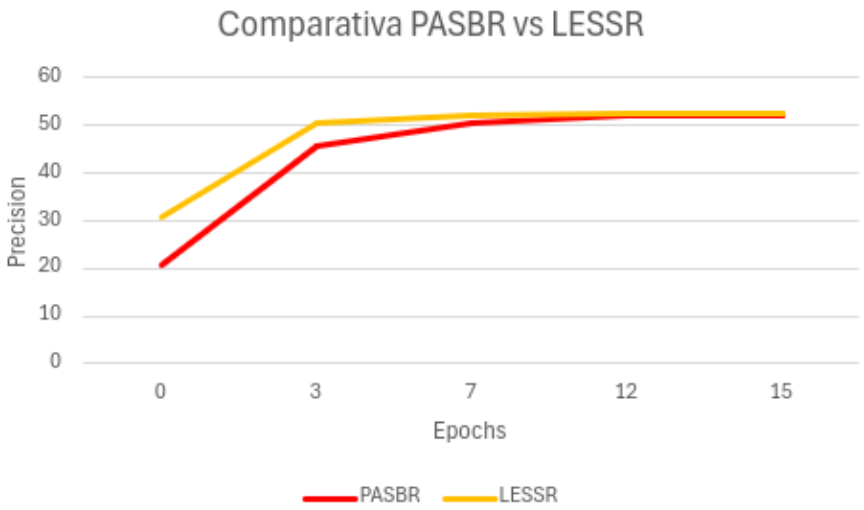

Figura 24: Comparativa PASBR vs LESSR

La motivació darrera d'aquest últim gràfic de la Figura 24 era comparar els resultats dels mètodes *PASBR* i *LESSR* al llarg del temps ja que aquests dos mètodes utilitzen un sistema molt similar per a codificar els grafs, les capes *EOP* i *SGAT*, i veure si realment afecta afegir el preu als models. De primeres es pot observar que a l'última iteració els resultats entre els dos mètodes són pràcticament idèntics, però *LESSR* hi arriba abans.

4.3 Discussió i Resum final

Acabem l'apartat de resultats intentant explicar el perquè darrera dels resultats obtinguts i si realment tenir en compte la sensibilitat de preu és una tècnica útil. També respondrem les preguntes de recerca plantejades anteriorment.

P1: Quin model obté millors resultats? Afecta la sensibilitat de preu? De forma general podem dir que *CoHHN* demostra tenir els millors resultats. A més podem dir que tenir en compte la sensibilitat de preu pot ser beneficiós a l'hora de realitzar les prediccions.

Els dos mètodes que ens poden aclarir millor si realment és viable considerar la sensibilitat de preu a un model són *LESSR* i *PASBR*, doncs el segon s'inspira en la forma de creació de grafs del primer i per tant, la diferència es troba en el processament de dades, on *LESSR* no incorpora la sensibilitat de preu però *PASBR* sí. Exceptuant el cas del conjunt de dades d'*Amazon*, on s'han obtingut uns resultats bastant confusos, *PASBR* no ha obtingut resultats inferiors als obtinguts per *LESSR*, fins hi tot, ha obtingut resultats notablement millors en el cas de *Cosmetics*. Amb aquestes dades es pot assumir que, en determinats conjunts de

dades és important tenir en compte la sensibilitat de preu i encara més important, no afecta negativament tenir present aquest element en els altres casos.

P2: Com afecten els paràmetres dels models als resultats? Cada mètode té paràmetres propis, però ni ha molts que es repeteixen al llarg de tots els models. A continuació veurem com afecten aquests paràmetres comuns entre la majoria de mètodes escollits al resultat final.

- **Epoch:** Una epoch és una iteració sencera a tot el conjunt d'entrenament.
- **Batch Size:** Ens indica el nombre d'elements que el model agafa per a cada iteració del entrenament. Com més gran sigui el valor del *Batch Size*, més ràpid anirà el model a completar cada iteració, però serà més imprecís.
- **Embedding Size:** Aquest paràmetre només esta present a *LESSR, PASBR* i *CoHHN*. Ens indica la mida dels vectors a ser codificats per a passar les dades dels conjunts a matrius per a entrenar el model. Una mida gran generalment donarà millors resultats, però fins a un cert límit, doncs pot causar un *overfitting*, és a dir, que el model memoritza el conjunt d'entrenament i per tant, no pot predir correctament els valors dels tests.
- **nHeads:** Aquest paràmetre només es present als models *PASBR* i *CoHHN*. Considero que s'ha d'explicar igualment doncs està present als dos mètodes basats en preu. És el nombre de fils en paral·lel que s'utilitzen en un sistema d'atenció multi-capa. De la mateixa manera que funciona la *Embedding Size*, a més alt és el número, millors resultats s'obtindrà, però un nombre molt alt pot portar a un *overfitting*.
- **Layer:** Aquest paràmetre també és exclusiu de *LESSR, PASBR* i *CoHHN*. Ens permet seleccionar el número de capes intermèdies entre la entrada i sortida, per exemple, en el cas de *LESSR*, indica quantes capes *EOP* i *SGAT* hi haurà. Passa també com amb la resta de paràmetres, com més alt és aquest valor, més dens és el model i més elements pot aprendre, però si el valor és molt alt, pot causar un *overfitting*.
- **Learning Rate:** S'utilitza per controlar el ritme al que un algorisme actualitza o aprèn els valors d'un paràmetre estimat. En unes altres paraules, controla el pes de la caiguda del gradient.

Cal comentar que el model *KNN*, al ser més senzill que la competència només te tres d'aquests paràmetres, *epochs*, *batch size* i *learning rate*. A part, té el paràmetre k, per a escollir els veïns, i el valor òptim serà 500.

Ara explicarem els valors òptims d'aquests paràmetres que s'han trobat durant l'execució dels codis. S'explicaran les proves realitzades amb cada paràmetre i com han afectat als resultats. Abans però, s'ha de posar una mica de context. Tots els codis s'han executat amb 30 *epochs* per a obtenir els resultats finals, però en la majoria de mètodes, el millor resultat s'ha obtingut entre la iteració 12 i 19. Per fer les proves dels paràmetres, s'han executat els codis amb només cinc iteracions i s'ha agafat el paràmetre que retornava una precisió superior. Per ser més extens, s'hauria d'executar múltiples variacions de paràmetres alhora, doncs podria

ser que un valor de *batch size* per si sol no donés grans resultats, però conjuntament amb un valor concret de *embedding size* hagués trobat un resultat millor. Així doncs, tenint present aquests punts, anem a veure quins valors s'han escollit finalment per a cada paràmetre i per cada mètode.

El paràmetre *batch size*, en la majoria de mètodes venia amb valor 100 per defecte. Excepte en *PASBR* i *LESSR* on era 512 i el *GRU4Rec* on era 50. La prova ha consistit en augmentar el valor a partir de 50 en potencies de dos, és a dir, començant per el 64 fins a 1024, per tant, 64, 128, 256, 512 i 1024. Fent aquestes proves s'ha trobat que el millor valor per aquest paràmetre ha sigut: 512 per a *PASBR* i *LESSR*, *256* per a *CoHHN* i *128* per a la resta.

El paràmetre *Embedding Size* per defecte ve de la següent forma: 32 per a *PASBR* i *LESSR*, 128 per a *CoHHN*. De la mateixa forma que el *batch-size*, s'ha anat augmentant en potencies de dos fins trobar que, per a *LESSR* el millor resultat es trobava deixant el paràmetre a 32, per *PASBR* s'havia d'augmentar a 64 i per *CoHHN*, a 128 igual.

Per a *CoHHN* i *PASBR*, el paràmetre *nHeads* ve donat com 4 i 2 respectivament. Els millors resultats han sigut obtinguts deixant tots dos models amb el paràmetre a 4.

El paràmetre *learning rate* s'ha deixat en 0.001 en tots els mètodes.

Per últim tenim el paràmetre *layer*. Per defecte ve com: 3 per a *LESSR*, el qual ja és el valor òptim, 1 per a *PASBR*, el qual augmentarem fins a 3 per a obtenir el valor òptim, i 2 per a CoHHN. Per a CoHHN els autors comenten que s'ha de deixar a dos per al conjunt de dades d'*Amazon* i 3 per a la resta de conjunts, i així s'ha fet.

5 Conclusions i Treball Futur

En aquest últim apartat acabarem el projecte amb unes conclusions finals, així com una vista al futur.

5.1 Conclusions

En la primera fase del projecte, durant l'etapa de lectura i resum per a elaborar l'estat de l'art, ens ha permès entendre com funcionen els sistemes de recomanació econòmics, quins tipus hi ha i més específicament, com funcionen els sistemes de recomanació econòmics basats en la sensibilitat de preu mitjançant una sèrie d'exemples.

Gràcies a l'apartat teòric també s'ha pogut entendre la motivació de considerar el preu a l'hora de realitzar les recomanacions i a quins problemes s'enfronta i com s'intenta solucionar-los.

La part pràctica ha servit per confirmar tot allò que s'ha aprés a la part teòrica. A més a més, s'ha pogut analitzar el codi dels recomanadors i aprendre les eines necessàries per a crear-ne un de nou, així com també aprendre com afecten tots els paràmetres dels models als resultats d'aquests.

En resum, s'ha realitzat una introducció teòrica als mètodes estudiats, s'ha revisat i analitzat el codi i finalment s'han modificat aquests per tal de comparar-los entre ells amb diferents conjunts de dades.

En relació als objectius plantejats a l'inici del projecte, es pot dir que s'han assolit satisfactòriament. Dit això, no s'ha aconseguit fer funcionar tots els mètodes que s'havia plantejat en un principi, doncs es volien afegir mètodes com el *PUP* [27] o el *PEDR* [8], tots dos recomanadors basats en la sensibilitat de preu, però que per dificultats a l'hora d'executar el codi no s'ha pogut realitzar.

5.2 Treball Futur

La primera tasca que es podria fer de cara a un futur, seria fer funcionar més mètodes basats en preu per a comprovar els seus resultats, com per exemple *PUP* [27] o *PASVD* [23], ja que es difícil, amb només dos mètodes basats en la sensibilitat de preu, arribar a una ferma conclusió.

Una altra tasca seria crear un nou mètode basat en preu utilitzant tot allò que hem aprés al llarg del projecte, com les capes *EOP* i *SGAT* de *LESSR*, les capes *LTIRL* o *STIRL* de *PASBR* i el mecanisme d'agregació de dos canals de *CoHHN*. Realitzar proves amb aquest nou mètode i arribar a unes conclusions.

6 Bibliografia

Referències

[1] E.T Berkman i J.L Livingston. "Diminishing Marginal Utility - An Overview". A: *ScienceDirect Topics* (2016). URL: https://www.sciencedirect.com/topics/psychology/diminishing-marginal-utility#:~:text=Diminishing%20marginal%20utility%20refers%20to.

[2] T. Chen i R. C.-W. Wong. "Handling Information Loss of Graph Neural Networks for Session-based Recommendation". A: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020. URL: https://doi.org/10.1145/3394486.3403170.

[3] A. De Biasio, N. Navarin i D. Jannach. "Economic recommender systems – a systematic review". A: *Electronic Commerce Research and Applications* 63 (2024), pàg. 101 - 352. URL: https://doi.org/10.1016/j.elerap.2023.101352.

[4] Evelyn. *Collaborative Filtering in Recommender System: An Overview*. Medium. Nov. de 2023. URL: https://medium.com/@evelyn.eve.9512/collaborative-filtering-in-recommender-system-an-overview-38dfa8462b61.

[5] J. Feng, Y. Wang i S. Chen. "Modeling Price-Aware Session-Based Recommendation Based on Graph Neural Network". A: *Computers, Materials & Continua* 76.1 (2023), pàg. 397 - 413. URL: https://doi.org/10.32604/cmc.2023.038741.

[6] Z. Fu, C. Wang i J. Xu. "Graph contextualized self-attention network for software service sequential recommendation". A: *Future Generation Computer Systems* 149 (2023), pàg. 509 - 517. URL: https://doi.org/10.1016/j.future.2023.07.041.

[7] Robert Garfinkel i et al. "Design of a Shopbot and Recommender System for Bundle Purchases". A: *Decision Support Systems* 42.3 (des. de 2006). Accessed: 25 November 2020, pàg. 1974 - 1986. DOI: 10.1016/j.dss.2006.05.005. URL: https://www.sciencedirect.com/science/article/abs/pii/S0167923606000741.

[8] W. Guo, J. Tian i M. Li. "Price-aware enhanced dynamic recommendation based on deep learning". A: *Journal of Retailing and Consumer Services* 75 (2023), pàg. 103 - 500. URL: https://doi.org/10.1016/j.jretconser.2023.103500.

[9] B. Hidasi et al. "SESSION-BASED RECOMMENDATIONS WITH RECURRENT NEURAL NETWORKS". A: (2016). URL: https://arxiv.org/pdf/1511.06939.

[10] Kartik Hosanagar i et al. "Recommended for You: The Impact of Profit Incentives on the Relevance of Online Recommendations". A: *Association for Information Systems AIS Electronic Library (AISeL)*. Recommended Citation. 2008.

[11] T. Iwata, K. Saito i T. Yamada. "Recommendation Method for Improving Customer Lifetime Value". A: *IEEE Transactions on Knowledge and Data Engineering* 20.9 (set. de 2008), pàg. 1254 - 1263. DOI: 10.1109/TKDE.2008.55.

[12] Yuanchun Jiang i et al. "Optimizing E-Tailer Profits and Customer Savings: Pricing Multistage Customized Online Bundles". A: *Marketing Science* 30.4 (jul. de 2011), pàg. 737-752. DOI: 10.1287/mksc.1100.0631. URL: https://doi.org/10.1287/mksc.1100.0631.

[13] İlyurek Kılıç. *Understanding Matrix Factorization: A Simple Guide*. Medium. Accessed: 6 June 2024. Set. de 2023. URL: https://medium.com/@ilyurek/understanding-matrix-factorization-a-simple-guide-20e2b32989eb%5C#:%5C~:text=Matrix%5C%20factorization%5C%20is%5C%20a%5C%20versatile.

[14] Kyung Young Lee et al. "Online consumers' reactions to price decreases: Amazon's Kindle 2 case". A: *Internet Research* 26.4 (2016), pàg. 1001-1026. ISSN: 1066-2243. DOI: 10.1108/IntR-04-2014-0097.

[15] Edward Loper i Steven Bird. "NLTK: The Natural Language Toolkit". A: (2002). URL: https://arxiv.org/pdf/cs/0205028.

[16] mikael556. *Introduction to Learn-To-Rank (LTR) and Its Application in Finance*. Markov Capital AB. Accessed: 6 June 2024. Ag. de 2023. URL: https://www.markovcapital.se/post/learn-to-rank-part-1.

[17] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". A: (2013). URL: https://arxiv.org/pdf/1301.3781.

[18] Changhua Pei et al. *Value-Aware Recommendation Based on Reinforced Profit Maximization in E-Commerce Systems*. 2019.

[19] Ruihong Qiu et al. "Rethinking the Item Order in Session-based Recommendation with Graph Neural Networks". A: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM)*. 2019, pàg. 579-588.

[20] M Shanmuganathan i et al. "Multi Attribute Utility Theory - An Overview". A: *International Journal of Scientific & Engineering Research* 9.3 (2018). URL: http://www.ijser.org/researchpaper/Multi-Attribute-Utility-Theory-An-Over-View.pdf.

[21] Dhilip Subramanian. *A Simple Introduction to K-Nearest Neighbors Algorithm*. Accessed: 8 June 2019. Juny de 2019. URL: https://towardsdatascience.com/a-simple-introduction-to-k-nearest-neighbors-algorithm-b3519ed98e.

[22] Fei Sun et al. "BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer". A: (2019). Accessed 26 Sept. 2022. DOI: 10.1145/3357384.3357895. URL: https://arxiv.org/pdf/1904.06690.pdf.

[23] J. Tian, X. Shi i M. Li. "Price-aware matrix factorization model for personalized recommendations". A: *Information Management* 60.5 (2023), pàg. 103-815. URL: https://doi.org/10.1016/j.im.2023.103815.

[24] Shu Wu et al. "Session-based Recommendation with Graph Neural Network". A: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. 2019, pàg. 346-353.

[25] K. Yehuda i R. Bell. *Data Science Repo - Machine Learning, AI, Stats*. Datajobs.com. 2010. URL: https://datajobs.com/data-science-repo/Collaborative-Filtering-[Koren-and-Bell].pdf.

[26] X. Zhang et al. "Price DOES Matter! Modeling Price and Interest Preferences in Session-based Recommendation". A: (2022). URL: https://doi.org/10.1145/3477495.3532043.

[27] Y. Zheng et al. "Incorporating Price into Recommendation with Graph Convolutional Networks". A: *IEEE Transactions on Knowledge and Data Engineering* (2021), pàg. 133-144. URL: https://doi.org/10.1109/tkde.2021.3091160.

[28] Tao Zhu i et al. "Bundle Recommendation in Ecommerce". A: *Proceedings of the 37th International ACM SIGIR Conference on Research Development in Information Retrieval*. Jul. de 2014. DOI: 10.1145/2600428.2609603. URL: https://doi.org/10.1145/2600428.2609603.