PROJEKTNA DOKUMENTACIJA

ExRec

Voditelj:

Zagreb, kolovoz 2020.

1. Općenito o ExRec-u

U radu (1) prepoznata je neefikanost prevladavajućih sustava otvorenih online tečajeva (MOOC) zbog dodjeljivanja istih vježbi svim studentima. S druge strane, personalizirani sustavi preporuka zadataka mogu poboljšati efikasnost učenja prilagođavajući se razini znanja svakog studenta (učenika, korisnika). Takvo razmišljanje uvelike odgovara našem problemu. U originalnom radu krenulo se od temeljne pretpostavke da je ubrzavanje učenja temeljni cilj svakog personaliziranog sustava preporuke zadataka u obrazovanju. Rad predlaže personalizirani sustav preporuke zadataka za online učenje poboljšanjem performansi postojećih *knowledge tracing* modela. Napominje se da postojeći modeli praćenja znanja ne koriste informacije o pripadnosti zadataka konceptima. Konkretno, ovim radom modificira se *dynamic key-value memory network knowledge tracing (DKVMN)* model tako da se memorijska struktura temelji na listi koncepata određenog tečaja eksplicitno bilježeći vezu zadataka i koncepata tijekom procesa praćenja znanja. Model je korišten za izgradnju simulatora studenata korištenog za treniranje strategije preporuke zadataka pomoću podržanog učenja.

1.1. Povezani radovi

ExRec je sustav sastavljen od praćenja znanja i preporuke zadataka. Nadogradnja je već poznatih i istraženih sustava. Praćenje znanja inače često koristi prije istražen i implementiran Bayesovski model praćenja znanja (BKT). Modelira studentovo znanje koncepta kao binarnu varijablu i, koristeći skriveni Markovljev model, ažurira vjerojatnost njegova ovladavanja konceptom uzimajući u obzir rezultate rješavanja zadataka. Taj je model na razini koncepata i zanemaruje odnose između različitih koncepata.
Drugi pristup može se pronaći u također već istraženom modelu dubokog praćenja znanja (DKT) s povratnom neuronskom mrežom. Modelira znanje studenta kao latentnu varijablu. DKT je korišten i za sustav preporuke.

Preporuke zadataka većinom koriste heurističke algoritme. Studentu se preporučuje zadatak ako je vjerojatnost da će ga točno riješiti oko 50%. Optimalnost takvog algoritma je upitna.
Koristi se i pristup određivanja ZPD-a (zona proksimalnog razvoja, engl. *Zone of Proximal Development*) na temelju trenutnog znanja učenika, a zatim se odabire najkorisniji zadatak pomoću algoritma više naoružanih razbojnika (engl. *multi-armed bandits algorithm*).
SPARFA framework se koristi za procjenu znanja svakog učenika iz njihovih prethodnih rezultata. Zatim koristi te profile znanja kao kontekst i primjenjuje kontekstualni algoritam naoružanih razbojnika za preporuku zadataka kako bi se maksimizirao učenikov neposredan uspjeh, odnosno njegov uspjeh na sljedećem zadatku. Problem ovog algoritma je što uzima u obzir samo sljedeći korak (kratkoročna nagrada) stoga njegova izvedba ne mora biti optimalna.

1.2. Pozadina sustava i dataset

Originalni ExRec sustav kao dataset uzima uzorke studentskih interakcija iz IPS-a (engl. *Intelligent practice System*. IPS je kineski online sustav za samostalno učenje. U IPS-u svaki tečaj ima na desetke gradiva, a student sam bira željeno gradivo. Svako gradivo ima 7 stadija iz kojeg je moguće izaći u bilo kojem trenutku (ili promijeniti gradivo):

1. vježbe za zagrijavanje prije nastave
2. vježbe na nastavi prije predavanja
3. video predavanja
4. vježbe na nastavi nakon predavanja
5. domaća zadaća
6. vježbe pregleda gradiva
7. vježbe pregleda raznih gradiva.

Svaka vježba (zadatak) ima tri hijerarhijske oznake koncepata (*tagove*) koje su im pridijelili eksperti. U stadijima od 1 do 5 uključeni su sadržaji jednog koncepta, dok 6 i 7 sadrže vježbe drugih koncepata zbog procjene naučenosti. Sustav sprema trajanje studentovog učenja u svakom stadiju, vježbe koje polaže i točnost rezultata.

Za potrebe našeg projekta napravljena je skripta koja pretvara Assistments dataset (skillbuilder.csv) u format potreban za generiranje preporuka.
Za svakog studenta rezervirana su 4 retka:

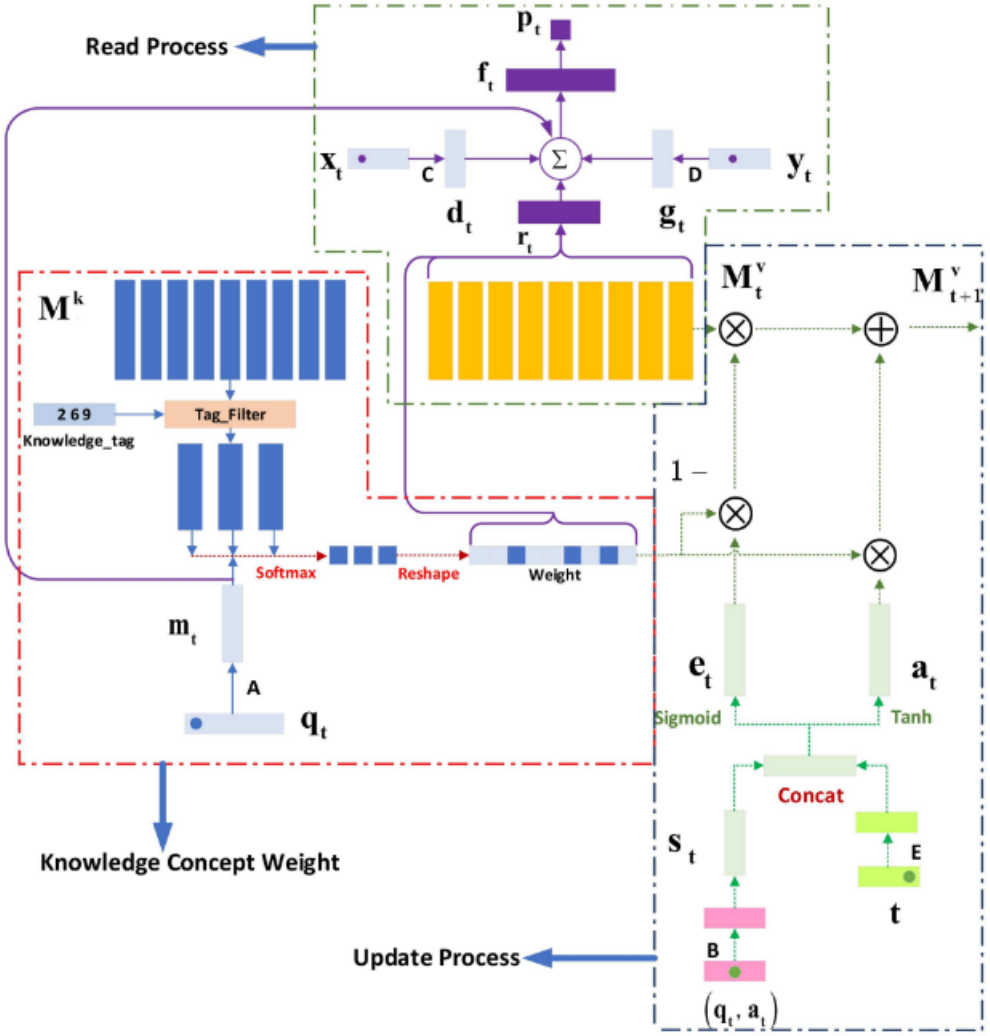
- prvi redak je broj odgovorenih pitanja/vježbi,
- drugi redak je lista identifikacijskih brojeva pitanja,
- treći redak redoslijed točnosti odgovora na pitanja, a
- četvrti redak sadrži koncept prve razine svakog pitanja.

1.3. DKVMN - model za praćenje znanja

DKVMN se generalno sastoji od statičke matrice (ključa) koji sprema latentne koncepte i dinamičke matrice (vrijednosti) koja sadrži razine savladnosti određenog koncepta. Model računa korelaciju vježbe i latentnog koncepta u ključu. Korelaciju koristi za čitanje studentovih razina savladanost koncepta i predviđa točnost ishoda rješavanja zadatka.

1.3.1. Concept Aware struktura

DKVMN ExRec praćenja znanja poboljšan u aspektima memorijske strukture, težina koncepata znanja te procesa pisanja i čitanja. Osnovni DKVMN dizajn je modificiran tako da memorijska struktura ovisi o listi koncepata nekog tečaja. Na slici <> je prikazana struktura modela. M_t^k je matrica ugrađenih koncepata veličine MXN gdje je N broj memorijskih lokacija, a M je veličina vektora na svakoj lokaciji. N se postavlja na broj koncepata znanja određenog tečaja. Pošto se u radu koristilo 1 koncept prve razine, 7 koncepata druge razine te 15 treće razine N=23. Nakon toga, na svakoj lokaciji koncepta znanja se sprema studentovo znanje.



Slika 1.1

1.3.2. Knowledge Concept Weight

Pošto je studentovo stanje koncepta znanja spremljeno u memoriji, kada se pojavi novi zadatak dohvaćaju se i osvježavaju samo memorijske lokacije povezane s tom vježbom. Za svaki koncept znanja se računa težina, težine se koriste kako bi se izračunala težinska suma trenutnih stanja koncepata znanja korisnika kako bi se predvidio njegov rezultat na vježbi. Također će se koristiti kako bi se osvježio korisnikovo stanje znanja nakon primitka rezultata od zadatka.

Prvo se dohvaćaju ugrađene vrijednosti od danog zadatka. Kao što se vidi na slici kada zadatak q_t dođe u trenutku t prvo se transformira u ugrađeni vektor m_t kroz ugrađujuću matricu A, tada se KCW računa pomoću algoritma 1. Ukratko, DKVMN računa težinske veze između zadataka i skrivenih koncepata znanja, dok se u ovom

kodu računaju samo veze između zadataka i poznatih povezanih koncepata, nepovezani koncepti se postavljaju na 0.

Algorithm 1 Knowledge Concept Weight Calculation

Input:
$\mathbf{q}_t$: embedding of the exercise arrived at time t
K_t: knowledge concept list of q_t
$\mathbf{M}^k$: the concept embedding matrix

Output:
$Weight$: Knowledge concept weight of the exercise arrived at time t

/ Calculate KCW */*
$R \Leftarrow [\,]$
for each $n \in K_t$ **do**
  $corr \Leftarrow \mathbf{m}_t^T \cdot \mathbf{M}^k[n]$
  $R.append(corr)$
end for
$R_s \Leftarrow Softmax(R)$
/ Reshape the weight vector to make its length equal to the number of concepts*/*
$Weight \Leftarrow [0, ..., 0]$
$i \Leftarrow 0$
for $i < 3$ **do**
  Weight$[K_t[\mathrm{i}]] \Leftarrow R_s[i]$
  $i \Leftarrow i + 1$
end for
return $Weight$

Slika 1.2

1.3.3. Proces čitanja

Nakon što se izračunati KCW iskoristi za izračun težinske sume stanja koncepata znanja određenog korisnika $r_t = \sum_{i=1}^{N} w_i M_t^v$, na tu sumu rt se nadodaju značajke težine zadatka i "stage feature" dt,gt. Rezultat se šalje u potpuno povezani sloj sa aktivacijskom funkcijom Tanh kako bi se dobio "summary" vektor koji sadrži sve informacije o studentovom znanju povezanom sa qt i značajkama zadatka.

$$f_t = Tanh(W_0^T[r_t, d_t, g_t, m_t]) \tag{1.1}$$

Na kraju, ft prolazi kroz potpuno povezani sloj koji daje kao izlaz vjerojatnosti da bi student odradio zadatake qt točno.

$$p = Sigmoid(W_1^T f_t) \tag{1.2}$$

1.3.4. Proces ažuriranja

Proces ažuriranja mijenja vrijednosti matrice M_t^v koja predstavlja trenutno stanje studentovog koncepta znanja k. Ovaj model je drukčiji od DKVMN po tome da se razmatra i trajanje rješavanja. Prema povezanim radovima, trajanje rješavanja zadatka je povezano sa razinom znanja studenta. Pošto je vrijeme rješavanja kontinuirana varijabla ona se prvo diskretizira po njenoj distribuciji i unda se prikaže kao varijabla t te se koristi kako bi se ažurirala matrica M. Ostali procesi su isti kao i u DKVMN, sastoji se od podprocessa dodavanja i brisanja. Vektor brisanjas e dobije $e = Sigmoid(E^T[s_t, t])$ dok se vektor dodavanja dobije kao $a = Tanh(D^T[s_t, t])$ te se nova matrica M računa kao

$$M_{t+1}^v(i+1) = M_t^v(i)[1 - w(i)e][1 + w(i)] \tag{1.3}$$

Parametri ovog modela se računaju tako da se minimizira "cross-entropy loss" između pretpostavljenog studentovog dobvivenog i pretpostavljenog rezultata.

$$L = -\sum_t ((y_i \log p_t) + (1 - y_t) \log(1 - p_t)) \tag{1.4}$$

1.4. Preporuka zadataka podržanim učenjem

1.4.1. Općenito o podržanom učenju

Glavni elementi svakog algoritma podržanog učenja su, osim agenta i okoline, strategija, nagrada, vrijednost (predikcija nagrade) i opcionalno model. Strategija predstavlja agentov način ponašanja u određenom trenutku. Ona je preslikavanje iz spoznatog stanja okoliša u akcije koje je potrebno izvršiti u tim stanjima. Agentova je srž, ona sama je dovoljna da odredi ponašanje.
Cilj podržanog učenja definira nagrada koja preslikava opaženo stanje okoliša (ili par stanje-akcija) u jedan broj koji odgovara poželjnosti tog stanja. Cilj agenta je maksimizirati ukupnu nagradu. Nagrada definira što je trenutno dobro, a funkcija vrijednosti definira što je dugoročno dobro. Vrijednost stanja je očekivanje ukupne količine nagrade koja bi se mogla akumulirati u budućnosti počevši od tog stanja. To očekivanje još se naziva i Q funkcija.
Model imitira ponašanje okoline. Za dano stanje i akciju, model može predvidjeti rezultantno iduće stanje i nagradu.

U ovom radu okolina se izgrađuje simulatorom temeljenim na DKVMN-CA (CA = *concept aware*). Personalizirani agent za preporučivanje zadataka trenira se dubokim

učenjem, točnije korištenjem GRU-a (engl. *Gated Recurrent Unit*): poboljšane verzije standarnih RNN.

Proces odlučivanja modelira se kao Markovljev proces odlučivanja s djelomično vidljivim stanjima (POMDP). POMDP je generalizacija Markovljevog procesa odlučivanja. Modelira agentov proces odlučivanja; vezu agenta i okoline. Formalno, POMDP je skup 7 varijabli (stanje, akcija, uvjetna vjerojatnost prijelaza među stanjima, nagrada, opažanja, uvjetne vjerojatnosti opažanja, koeficijent umanjenja nagrade). Agent u svakom stanju bira akciju koja će maksimizirati buduću nagradu umanjenu za faktor umanjenja.
Osnovna je pretpostavka u podržanom učenju da agent vidi stvarno stanje svijeta. No, agentova opažanja u stvarnom svijetu nisu nužno isto što i pravo stanje: uvodi se vjerojatnost stanja $p(o_i|s_i)$ - vjerojatnost da se za opažanje o_i svijet nalazi u s_i. Sljedeće stanje u svijetu ovisi o trenutnom stanju i agentovoj akciji. Dva stanja svijeta mogu rezultirati u jednakom opažanju agenata. Za opažanja ne vrijedi Markovljevo svojstvo (sljedeće opažanje stanja ne ovisi isključivo o trenutnom opažanju i akciji). Potrebno je uzeti u obzir putanju agenta.

1.4.2. Primjena na ExRec

Konkretno, u ExRec-u stanje modela je studentovo konkretno latentno znanje, a akcija je preporuka zadatka. Strategija preporuka direktno funkcionira na sirovim opažanjima studentove povijesti rješavanja zadataka. U trenutku t agent ne može vidjeti studentovo znanje s_t. No, opažanje mu je o_t - zadatak i točnost rješavanja uvjetovani pomoću s_t, $p(o_t|s_t)$. Agent preporučuje a_t na temelju povijesti studentovih zadataka $h_t = (o_1, a_1, o_2, a_2, ..., o_{t-1}, a_{t-1})$. Nakon završetka preporučenog zadatka, latentno znanje prelazi u s_{t+1} pomoću $p(s_{t+1}|s_t, a_t)$.
Agentu je potrebna strategija kako bi izabrao najbolju akciju za određeno stanje. Nagrada akcije a_t je usrednjena suma vjerojatnosti da će student točno riješiti sljedeći zadatak, q, nakon što riješi predloženi zadatak u stanju s_{t+1}. Usrednjava se brojem zadataka. Taj sljedeći zadatak predviđen je simulatorom. Algoritam preporučuje zadatak q, čija je nagrada maksimalna. Opisano je vidljivo jednadžbom 1.3.

$$r_t = \frac{1}{K} \sum_{i=1}^{K} P_{t+1}(q_i) \tag{1.5}$$

Krajnji cilj je maksimizirati nagradu određene strategije gdje su trajektorije $(s_1, o_1, a_1, ...)$ preuzete iz distribucije trajektorija induciranih strategijom π. Nagrada je izražena jednadžbom 1.4.

$$R = \mathbb{E}_\tau \left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t) \right] \tag{1.6}$$

POMDP se rješava pomoću TRPO algoritma (engl. *Trust Region Policy Optimization*). TRPO je algoritam optimizacije robustan na velik raspon zadataka dajući monotono poboljšanje izmjenom malog broja parametara. Algoritam stalno iznova optimizira lokalne aproksimacije očekivane povratne vrijednosti strategije s kaznom KL-divergencije.

1.5. Evaluacija performansi

Originalni ExRec model evaluiran je na IPS datasetu provodenjem 50 eksperimenata podjelom korisnika na skupove za treniranje i testiranje u omjeru 70:30. Kao mjera performansi odabrana je AUC krivulja. Procjenjena je i efikasnost korištenjem dodatnih značajki modela, poput težine zadataka, faze rješavanja i trajanja rješavanja čija upotreba dokazano poboljšava performanse modela.

1.5.1. Preporuke zadataka

Evaluacija porasta znanja studenta

Algoritmi za treniranje strategije preporuke zadataka pomoću podržanog učenja većinom su heuristički; zadatke ocijenjene kao prelagane ili preteške potrebno je izbjegavati. No, optimalnost takvih algoritama nije zadovoljena jer se u obzir uzima samo kratkoročna nagrada. U ExRec-u se u obzir uzima dugoročna nagrada.
Kao strategije algoritma podržanog učenja u originalnom ExRec-u korištena su dva algoritma, Expectimax i RL. Kod Expectimaxa prvo se računa iznos predviđenog znanja

u slučaju da se preporuči određen zadatak. Sustav kao preporuku izabire zadatak kojim bi maksimizirao predviđeno znanje u tom trenutku. RL razmatra dugoročnu nagradu akcije maksimizacijom Q funkcije.
Za usporedbu algoritama, autori rada izvlače 15 studenata iz dataseta. Za oba algoritma inicijaliziraju se simulatori pomoću slijeda prethodno riješenih zadataka svakog studenta. Preporuča se 50 zadataka. Sprema se prosjek predviđenog znanja svih studenata u svakom koraku preporuke. Na slici je vidljivo da s RL algoritmom srednje predviđeno znanje stalno raste - moguće je pronaci zadatke koji poboljšavaju performanse.

Evaluacija preporuka

Za proučavanje RL strategije autori su osmislili dodatni eksperiment kojim se uzme student i njegovih prijašnje riješenih 5 zadataka te pokrene simulator. Preporuči mu se novih 5 zadataka pomocu RL. Vizualno se prikaže tih 10 zadataka u obliku (broj zadatka, povezani koncept, točnost). Promatra se povezanost točnosti zadataka i odgovarajućih koncepata. Ako ne postoji znanje o nekom konceptu, grafički prikaz je crne boje. Kako znanje raste, boja je svjetlija.
Kad student netočno riješi zadatak, algoritam mu preporuči povezani koncept. Ako padne i preporučeni zadatak, sustav mu nudi ponovno rješavanje.
Nakon povećanja znanja o nekom konceptu, prebacuje se na neki drugi koncept. Ako je zadatak okarakteriziran kao lagan, znanje se ne povećava pretjerano iako je točno riješen.
Ponovno se preporučuje početni zadatak i stvar se ponavlja dok se ne riješi točno.

1.6. Poveznice

`http://incompleteideas.net/book/first/ebook/node9.html` `https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be` `https://en.wikipedia.org/wiki/Partially_observable_Markov_decision_process` `https://medium.com/@jonathan_hui/rl-trust-region-poli`

https://arxiv.org/abs/1502.05477

2. Literatura

[1] Concept-Aware Deep Knowledge Tracing and Exercise Recommendation in an Online Learning System (ExRec), https://files.eric.ed.gov/fulltext/ED599194.pdf