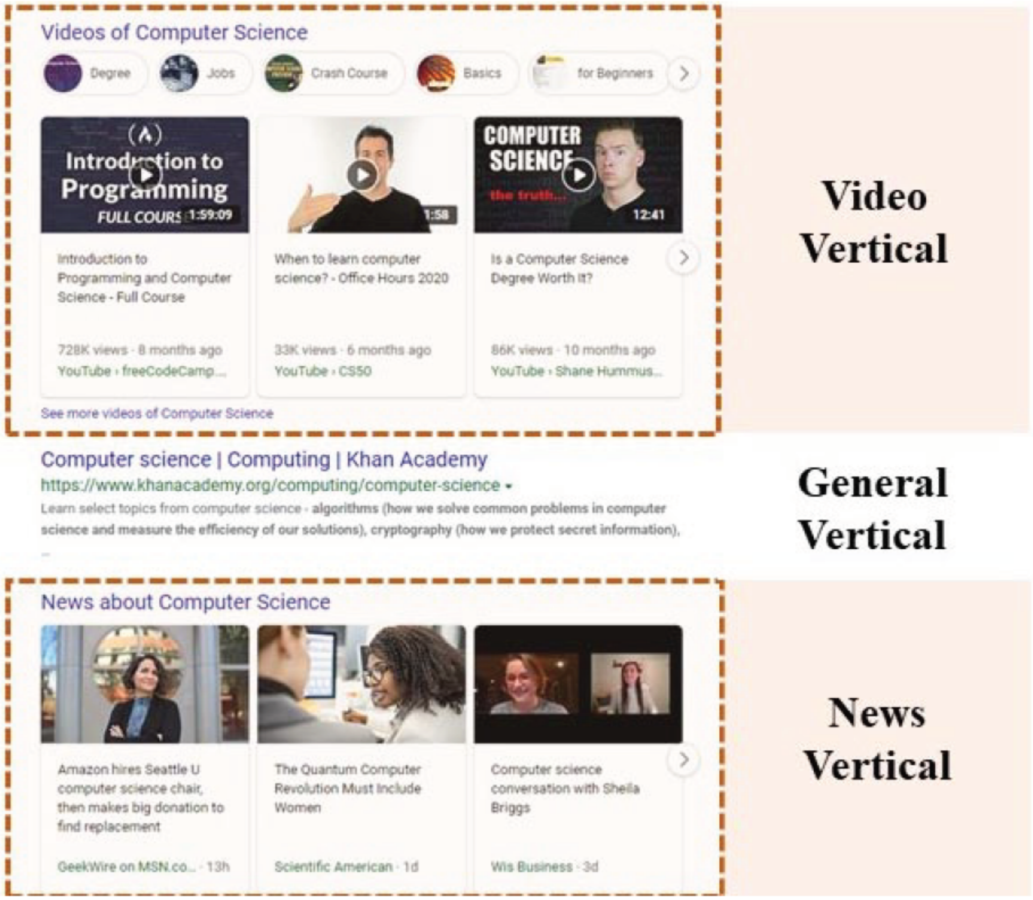
Hierarchical Reinforcement Learning for Aggregated Search

Pu Yang, Yin Zhang
zhangyin98@zju.edu.cn
Zhejiang University
15/07/2021

Aggregated Search Engine

Traditional Search Engine

Aggregated Search Engine

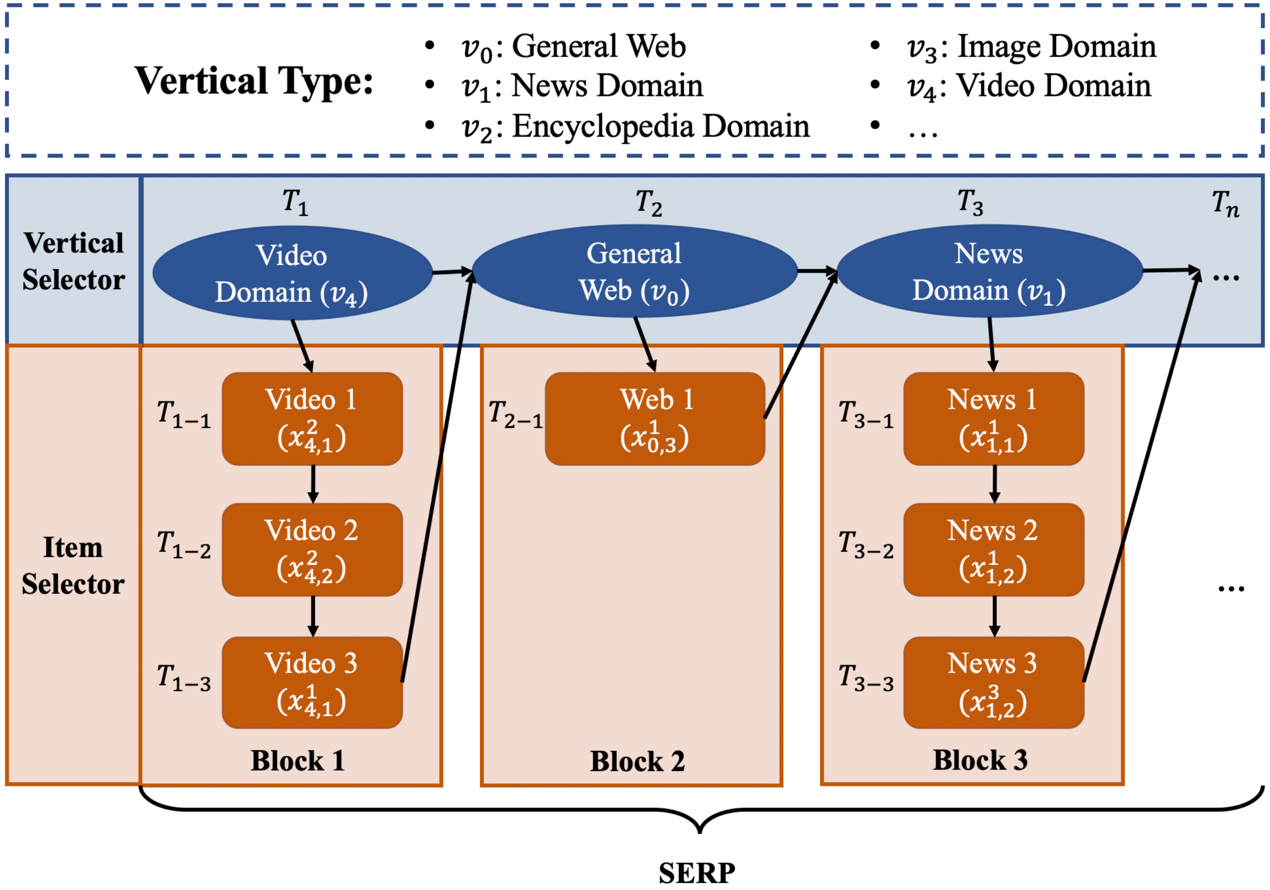


Traditional approaches

- **Vertical Domain Selection**
  - Determine the vertical domain that content blocks belong to.
  - Formulated as binary classification problems.
- **Item Ranking in a Vertical Domain**
  - Determine the ranking of result in a certain vertical area.
  - Solved by various L2R algorithms or modeled as sequential decision-making problems.
- **Global Result Ranking**
  - Determine the ranking for content blocks to form a page.
  - Solved by various L2R algorithms.

Motivation

- **Traditional approaches solve the three subtasks separately in a pipeline.**
  - Models are complex and errors might be accumulated.
- **Correlation among the subtasks is ignored.**
  - Vertical selection determines the result options.
  - Items selected have impact on the selection of subsequent vertical domain.



- **How to develop an end-to-end model and jointly optimize the three subtasks?**

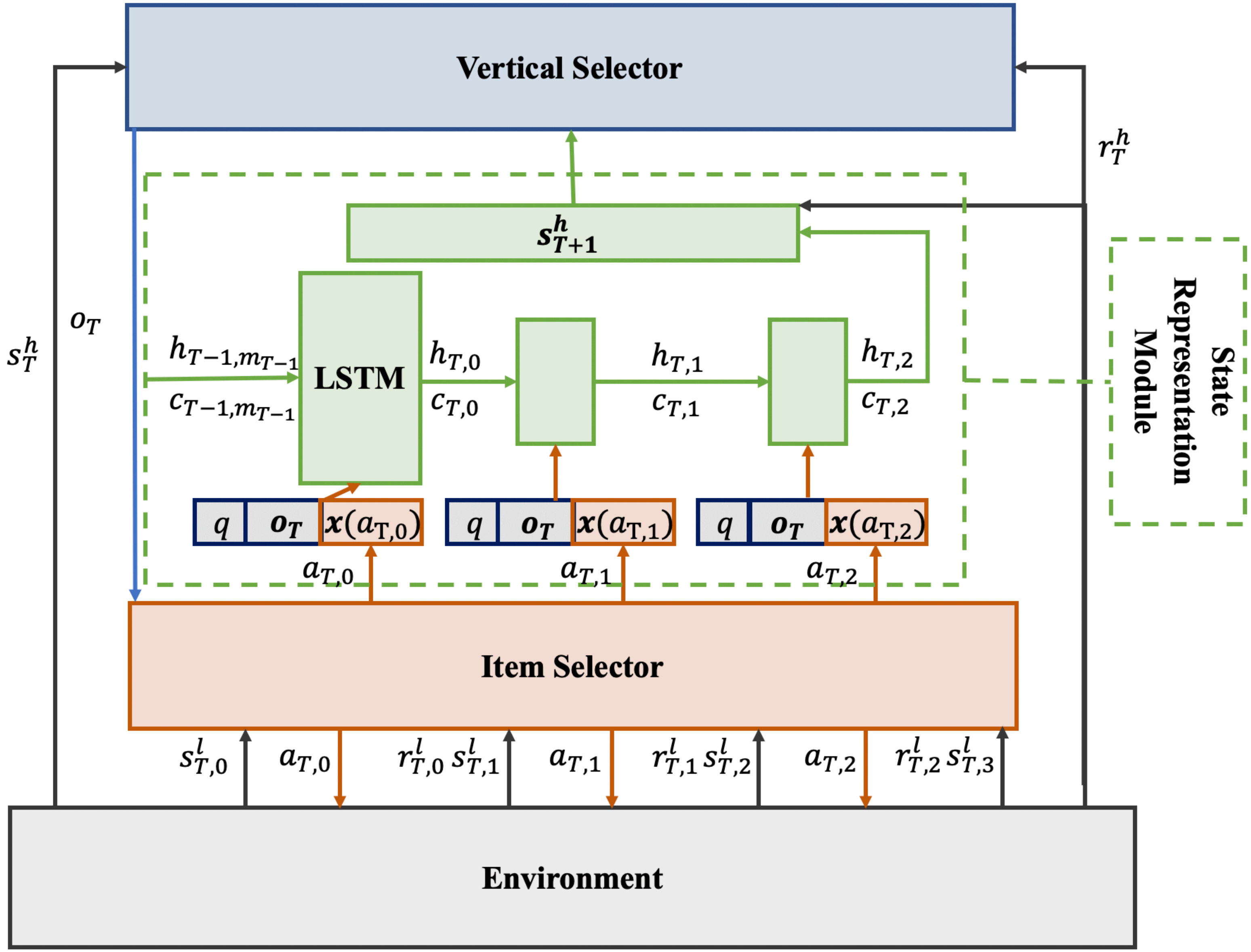
Related Work

- **BC+ISLTR+RPLTR**
  - A traditional pipeline for aggregated search framework.
  - A two-layer MLP is used for vertical selection and learning-to-rank method (LTR) for item selection and result presentation
- **BC+Low-level RL+RPLTR**
  - Replace ISLTR with low-level RL method for item selection.
- **NDCG**
  - Normalized Discounted Cumulative Gain
  - Measure up relevance of the selected items to the given query.
- **NDCG-IA**
  - Intent-aware NDCG, an extension on NDCG.

Our HRL Approach

- **Overall Framework**
- **High-level RL: Vertical Selector**
- **Low-level RL : Item Selector**
- **Self-supervised State Representation Learning**

Overall Framework

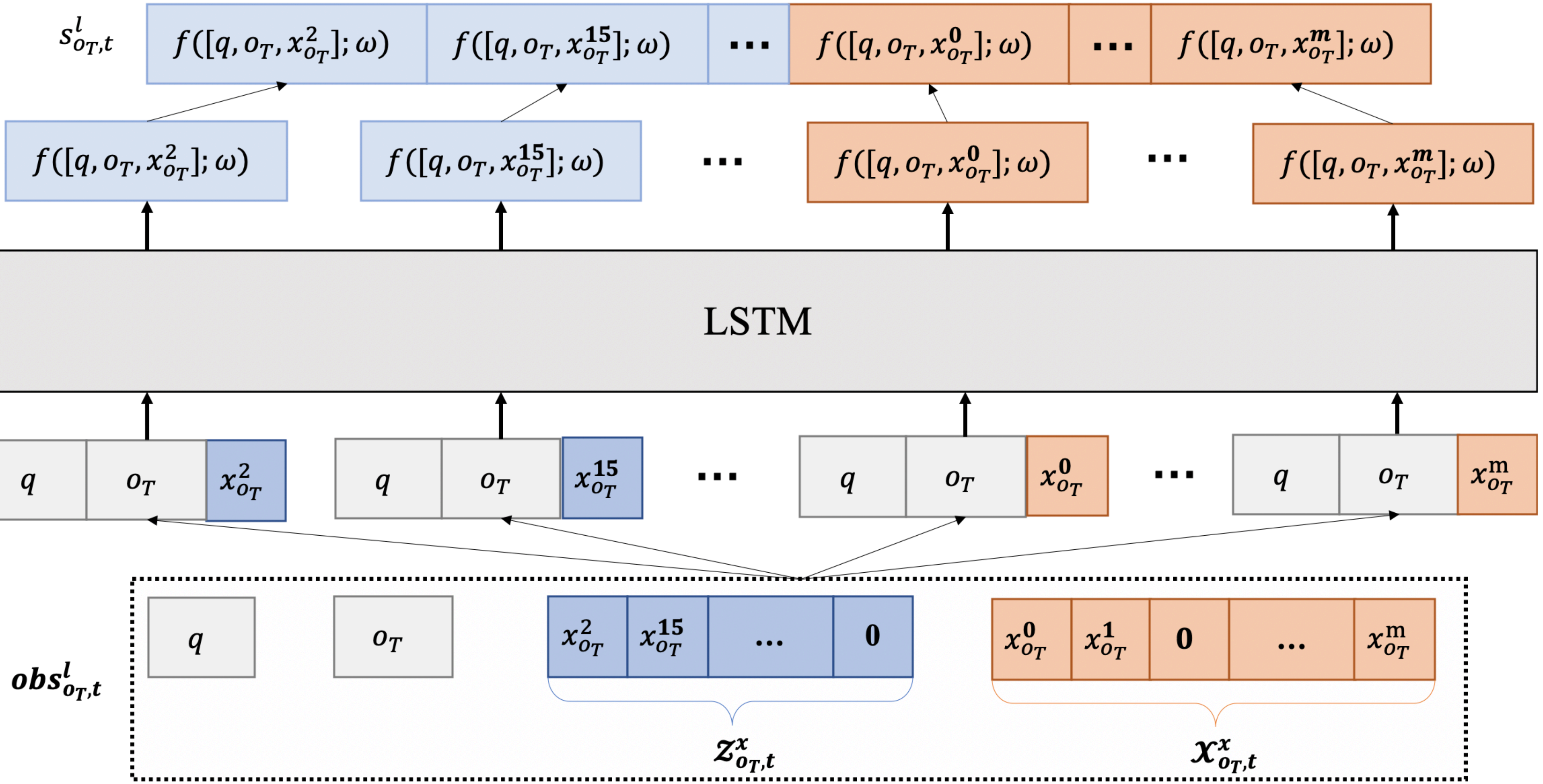


Vertical Selector MDP

- **State $\boldsymbol{\mathcal{S}}$**： Formulated as $s_T^h = f[q, o_T, h_t]$ which includes the query q, current option o_T, and the encoding of history options $(o_0 \dots o_{T-1})$, and selected items by LSTM.
- **Options $\boldsymbol{\mathcal{O}}$**：A vertical domain chosen from candidate set $\mathcal{X}_T^v$.
- **Transition Probability** $\mathbb{P}$：Feed o_T into LSTM to guarantee the MDP property.
- **Reward Signal $\boldsymbol{R}$**: $r_t^h = \alpha \Delta F1 + \beta \Delta NDCG{-}IA + (1-\alpha-\beta) \Delta NDCG$

Item Selector MDP

- **State $\mathcal{S}$**: Formulated as $s_t^l = f[q, o_T, \mathcal{Z}_{o_T,t}^x, \mathcal{X}_{o_T,t}^x]$, which includes query q、the options o_T chosen by high-level RL, partial ranked result $\mathcal{Z}_{o_T,t}^x$ at time t, remaining candidate item set $\mathcal{X}_{o_T,t}^x$ at time t.



Item Selector MDP

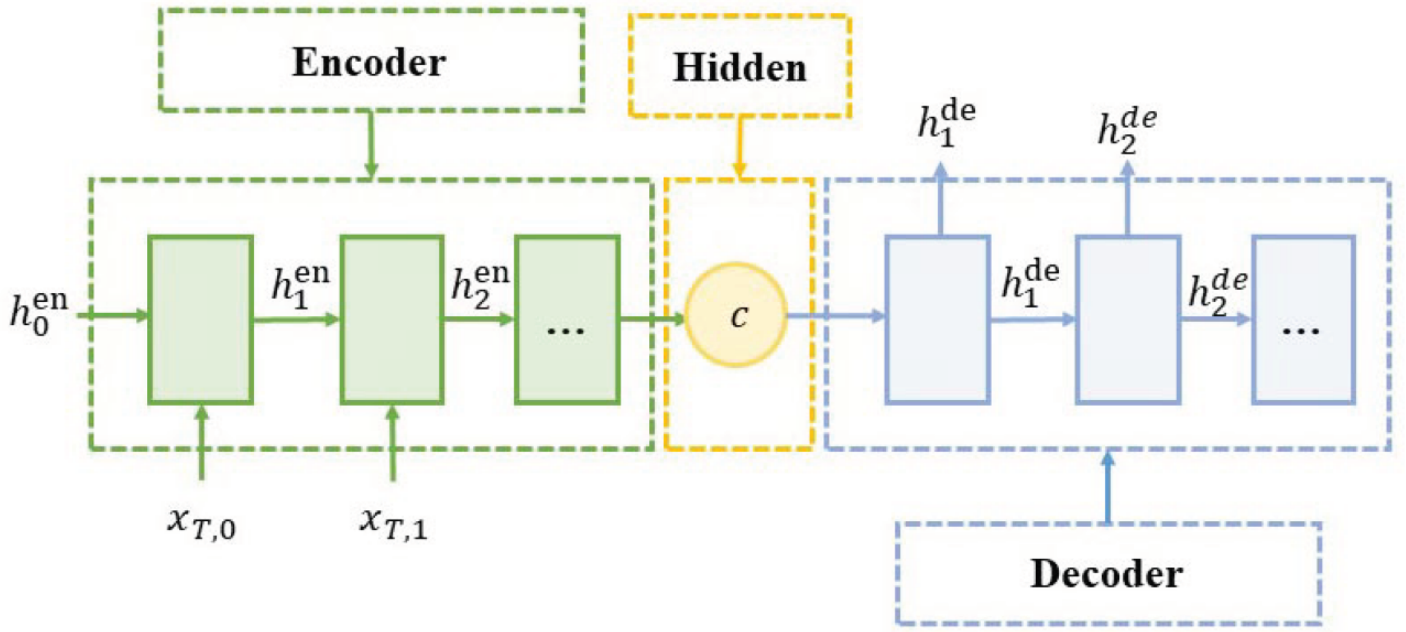
- **Action $\mathcal{A}$** ：Choose the item $x(a_{o_T,t})$ from the candidate set $\mathcal{X}^{x}_{o_T,}$ and rank it onto the t-th position.
- **State Transition** $\mathbb{P}$ ：Add the chosen item $x(a_{o_T,t})$ from candidate set $\mathcal{X}^{x}_{o_T,t}$ to the ranked list $\mathcal{Z}^{x}_{o_T,t}$. The new ranked list and the candidate set are fed into the state representation module to generate the following state $s^{l}_{o_T,t+1}$。
- **Reward Signal *R***： $r^{l}_{o_T,t} = \Delta\text{NDCG}$

Self-supervised State Representation Learning

- **Use the auto-encoder structure to train the State Representation Module**

$$i_{T,t} = \sigma(W^{(i)} x_{T,t} + U^{(i)} h_{T,t-1})$$
$$f_{T,t} = \sigma(W^{(f)} x_{T,t} + U^{(f)} h_{T,t-1})$$
$$e_{T,t} = \sigma(W^{(e)} x_{T,t} + U^{(e)} h_{T,t-1})$$
$$\widetilde{c}_{T,t} = \tanh(W^{(c)} x_{T,t} + U^{(c)} h_{T,t-1})$$
$$c_{T,t} = f_{T,t} \odot c_{T,t-1} + i_{T,t} \odot \widetilde{c}_{T,t}$$
$$h_{T,t} = e_{T,t} \odot \tanh(c_{T,t})$$

$$L(\delta_{en}, \delta_{de}) = \sum_{k=0}^{K} (x_{T,k} - h_k^{de})^2$$



Self-supervised State Representation Learning

- **Alternating Training**： State Representation module and RL are trained alternatively and updated according to their individual loss function.
- **Hybrid loss function**：Sum up the loss function of auto-encoder of State Representation module with that of high-level RL as the unified loss function. Both modules are updated according to the hybrid loss function:

$$L(\theta, \delta_{en}, \delta_{de}) = \log \pi_\theta(o_T|s_T)\, \mathrm{G}_T^h + \beta_{\mathrm{rep}} \sum_{k=0}^{K} \left(I_{T,k} - h_k^{de}\right)^2$$

Experiments

Dataset	Number of Queries	Number of Vertical Domains	Number of Search Engines	Number of Query Results with Relevance Score	Ratio of Labeled Query Results
FedWeb13	50	24	150	32096	17.62%
FedWeb14	50	24	150	34003	17.50%

Baselines

	VS tasks	IR tasks			RP tasks	
	BC	RN	LR	REINFORCE	RN	LR
$BC_{VS} + RN_{IR} + RN_{RP}$	√	√			√	
$BC_{VS} + RN_{IR} + LR_{RP}$	√	√				√
$BC_{VS} + LR_{IR} + RN_{RP}$	√		√		√	
$BC_{VS} + LR_{IR} + LR_{RP}$	√		√			√
$BC_{VS} + REIN_{IR} + RN_{RP}$	√			√	√	
$BC_{VS} + REIN_{IR} + LR_{RP}$	√			√		√

BC = Binary Classifier; RN = RankNet; LR = LambdaRank

Results

Method	NDCG@10	NDCG@20	NDCG-IA@10	NDCG-IA@20
BC + ISRankNet + RPRankNet	26.47	26.18	5.78	6.76
BC + ISLambdaRank + RPLambdaRank	**29.16**	**27.39**	**6.48**	7.07
High-level RL + ISRankNet	20.14	24.72	4.75	6.84
High-level RL + ISLambdaRank	21.20	25.31	4.97	6.72
BC + Low-level RL + RPRankNet	23.00	24.62	4.94	6.46
BC + Low-level RL + RPLambdaRank	22.91	22.12	4.85	5.58
HRL	25.38	25.64	6.34	**8.20**
HRL(without state representation module)	22.56	24.10	4.99	7.10

Table 3: Performance comparison with baseline on dataset FedWeb13

Method	NDCG@10	NDCG@20	NDCG-IA@10	NDCG-IA@20
BC + ISRankNet + RPRankNet	27.89	30.32	7.28	8.95
BC + ISLambdaRank + RPLambdaRank	34.73	33.37	9.09	10.01
High-level RL + ISRankNet	32.83	34.77	8.30	10.19
High-level RL + ISLambdaRank	31.36	34.54	7.81	10.03
BC + Low-level RL + RPRankNet	26.75	29.79	6.32	8.39
BC + Low-level RL + RPLambdaRank	29.54	30.42	6.78	8.21
HRL	**40.77**	**38.69**	**10.83**	**12.94**
HRL(without state representation module)	35.53	35.35	8.70	10.77

Table 4: Performance comparison with baseline on dataset FedWeb14

Results

Method	NDCG@10	NDCG@20	NDCG-IA@10	NDCG-IA@20
HRL(without self-supervised learning)	24.26	24.11	5.90	7.35
HRL(alternative training)	24.36	24.62	5.90	7.51
HRL(hybrid loss training, $\beta_{rep} = 0.1$)	23.28	23.54	5.30	6.79
HRL(hybrid loss training, $\beta_{rep} = 1$)	**25.38**	**25.64**	**6.34**	**8.20**
HRL(hybrid loss training, $\beta_{rep} = 10$)	23.36	24.05	5.80	7.55

Table 5: Performance comparison between different training methods on dataset FedWeb13

Method	NDCG@10	NDCG@20	NDCG-IA@10	NDCG-IA@20
HRL(without self-supervised learning)	36.41	35.73	9.50	11.63
HRL(alternative training)	38.07	36.48	9.99	11.94
HRL(hybrid loss training, $\beta_{rep} = 0.1$)	**40.77**	**38.69**	**10.83**	**12.94**
HRL(hybrid loss training, $\beta_{rep} = 1$)	37.16	36.23	9.93	11.97
HRL(hybrid loss training, $\beta_{rep} = 10$)	37.62	36.46	9.90	11.98

Table 6: Performance comparison between different training methods on dataset FedWeb14

Recap

- We model the aggregated search problem in a novel hierarchical end-to-end manner, the high level for vertical selection and result presentation, while the low level for item selection.
- We introduce hierarchical reinforcement learning to solve this problem. In addition, self-supervised learning based state representation methods are used to strengthen the association among different subtasks.

Thank you！