Learning and adaptivity in interactive recommender systems

Autores: Tariq Mahmood, Francesco Ricci

Pontificia Universidad Católica

October 29, 2015

Overview

1. Motivación
2. Modelo
3. Estudio de un Caso
4. Evaluación
5. Resultados
6. Conclusiones

Motivación

- Los sistemas de recomendación tradicionales se desvían de una manera natural de recomendar.
- Los sistemas de recomendación conversacionales, no son adaptivos.

Solución?

MDPs

Idea de implementación

- La idea es construir un sistema con dos componentes, una que muestra información al usuario y la otra que se dedica a optimizar las recomendaciones hechas al usuario dada sus acciones.
- El sistema decide mostrar o no la peticion hecha por el usuario según lo que lo acerque más a su objetivo.
- El sistema optimiza su estrategia de recomendación para que el usuario alcance su objetivo.

Modelo

Recordemos que un MDP es una tupla $< S, A, T, R >$ donde

- S es el conjunto de estados, que representan las diferentes situaciones que el agente recomendador se puede encontrar mientras interactua con el usuario. Cada estado $s \in S$ es una tupla $< u, r >$ donde u corresponde al estado actual del usuario y r es el estado actual del agente recomendador.
- A es un conjunto de las posibles acciones del sistema. Este conjunto de acciones se relaciona con la política. Diremos que una política π es una funcion $\pi : S \rightarrow A$, en donde para $s \in S$ indica que $\pi(s) = a$ para $a \in A$.

Modelo

- T una función de transición. Donde $T(s, a, s')$ es la probabilidad de terminar en el estado s' si ejecutamos a desde s.
- R una función de recompensa. Donde $R(s, a)$, asigna un valor escalar por cada accion a ejecutada en s. Para efectos del problema, asumiremos recompensa negativa para cualquier estado no terminal y recompensa positiva por terminar en el objetivo.

Reinforcement Learning al Rescate

Policy iteration

La idea del algoritmo de optimización es la siguiente, supongamos que el sistema está en el estado S_0, entonces dada una política inicial arbitraria, el sistema decide ejecutar a_0. En respuesta, el usuario elige alguna accion lo que lleva al sistema al estado S_1 y recompenza al agente con r_0. Es intuitivo querer encontrar la política π tal que se maximize la recompenza.

Policy iteration

Definimos que el valor de un estado s bajo la política π, $V^{\pi}(s)$, como la recompenza acoumulada esperada, cuando el agente comienza en s y sigue π después. Entonces tenemos que:

$$V^{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') V^{\pi}(s')$$

Donde $s' \in S$ es el siguiente estado alcanzado, cuando el agente toma la acción $\pi(s)$ en s. Entonces, para calcular el π^* óptimo se reduce al siguiente problema de optimización:

$$V^{\pi^*}(s) = max_{\pi} E(\sum_{t=0}^{\infty} \gamma^t R_t)$$

Policy iteration

$$V^{\pi^*}(s) = max_a R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^{\pi^*}(s')$$

$$\pi^*(s) = argmax_a R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^{\pi^*}(s')$$

Recomendación adaptativa

Con el modelo dado y una política inicial arbitraria, el agente itera sobre los siguientes pasos en cada ejecución.

- Evaluación de la política: se computa V^{π} como lo señalamos anteriormente.
- Mejora de la política: se utiliza V^{π} para mejorar la política.

Estudio de un Caso

NutKing, es una empresa de viaje que quiere recomendar a sus turistas cual sería su plan de viaje ideal.

- NutKing,utiliza un sistema recomendador conversacional basado en filtrado colaborativo.
- Nutking, imita lo que sería una sesión de recomendación tradicional con el agente de viaje.
- Al comenzar, el usuario indica sus preferencias, estas son utilizadas como restricciones para generar recomendaciones y guíar la busqueda.

Funcionamiento Nutking

Funcionamiento Nutking

Funcionamiento Nutking

- Política rígida no adaptativa.
- Baja aceptación de las sugerencias 28%.
- Posibilidad de mejora.

Cómo construimos el modelo MDP

Primero modelamos el sistema que interactua con el usuario, este consiste en cinco páginas $P = \{S, QF, R, T, G\}$ donde S es la página de inicio donde el usuario inicia sesión, QF es el formulario para generar una consulta, T es la página de ajuste de búsqueda R el resultado y G el objetivo. Un set de seis funciones de información $F = \{go, exec, modq, acct, rejct, add\}$.

Cómo construimos el modelo MDP

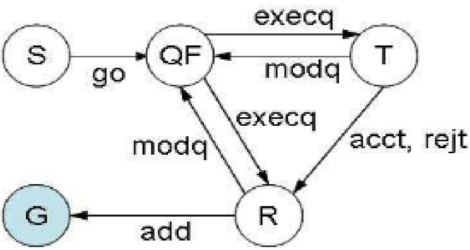

Figure: Modelo de funcionamiento

Cómo construimos el modelo MDP

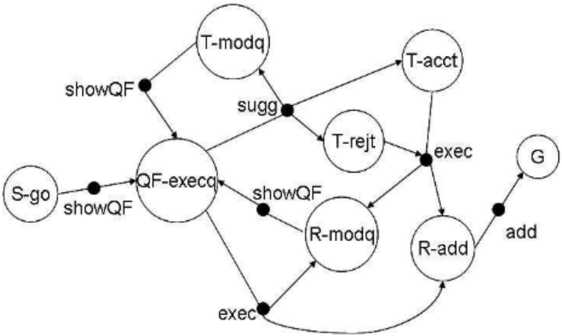

Figure: Diagrama de estados con las acciones del sistema

Cómo construimos el modelo MDP

La función de recompenza es simple, es constante y negativa para cualquier estado no terminal y positiva para cualquier estado objetivo.

Evaluación

Para evaluarlo se utilizaron usuarios simulados, donde se mejoraba la política con Policy Iteration. Esta simulación de usuario corresponde a un usuario que comienza en $S - go$ y esta en búsqueda de un item $t = (v_1, ..., v_n)$. Donde $v_1, ..., v_n$ son los valores usados por el usuario para ajustar la búsqueda.

Modelo de comportamiento del usuario

- Cuando el usuario está en $QF - exec$, como modificara la query actual. El usuario simplemente agrega a la query el siguiente valor del item de prueba.
- Cuando el sistema sugiere un ajuste de query, el usuario puede elegir $T - modq$, $T - acct$ o $T - rejct$, el usuario acepta si alguna de las features está presente en el item. Rechaza si la lista es más pequeña que un threshold y ejecuta la query original. En cualquier otro caso modifica la query
- El usuario agregará al carro si el item de test es encontrado en el top N de los items retornados por la query, sino optará por modificar la consulta.

Aprendizaje de Política

Se estimaron las transiciones de probabilidad con simulaciónes aleatorias de usuarios. Y después se ejecuto iteración de política según lo explicado anteriormente.

Resultados

Se dejó que el algoritmo de evaluación de política corriera por 100 intentos y el algoritmo de iteración por 10 iteraciones.

Transiciones

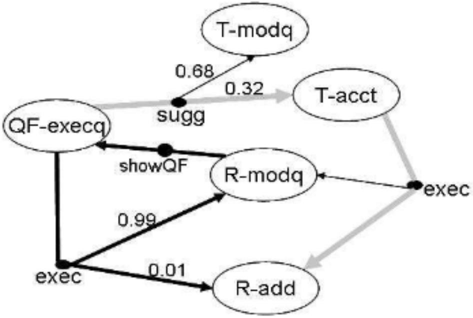

Figure: Transiciones para QF-execq

Politica óptima

	Optimal Actions for states with *pua=QF-execq*					
$cost$	(s,s)	(m,s)	(m,m)	(l,s)	(l,m)	(l,l)
$InitPol.$	exec	exec	exec	sugg	sugg	sugg
-0.01	exec	exec	exec	exec	exec	exec
-0.02	exec	exec	exec	exec	exec	sugg
-0.04	exec	exec	sugg	exec	sugg	sugg
-0.08	exec	sugg	sugg	sugg	sugg	sugg
-0.12	sugg	sugg	sugg	sugg	sugg	sugg

Figure: Política óptima para γ cte.

Política óptima

	Optimal Actions for states with *pua* = *QF-execq*					
γ	(s, s)	(m, s)	(m, m)	(l, s)	(l, m)	(l, l)
Init Pol.	exec	exec	exec	sugg	sugg	sugg
0.9	exec	exec	exec	exec	sugg	sugg
0.7	exec	exec	sugg	sugg	sugg	sugg
0.5	exec	sugg	sugg	sugg	sugg	sugg
0.3	exec	sugg	sugg	sugg	sugg	sugg
0.1	sugg	sugg	sugg	sugg	sugg	sugg

Figure: Política óptima para costo cte.

Conclusiones y Trabajo a futuro

- El sistema efectivamente mejora las políticas tomadas versus un sistema conversacional rígido.
- Se muestra que un sistema de recomendación adaptivo es factible utilizando MDPs.
- Se debe probar este sistema con usuarios reales para ver si efectivamente se mejora la tasa de aceptación.