Name: Mohammad Altahan

Seminar Name: Data Science

Course Code: 63739

Context-Aware Advertisement Recommendation for

High-Speed Social News Feeding

Introduction:

Every one of us I think in this new world, the world of technology, has experienced the social media websites (Facebook, twitter,), beside that I guess most of us has encountered to advertisements, which sometimes could be interesting, meaning full, relevance to our needs and sometimes annoying. But also guess that, somehow you have wondered, how could such an advertisement, be so much relevant to my interests. Maybe you have thought, do they spy on me or stealing my information, but in fact there is calculations happens in the background, you are not aware of it. Which studying what we are interest in and what not. Let us have a look at one of the most powerful models to handle and provide such advertisements in an efficient way.

Yuchen Li, Dongxiang Zhang, Ziquan Lan and Kian-Lee Tan are researchers in the

NUS Graduate School of Integrative Science and Engineering, National University of Singapore. They came up with an idea and a challenge, to develop a model, that is capable of making the user recommendation over the social media, more efficient, real time, less annoying and willing to make the user hit the advertisement icons, which satisfy his needs. Their idea came from the fact, that every person in this life, has his own static interest, so there is a possibility to make a system, that is able to recommend some advertisements for him, but they have discovered that the system will not be so accurate, since the user also has a dynamic interests, which could be also changing, due to the news feed, that he could get from his friends, which it could be somehow changing his interests in a way or another so, their challenge now is to combine the static interest and the dynamic interests into one model, that can recommend the top relevant advertisement, that meet the user’s interests.

But they had to put in their mind, the fact that, this model could be computationally expensive.

Content:

Probably some of you know, that the social media advertisement has become the major revenue, even for the dominators of the markets such as (Facebook and twitter) it is actually a multi-billion-dollar market. For Facebook or twitter, In order to deliver an ads to potently interested user, they have to learn a model to predicate their interests, based on their personal interests, but It is not that efficient, since the user interests are growing slowly, thus the user may end up receiving repetitive ads.

Our group of researchers as we have mentioned before proposed a context aware advertisement framework, that combines the relatively personal interests and the dynamic news feed from friends to increase the possibility, that the user will hit the ads button. For example, when a friend shows the status in hospital, displaying gift delivery ads is a good choice. To do that They have proposed a hybridmodels, which combines the advantages of the online retrieval strategy, which is able to find the most relevant ads matching the dynamic context when a read operation is triggered, and the safe region method which has been developed, to avoid the frequent computations, when the context varies a little and to detected if the top ads has been changed. This hybrid model has been tested on multiple social media, and it has proven, that it is efficient and robustness. Let’s us have a look how they could achieve that. Before of going into the hybrid model details, I would like to give a look on the related works, that the researches introduced, they have studied these works, analysed it, discovered what are the advantages and the disadvantages, to improve the quality of their model. Let us start with the Publish/Subscribe System.

**Pub/Sub System**: A publish/subscribe system is a middleware for matching events, which are generated by data sources (publishers), to subscriptions, which specify the interests of users (subscribers). Traditional publish/subscribe systems only support stateless subscriptions, defined as filters over the contents of individual events. (e.g., stock quotes) against a set of subscriptions (e.g., trader profiles specifying quotes of interest). There are two major difference between this system and the context aware system, since the pub/sub is using Boolean expression matching, which means an event either matches a subscription or it does not, for instance, a stock quote will either match or not match a trader profile, but the problem is maybe, they could be a lot of events, which matching the user subscription, so the user will be end up with so many ads, which will make him so annoyed about them.

In the context aware ad recommendation, will be display only the most relevant ads in the user news feeds. The second difference, that the subscription has been build base on the static interests of the user, however in the context aware, the recommendation has been build based on the combination of the static interests and the dynamic interests which are the content of the news feeds. As these contents are continuously changing, this kind of solutions can’t be applied. Thus, it need another kind of solutions, will able of handling the dynamic user interests.

**Top-K Aggregation Query:** this kind of approaches, consider that each object attribute has it is own score. In order to calculate the total score for an object, they are using a monotonic aggregation function. After that they are using some kind of algorithms, such as the threshold (TA, CA) to obtain the most relevant ads for a user.

**Local immutable region:** which determines immutable regions on individual decision factors. An immutable region there takes the form of a validity interval for an isolated query weight, assuming that all the other weights are kept constant. interval is defined for each decision factor. However, due to the local nature of the LIRs, it cannot support simultaneous readjustments to multiple weights.

**Global immutable region:** The GIR indicates all the possible weight settings for which the current top-k recommendation holds. For the common case of linear scoring functions, the GIR is a convex polytope in query space, wherein the query vector may freely shift without inducing any changes in the result. Unfortunately, GIR is computationally expensive as it takes minutes or even hours to get the valid region for a given query vector with only 5-8 dimensions. This makes GIR infeasible to handle the dynamic nature of social news feeds. To overcome this issue, we design a series of techniques to quickly compute a subspace of GIR so that the maintenance cost is greatly reduced.

**Microblog Search in Social Networks**: Chen et al. Introduced a novel indexing and ranking mechanism for enabling real-time search in microblogging systems such as Twitter. The TI is designed based on the observation that most tweets will not appear in the search results. Therefore, we can significantly reduce the indexing cost by delaying indexing less useful tweets. In essence, the TI classifies the tweets into two types, distinguished tweets and noisy tweets. The TI consists of two indexing schemes: a real-time indexing scheme for distinguished tweets and a background batch indexing scheme for noisy tweets, another works is for Tao et al. Which is a provenance model to capture connections between micro-blog messages. Provenance refers to data origin identification and transformation, and for Li et al who introduce a framework is based on a general ranking function that incorporates time freshness, social relevance and textual similarity. However, these indices are designed to search the microblogs whereas in their case the microblogs are used as queries to retrieve relevant ads. So, it is not useful for them, since the dynamism of the query is not considered.

**Construct the hybrid Model equations and algorithms:**

Let us say, that we have an advertisement database A, their goal is to recommend the most relevant ad from this database, when a user request for his news feed.

And since they can classify the ads into multi-dimensional topic vector (T).

They have studied previous works to measure the relevance between static user interests(profiles) and an ad and they obtain the following equation

|  |  |
| --- | --- |
|  | (1) |

.

Where is the relations between static user profile and ads, rel(u, w) ∈ [0, 1] denotes the relevance between a user u and a topic(w) in T and rel(a, w) denotes the relations between an ad and a topic w in T. but their context aware is also considering the dynamic news feed, when they recommend an ads for a given user.

So, they have used a sliding window to store m most recent posts, to serve as a dynamic context for ad recommendation, so they apply the same topic modelling technique to project each post in the window to the latent topic space and use rel(d,w) ∈ [0, 1] to measure the relevance between a post and a topic.

And they come up with the following equation.

|  |  |
| --- | --- |
|  | (2) |

Where rel(a,w) is the relation between an ad(a) and a topic(w), Where is the relations between dynamic user profile and ads.

You can imagine the overall system like in the figure below.

A close up of a map

Description generated with high confidence

Fig. 1: System Overview of Context-Aware Advertisement Recommendation in Social Networks.

Each user in social media, is either publisher or subscriber, when the user composes, shares or likes a post, we say the user, as a publisher, triggers a write operation. And his post is saved in the database and may later retrieved to appear in his friend’s news feeds. When a user login or refresh his/her news feed, we say the user, as a subscriber, triggers a read operation. Then, the posts from friends are retrieved and sorted chronologically and a sliding window containing m recent unread posts are returned.

finally, they have summed up these two equations into one which is presented by this linear equation.

(u, a) = α · s(u, a) + (1 − α) · d(u, a) (3)

Where α ∈ [0, 1] is a system parameter to balance the importance between personal interests and dynamic context and can be set based on the application requirements.

So, when α is close to 1, the ads recommendation based mainly on the static user profile, when it is 0 then the recommendation based on the dynamic context.

Then they have defined their problems as follows:

**Definition 1:** For any user u, the context-aware ad recommendation finds a set of ads, i.e. R, which has a size of k and satisfies (u, a) ≥ (u, a’) ∀a ∈ R ∧ ∀a’ ∈ A \R.

In the equation (3) they have aggregate the dynamic news feed with the static personal profile, to query the ad database, they have called the aggregated vector context-aware query vector, denoted by Qu.

**ONLINE RETRIEVAL ALGORITHM:**

In the current social media, they have developed models, where they can calculate the top ad for personal interests offline, since the profiles are static. After that they return they return it together with the news feed, when the user request for his new feeds. However, they have to include the dynamic context in the recommendation calculations, therefore they are not able to do the calculation offline, because each write operation, will cause the news feeds for all the user’s friends to vary, which is computationally expensive. The online retrieval algorithm will bring the top k “on the fly”.

If they want to retrieve the most relevant ads to a given user, they have to construct a query vector, which consist of the distribution of user static profile and dynamic context (which consist of the most recent, unread posts from his friends) and scan it against the ads database, but without proper indexing, it will scan all the ads database, to find the most relevant ads with the highest score. Which will be computationally expensive.

To handle this problem effectively, they reconstruct the equation (3) to be like this

(u, a) = α · s(u, a) + (1 − α) · d(u, a) =

**Qu(w**)

where Qu(w) is the aggregated relevance between user u and topic w. now their ranking function is consist of two terms(Qu(w) and rel(a,w)). since rel(a,w) is independent of the dynamic context it could be computed and sorted offline.

They Qu(w) will become constant, if the (u, a) is determined. So, it will not affect the ordering of the rel(a,w). therefore, we could establish |T| inverted lists, sorted by rel(a,w) for each user. So, when a read a operation is triggered, they can we can retrieve the sorted lists and directly apply standard top-k aggregation techniques such as Threshold Algorithm(TA), which works as follows:

1) Perform a sorted access in parallel to each of the |T| sorted lists. For each document accessed, perform a random access to other topics and compute the aggregated score of (u, a). If the computed aggregated score is one of the k highest we have seen so far, remember the ad and its score.

2) For each list Li, let high[i] be the score of the last ad seen under sorted access. Define the threshold value Bk to be the aggregated score of high[i] by the aggregation function (u, a). As soon as at least k ads have been seen whose score is at least equal to Bk, the algorithm terminates. Here is an example where we could understand it better:

**Example 1:** Let the window size m = 3, the weighting parameter α = 0.25 and the number of topics |T| = 2. Given a user u, let Hu = (0.4, 0.6) be the topic distributions

of his/her static interests. Suppose the topic distributions of the three posts in the window are (0.2, 0.8), (0.1, 0.9) and (1.0, 0) respectively. When u triggers a read operation, the context-aware query vector Qu is calculated as

Qu = 0.25 · (0.4, 0.6) + 1−0.25 3 [(0.2, 0.8) + (0.1, 0.9) + (1.0, 0)] =

(0.55, 0.45) = (0.425, 0.575). Suppose Qu is used to query an ad database with four tuples {a1 = (0.3, 0.9), a2 = (0.4, 0.7), a3 = (0.5, 0.8) and a4 = (1.0, 0)}. To support top-k aggregation, we pre-compute two inverted lists lw1 and lw2 for the topics and get lw1 = {(a4, 1.0), (a3, 0.5), (a2, 0.4), (a1, 0.3)} and lw2 = {(a1, 0.9), (a3, 0.8), (a2, 0.7), (a1, 0.0)}. By calling the TA algorithm presented above, a3 will be returned as the most relevant ad if k is set to 1.