Indian Statistical Institute
Kolkata

M.Tech. (Computer Science) Dissertation

# Contextual Suggestion

A dissertation submitted in partial fulfillment of the requirements
for the award of Master of Technology
in
Computer Science

*Supervisor:*
Dr. Mandar Mitra
Computer Vision and
Pattern Recognition
Unit

*Author:*
Suraj Agrawal
Roll No: MTC 1513

**M.TECH(CS) DISSERTATION THESIS COMPLETION CERTIFICATE**

**Student : Suraj Agrawal (MTC1513)**
**Topic : Contextual Suggestion**
**Supervisor : Dr. Mandar Mitra**

This is to certify that the thesis titled "Contextual Suggestion" submitted by Suraj Agrawal in partial fulfillment for the award of the degree of Master of Technology is a bonafide record of work carried out by him under our supervision. The thesis has fulfilled all the requirements as per the regulations of this Institute and, in our opinion, has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other university for the award of any degree or diploma.

**Mandar Mitra**

**Date :** $7^{th}$ *July*, 2017

# Acknowledgements

I would like to thank my dissertation advisor Dr. Mandar Mitra for suggesting this topic to me in the first place, and for helping me with the challenges I have faced.

I would also like to thank all my classmates for always being willing to hold a discussion with me. Their timely help facilitated me in understanding many concepts quicker and better, and the lively discussions have always led to the genesis of new ideas.

# Abstract

In this report we give the approaches that we applied to solve TREC 2016 Contextual Suggestion Track. The goal of the Contextual Suggestion Track is to build a system capable of proposing venues which a user might be interested to visit, using any contextual and personal information. We present our approaches to model Point Of Interests(POI) and user profile based on tags' word embedding(specifically Word2Vec). We also present model for contextual relevance and POI relevance. We also compare different ways to tune the parameters. Our approaches work better than other existing approaches presented in TREC Contextual Suggestion 2016 Track.

# Contents

# List of Tables

# Chapter 1

# Introduction[1]

Contextual Suggestion is a TREC Track that ran during 2012-16. The aim of the track is to investigate search techniques for complex information needs that are highly dependent on context and user interests. According to a report from the Second Strategic Workshop on Information Retrieval in Lorne [2]: 'Future information retrieval systems must anticipate user needs and respond with information appropriate to the current context without the user having to enter an explicit query. In a mobile context such a system might take the form of an app that recommends interesting places and activities based on the user's location, personal preferences, past history, and environmental factors such as weather and time. In contrast to many traditional recommender systems, these systems must be open domain, ideally able to make suggestion and synthesize information from multiple sources.'

For example, imagine a group of Indian Statistical Institute, Delhi students spending their afternoon in Kolkata. The contextual suggestion system can recommend a visit to the Indian National museum, dinner at Peter Cat or a trip to Sunderban. The primary goal of this track was to develop evaluation methodologies for such systems.

## 1.1 Terminology

Point Of Interest(POI) is the attraction that may be interested to the user. For example, Indian Museum, Victoria Palace, Peter Cat, Howrah Bridge, Botanical Garden etc. We use the term POI and place interchangeably.

[1] Most of the content in this chapter is taken from [3]

Endorsement/tags are categories related to a POI. For example, 'Indian Museum' has tags: *Museum*, *History* and 'Academy Of Fine Arts' has tags: *Art galleries*.

Profile is a single user's preferences (list of POIs rated by user with their tags/endorsements), their gender and age. For example a male person heaving age 29 has rated *Victoria Palace* with 3, *Indian Museum* with rating 4 and *bistro* with rating 1 will represent his user profile..

Context is the information about target city(i.e target location) of the trip, trip type, trip duration, type of group the person is travelling with and season of the trip. For example, a person is visiting to Kolkata, with his family on weekend in winter will represent the context.

Details of Profile and Context is described in Section 1.3.3.

## 1.2 Problem Statement

Assume a traveller in a specific context (e.g., a city and trip type) is seeking things to do that reflect their own interests, which is supposed to be inferred from their interests in the given context and a visited city. Given a user's contexts and profile including a Point Of Interest list, their tags/endorsements, and ratings from the visited cities, we need to make recommendations for attractions in a new context (including the target city as the location).

## 1.3 Task Description

As described in the TREC Contextual Suggestion 2016 overview paper, this task has two different phases. In both phase 1 and phase 2 tasks, participants were asked to develop a system that is able to make suggestions for a specific person based on their given profile and context. As input of the task, the track organizers provide a set of profiles, a set of contexts and a set of example suggestions (URLs of pages corresponding to POIs in a given context). Each profile corresponded to a single user's preferences in example suggestions of another context or city, their gender and age, and each context includes information about the target city (i.e., the target location), a trip type, a trip duration, a type of group the person is travelling with, and a season the trip will occur in.

Profiles correspond to the stated preferences of real individuals, who were either recruited through crowd sourcing or as editorial judges.

These assessors first judged example attractions in seed locations, later returning to judge suggestions proposed by the phase 1 participants for various contexts.

### 1.3.1 Phase 1 task

Phase 1 involves a collection based task in which, for each context/profile pair, we were required to develop a contextual suggestion system that is able to make suggestions for a particular user in a specific context. This system is supposed to return a ranked list of 50 suggestions for each profile/context pair. Each suggestion is expected to be relevant to the given profile and the context.

### 1.3.2 Phase 2 task

Phase 2 task is a reranking task, in which a suggestion candidate set is provided for each request.

## 1.4 Test Collection

We used the TREC 2016 contextual suggestion test collection, that consists of a corpus (including TREC contextual suggestion collection and the web corpus), a set of requests, and relevance judgments. In addition the organizers have also released suggestions' endorsements/tags, as described in TREC Contextual Suggestion 2016 overview paper,

### 1.4.1 TREC CS Collection

The collection consists of a set of attractions. For each attraction there are:

- An attraction ID, which contains three parts separated by dashes (-)
  - The string 'TRECCS'
  - An 8 digit number
  - A three digit number corresponding to that attraction's city ID
- A city ID which indicates which city this attraction is in

- A URL with more information about the attraction
- A title

For example, the attraction ‘Eatly NYC’ contains the following:

- Id: TRECCS-00023209-151
- Context: 151
- Url: http://www.eataly.com/nyc/
- Title: Eataly NYC

### 1.4.2 TREC CS Web Corpus

The TREC CS web corpus is a web crawl of the suggestions’ URLs available at the TREC contextual suggestion collection. In this crawl, the organizing team managed to fetch 77.39% of the whole TREC Contextual Suggestion collection, which is 956,437 web pages out of 1,235,844 URLs. This crawl includes web pages from different domains like Yelp, Tripadvisor and Foursquare.

### 1.4.3 Requests

In both phase 1 and phase 2 experiments, each request contains information about assessors’ preferences as profiles and their chosen context. Moreover, phase 2 requests contains suggestion candidates related to each profile and context pair. Each profile consists of a list of attractions the assessor has previously rated, their gender and their age. For each attraction the profile will include:

- A rating:
  - 4: Strongly interested
  - 3: Interested
  - 2: Neither interested or uninterested
  - 1: Uninterested
  - 0: Strongly uninterested
  - -1: Not loaded or no rating given
- Tags/endorsements if it is applicable.

Each context consists of a city name which represents which city the trip will occur in and several pieces of data about the trip. The context is as follows:

- A city the trip will occur in (e.g., Seattle)
- A trip type (e.g., Business)
- A trip duration (e.g., Weekend trip)
- A type of group the person is travelling with (e.g., Travelling with a group of friends as ‘Friends’)
- A season the trip will occur in (e.g., Summer)

An example of a Phase 2 request is given below:

```
{‘id’:743, ‘body’:{‘group’: ‘Friends’, ‘season’:‘Summer’, ‘
   trip_type’:‘Holiday’, ‘duration’:‘Weekend trip’,‘location
   ’:{‘state’:‘TX’,‘id’:306, ‘name’:‘Waco’,‘lat’:31.54933,‘
   lng’:-97.14667},‘person’: {‘gender’: ‘Male’,‘age’: 28,‘id
   ’: 15012,‘preferences’:[{‘rating’:4,‘documentId’:‘TRECCS
   \-00211395\-161’,‘tags’:[‘Beer’,‘Culture’,‘Cocktails’,‘
   Restaurants’,‘Food’,‘pub\-hopping’,‘cocktails’,‘bar\-
   hopping’]},..]}},‘candidates’:[{‘documentId’:‘TRECCS\-002
   67253\-306’,‘tags’:[‘Beer’,‘Cocktails’,‘Family Friendly’,
   ‘Restaurants’, ‘Food’]},{‘documentId’:‘TRECCS\-00294259\-
   306’,‘tags’:[‘Tourism’,‘Bar\-hopping’,‘Restaurants’,‘
   Entertainment’,‘Live Music’]},...]}
```

### 1.4.4 Relevant Judgment

Relevance judgments were collected through crowd sourcing and by the help of a group of graduate students. They were asked to rate suggestions using the same scale as presented in the previous section. However, in the qrels, the raw assessors’ 5 point scale judgments were shifted by -2, making the judgments in the range -3 to 2, and making a score of 1.0 or higher correspond to a ‘interested’ or ‘strongly interested’ judgment, so that the trec-eval can be used to evaluate contextual suggestion runs based on all the common IR measures, included graded measures like NDCG

### 1.4.5 Endorsement Tags

In addition to the relevance judgments based on the ratings, they include endorsements/tags within both profiles and suggestion candidates of the phase 2 requests.

## 1.5 Evaluation Measure

Three measures are used to rank both phase 1 and phase 2 runs. The main measure used is NDCG@5; in addition, P@5 and MRR are also used. They also include measures taking more of the ranking into account, such as P@10, NDCG, MAP, Rprec and bpref.

# Chapter 2

# Related Work

These are the best performing approaches in TREC Contextual Suggestion Track 2016 for Phase 2 task.

## 2.1 DUTH

They collect more information about places from website like Yelp, Foursquare[1].

### 2.1.1 Suggestion Model based on a Weighted k-NN Classifier

They tried to predict a user rating for each candidate venue based on the actual user rating of the k neighbours semantically closer to the candidates. They generate index per user, that collect the data from user preference. Data contains bag of words contains meta data (title, description, keywords), foursquare data(description, title, tags,phrases), Yelp data(description, title, category). They generate query for each candidate venue that contains bag of words of meta data, foursquare profile, Yelp profile as used in indexing. Each query is submitted to the user's index that will return a list of user's preference scored for their semantic similarity to the candidate venue. To compute the rating p for the candidate, they used weighted average of the nearest neighbour rating.

$$p = \frac{\sum_{i=1}^{k} s_i R_i}{\sum_{i=1}^{k} s_i}$$

[1]Most of the content in this section is taken from [5]

, where $s_i$ is tf-idf similarity between the candidate and the i-th neighbour and $R_i$ is the user's rating for the i-th neighbour.

### 2.1.2 Suggestion Model based on Rated Rocchio Method

In this approach they create query per user using Rocchio. They create index per context, containing all POIs in the area of interest. They collect data from metadata, Yelp data, foursquare data in similar manner as did in K Nearest Neighbour. Let there are m term in the training set of all the preference of a user. Consider a document $D_i =< d_{i,1}, d_{i,2}, \ldots, d_{i,m} >$ ,where $d_{i,j} = 1 + log(f_{i,j})$ is the weight of jth term in ith document and $f_{i,j}$ is frequency of jth term in $D_i$ They used following formula to calculate the query:

$$Q = \sum_{j=0}^{4}((j-2)\frac{1}{|R_j|}\sum_{D \epsilon R_j}\vec{D})$$

, where $R_j$ set of places vector rated as j by user. Submit Query Q on the index of the context in which user is interested, that will provide ranked list of POI to the user.

## 2.2 USI

They computed a set of multimodal scores from multiple location based social networks (LBSNs) and combined them with a score that predicts the level of appropriateness of a venue to a given user context. Briefly, the scores are calculated as follows: positive and negative reviews are used to create user profiles to train a classifier which then predicts how much a particular user will like a new venue. Moreover, the frequency-based scores are calculated based on the venue categories and taste keywords. As for the prediction of appropriateness, they created two datasets using crowd sourcing and trained a classifier with the features they extracted from the datasets. A linear combination of all the scores produced the final ranking of the candidate suggestions[2]. This approach consist 5 steps:

[2]Most of the content in this section is taken from [1]

### 2.2.1 Information Gathering

They download page for each venue from foursquare and Yelp. They discard the pages heaving user rating -1(Not Rated) or 2(Neutral rating).

### 2.2.2 User Modeling

To Train the classifier per user they extracted negative samples of negatively rated venues and positive samples of positively rated venues in user profile. Negative samples are the reviews with rating 0 or 1 and positive samples are the reviews with rating 3 or 4 on foursquare page of the venue. They adapt a binary classifier to learn why he/she like or dislike a particular venue. They use tf-idf vector as feature vector for samples(reviews). They use SVM classifier and output SVM decision function is consider as $S_{rev}$. It will tell how close a place is to user relevance.

### 2.2.3 User Model Enrichment

They use frequency based score to enrich the user model. It is based on the assumption that a user visits the venue that she/he likes and rates positively.

They created the positive and negative profile for user based on the category (From foursquare and Yelp page of the venue) of the venues user visited and calculate there normalized frequencies. The new venue is compared with the user's profile to compute similarity score.

Given a user u and her history of rated venues $h_u = \{v_1, \ldots, v_n\}$, each venue has a corresponding list of categories $C(v_i) = \{c_1, \ldots, c_k\}$.

A Positive category Profile is a set of all distinct categories belonging to venues that a particular user has previously rated positively. A Negative-Category Profile is defined analogously for the venues that are rated negatively. They assigned a user-level-normalized frequency value to each category in the positive/negative category profile.

A User-level-Normalized Frequency for an item (e.g., category) in a profile (e.g., positive-category profile) is defined as:

$$cf_u^+(c_i) = \frac{count(c_i)}{\sum_{v_k \epsilon h_u} \sum_{c_f \epsilon C(v_k)} 1}$$

A user-level-normalized frequency for negative category profile, $cf^-$ is calculated analogously.

They created positive/negative category profiles for each user. Let u be a user and v be a candidate venue, then the category-based similarity score $S_{cat}(u, v)$ is calculated as follows:

$$S_{cat}(u, v) = \sum_{c_i \epsilon C(v)} cf_u^+(c_i) - cf_u^-(c_i)$$

Frequency based similarity is calculated for category defined by foursquare($S^F_{cat}$) and Yelp ($S^F_{cat}$).

Venue taste keyword of foursquare is also very informative that is the keywords extracted from tips. They also create taste based positive and negative profile in the similar manner and also calculate the taste based similarity ($S^F_{key}$) in the similar manner.

### 2.2.4 Contextual Appropriateness Prediction

They tried to find out $F_{ap}(v, ux)$, which will tell how appropriate venue v is to the context ux. They assume that the venue can be represented by categories and find the appropriateness of the context to the categories for the venue ($F_{ap,c}(c_i, ux)$). If a venue has category $c = c_1, c_2, \ldots, c_n$, then

$$F_{ap}(v, ux) = min[F_{ap,c}(c_1, ux), \ldots, F_{ap,c}(c_n, ux)]$$

They trained SVM classifier to calculate $F_{ap,c}$. As a training sample they picked 10% of the sample of data set. Each category, context pair is assigned to 3 human assessor. A category is relevant to context if more than 2 human assessor agreed on it. As features for classification, they considered the appropriateness of each venue category to each contextual dimension. Therefore, for all pairs of category-context, we needed to define the appropriateness of the pairs. This is not a trivial task since it could be very subjective. For instance, for a 'family' (group type), it is supposedly not appropriate to visit a 'nightlife spot' (objective). While on a 'business trip' (trip type), visiting a 'pharmacy' depends mostly on the user and other subjective factors. In order to determine how subjective is a pair, they asked human workers to assess the appropriateness of each pair. For each pair they made sure that at least 5 different workers assessed it. The level of agreement between workers was considered as the level of subjectivity of each pair.

The value of $F_{ap}(v, ux)$ is considered as another similarity score in their model known as $S^F_{ctx}$.

### 2.2.5 Suggestion Ranking

They estimate the score between user u and suggestion v as following:

$score(u, v) = \omega_1 S_{rev}(u, v) + \omega_2 S^F_{cat}(u, v) + \omega_3 S^Y_{cat}(u, v) + \omega_4 S_{key}(u, v) + \omega_5 S^F_{ctx}(u, v)$

They use five cross validation for setting parameters $\omega 1 \ldots 5$

## 2.3 UAmsterdam

They tried to efficiently model user profile using neural language modeling and using neural category modeling. They applied Neural language modeling approach to model user profile and use it in a content based filtering model and also tried to learn deep multilayer perceptron to learn category preference[3].

### 2.3.1 User Profiling Using Word Embedding

**Personalized Document Language Model**

They uses tags to create personalized document language model.

Tags in the document d are $TG_u(d) = \{tg_1, tg_2, \ldots, tg_n\}$,

Terms in the document d = $\{t_1, t_2, \ldots, t_m\}$

probability of term t occurring in document d,

$$P(t|\theta_d) = \frac{tf(t, d)}{|d|}$$

Then personalized document model $\theta_{du}$ using tags $TG_u(d)$ as follows:

$$P(t|\theta_{d_u}) = \frac{\sum_{tg \epsilon TG_u(d)} P(t|\theta_d) P(t|tg)}{|TG_u(d)|}$$

$P(t|tg)$ is computed using the cosine similarity between the two word embedding vectors corresponding to term t and tag tg.

**Constructing User Profiles**

They estimate user document model denoted as $\theta_u$ as raw probabilistic estimation for each term in a user vocabulary, then for each term t in

[3]Most of the content in this section is taken from [4]

user vocabulary, we estimate its probability as follows:

$$P(t|\theta_u) = \frac{\sum_{d\epsilon D_u} P(t|\theta_{du})}{|D_u|}$$

They used KL divergence between user profile and suggestion candidate to rank them.

### 2.3.2 Neural Category Preference Modelling

In this method they consider relevant suggestion problem as binary classification problem. They learn deep neural network with 4 hidden layer heaving 478 units to predict relevant suggestion candidate to the given user profile and context by the help of user's category preferences. They create a profile of each given category in the contextual suggestion. They find KL divergence of suggestion profile with category profile, that will give feature vector of 123 dimension since there are 123 category. This feature vector is used in deep neural network.

## 2.4 Result

| | NDCG@5 | P@5 | MRR | NDCG | MAP | bpref | P@10 | Rpref |
|---|---|---|---|---|---|---|---|---|
| DUTH_rocchio | 0.3306 | 0.4724 | 0.6801 | 0.6835 | 0.4497 | 0.4704 | 0.4552 | 0.4245 |
| USI5 | 0.3265 | 0.5069 | 0.6796 | 0.6804 | 0.4590 | 0.4507 | 0.4603 | 0.4177 |
| DUTH_bcf | 0.3259 | 0.4724 | 0.5971 | 0.6829 | 0.4606 | 0.4845 | 0.4431 | 0.4321 |
| USI4 | 0.3234 | 0.4828 | 0.6854 | 0.6813 | 0.4576 | 0.4494 | 0.4552 | 0.4229 |
| DUTH_knn | 0.3116 | 0.4345 | 0.6131 | 0.6763 | 0.4456 | 0.4825 | 0.4448 | 0.4189 |
| UAmsterdamDL | 0.2824 | 0.4448 | 0.5924 | 0.6544 | 0.4168 | 0.4452 | 0.4310 | 0.3881 |

Table 2.1: Results of Related Work

# Chapter 3

# Our Work

## 3.1 Problem Definition

As per our discussion in the Introduction, contextual suggestion should provide interesting places based on the context and user's preference. We focus on the Phase 2 task (reranking task), where we need to rerank candidate suggestions. A request file for reranking contains many user profiles with context information like trip type, season, group with whom one is travelling, duration, name of place where user is, latitude, longitude, gender, age. Along with the context information, it also contains preferences that contain places rated by the user, rating given by user and endorsed tags for that place. Our main objective is to rerank the candidates that are also given in the user profile. Each place in the candidate list contains a documentId as well as endorsed tags corresponding to that place.

## 3.2 Motivation of Approaches

Our hypothesis is that tags are able to represent a place as well as user preferences. So we tried to use tags for our purpose. Some examples of the tags are *bar-hopping*, *romantic*, *healthy food*, *sky diving*, *coffee* etc. There could be a relationship between tags, such as *'healthy food'* and *'yoga'*, because it is possible a person interested in healthy food is health conscious, so he might also like yoga. We are motivated to capture such a relationship between tags.

## 3.3 Modelling Tags

We need to model tags in a way that captures relationships between tags. Each POI has a set of tags that will represent that POI. Similar kind of POI has some tags common and uncommon tags are related. For example, POI 1 has tags: *Outdoor activities*, *family friendly*, *parks*, *entertainment* and POI 2 has tags: *Outdoor Activities*, *Citywalks*, *Scenery*. The person interested in *outdoor activities*, *parks* will also be interested in *citywalks* and *scenery*. So we find that *citywalks*, *parks*, *scenery* are related.

We use word embedding( specifically Word2Vec[6]) to capture this relationship, that will try to predict a tag of POI based on other tags of that POI, so the tags that were common between two POI will provide close vector to uncommon tags.

To train Word2Vec, we consider all the tags assigned to a particular POI as a single sentence. For example tags *Beer*, *Tourism*, *Outdoor Activities*, *Culture*, *History*, *Family Friendly*, *Food*, *Parks*, *Entertainment*, *Live Music* are assigned to the document. The sentence corresponding to this place will be 'beer tourism outdooractivities culture history familyfriendly food parks entertainment livemusic'. We create sentences for all the POIs in the request file and train word2vec. This will give an embedding corresponding to each tag. We expect related tags(for example, *healthy food* and *yoga*, *romantic* and *boating*) will have close vectors. We trained word2vec using two datasets: (i) The TREC Contextual Suggestion 2016 Phase 2 request file, and (ii) TREC Contextual Suggestion 2015 as well 2016 request file.

## 3.4 Modelling POIs

We create a POI vector by summing the vectors of all tags corresponding to that POI. Let POI $P_i$ have tags $TG(P_i) =< tg_1, tg_2, \ldots >$, then the vector corresponds to $P_i$ is:

$$\vec{P_i} = \sum_{tg \epsilon TG(P_i)} \vec{tg}$$

## 3.5 Modelling Positive, Neutral and Negative User Profiles

Users rate the POIs on a scale of 0 to 4. The POI with rating greater than 2 are relevant or positive, rating 2 denotes neutral and a rating of less than 2 is negative. Positive, neutral and negative profile can be modeled in two ways. All positive (likewise negative) profiles may be assigned equal importance. Alternatively, we may choose to assign greater importance to strongly positive profiles (i.e. those rated 4) than to weakly positive profiles (i.e. those rated 3); and likewise for strongly and weekly negative profiles.

### 3.5.1 Unweighted

The positive profile for a user is created by summing vectors corresponding to POIs relevant to the user. Let the relevant POIs for user $u$ be: $prof^{+}(u) =< P_1, P_2, ... >$. Then

$$\overrightarrow{prof^{+}(u)} = \sum_{p \epsilon prof^{+}(u)} \vec{p}$$

Similarly we can get a negative profile vector $\overrightarrow{prof^{-}(u)}$, as well as a neutral profile vector $\overrightarrow{prof^{o}(u)}$ by taking the neutral and irrelevant POIs to user.

### 3.5.2 Weighted

The positive profile may be created in the following manner. Let relevant POIs to user $u$ be: $prof^{+}(u) =< P_1, P_2, ... >$. Then

$$\overrightarrow{prof^{+}(u)} = \sum_{p \epsilon Prof^{+}(u)} \vec{p} * rating(p)$$

Similarly we can get a negative profile vector $\overrightarrow{prof^{-}(u)}$ and neutral profile vector $\overrightarrow{prof^{o}(u)}$ by taking neutral and irrelevant POIs to user.

## 3.6 Ranking Method

There are two ranking methods that we used.

### 3.6.1 True Rocchio

First we take a the linear combination of positive user profile vector, neutral profile vector and negative profile vector to create an overall user profile vector[7] as follows:

$$\overrightarrow{prof(u)} = \alpha * \overrightarrow{prof^{+}(u)} + \beta * \overrightarrow{prof^{o}(u)} + \gamma * \overrightarrow{prof^{-}(u)}$$

where $\alpha, \beta, \gamma$ are parameters to tune. Then, candidate suggestions are ranked based on the cosine similarity between user profile vector $\overrightarrow{prof(u)}$ and the POI vector $\vec{P_i}$.

### 3.6.2 Cosine Similarity

Instead of creating of a single overall profile vector, we compute the cosine similarity of the POI vector with positive profile vector, neutral profile vector and negative profile vector. Let the cosine similarity scores obtained be

$$sim^{+}(u, P_i), sim^{o}(u, P_i), sim^{-}(u, P_i)$$

$$sim^{+}(u, P_i) = \frac{\overrightarrow{prof^{+}(u)}.\vec{P_i}}{|\overrightarrow{prof^{+}(u)}| * |\vec{P_i}|}$$

Similarly we can get $sim^{-}(u, P_i)$ and $sim^{o}(u, P_i)$ This will provide a 3-dimensional similarity vector for user $u$ and POI $P_i$.

$$\overrightarrow{sim(u, P_i)} =< sim^{+}(u, P_i), sim^{o}(u, P_i), sim^{-}(u, P_i) >$$

We rank the candidate suggestions using the cosine similarity between the parameter vector $< \alpha, \beta, \gamma >$ and similarity vector $\overrightarrow{sim(u, P_i)}$.

## 3.7 Modelling Context Relevance

We also tried to use a classifier to find if a POI is relevant to the context. For example, ‘Backstage’ is a pub that is not relevant to visit with family.

We create training data using TREC Contextual Suggestion 2015 request file and qrels. It contains two context information: (i) Season (winter, summer, autumn, spring), and (ii) Group (friends, family, alone, others).

The features of the classifier are POI vector and a 8 dimensional one hot encoded vector representing season, group defined as following:

$$<' winter', `summer', `autumn', `spring', `friends', `family', `alone', `others' >$$

We use kNN as a classifier with k=5.

## 3.8 Modelling POI Relevance

We also tried to use classifier to find if a POI is irrelevant to a user's profile, so we can rank them at last. We trained a classifier per user profile for getting relevancy of a POI to user. Training data contains the POIs rated by the user. The POIs with rating greater than or equal to 2 are considered to be relevant and POIs with rating 0, 1 are irrelevant. We tried k-NN as well as SVM as a classifier.

We tried many approaches using various ways to combine the model discussed before with different ways to find the parameters $\alpha, \beta, \gamma$. These approaches are discussed in next chapter.

# Chapter 4

# Approaches

Approaches can be classified in four ways on the basis of different combination of model described in Chapter 3.

There are 58 requests present in the TREC Contextual Suggestion 2016 query file. We divide the query set in 4 parts: Two parts contain 14 queries, and other 2 contain 15 queries. For 4-cross validation, 3 parts are used to tune parameters, and one part is used to evaluate the result.

## 4.1 User profile model with Ranking

We created user profiles using weighted as well as unweighted profile model and ranked the candidate suggestions based on true rocchio based method as well as cosine similarity based method. Using 4 cross validation on TREC CS 2016 qrel, unweighted user profile with cosine similarity based method work best, when we consider $\beta$ to be 1.0 and tune $\alpha, \gamma$. We consider $-4.0 <= \alpha <= 4.0$ and $-4.0 <= \gamma < 4.0$ and exhaustively check for each $\alpha, \gamma$ in interval of 0.2. We choose $\alpha, \gamma$ that will maximize NDCG score on train fold.

The result of user profile model with ranking are following:

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | Avg. NDCG@5 |
|---|---|---|---|---|---|
| Unweighted Cosine Similarity based | 0.2822 | 0.4412 | 0.2867 | 0.3818 | 0.3479 |
| Unweighted Rocchio based | 0.2909 | 0.2745 | 0.2391 | 0.3301 | 0.2836 |
| Weighted Cosine Similarity based | 0.2276 | 0.4914 | 0.2796 | 0.3150 | 0.3284 |
| Weighted Rocchio based | 0.3113 | 0.2480 | 0.2416 | 0.2623 | 0.2658 |

Table 4.1: NDCG@5 of user profile with ranking method

## 4.2 User profile, context model with Ranking

In this approach, candidate suggestions are divided into two groups based on context classifier output. Group 1 contains all the relevant candidate suggestion classified by context relevance classifier and the remaining are put in group 2. Individual groups are ranked separately, and group 1 candidates are ranked before group 2 candidates.

For creating user profile we try both weighted as well as unweighted method with true rocchio based ranking as well as cosine similarity based ranking.

The result of unweighted user profile, context model with ranking method are following:

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | Avg. NDCG@5 |
|---|---|---|---|---|---|
| Cosine Similarity based | 0.2586 | 0.3625 | 0.3044 | 0.2848 | 0.3025 |
| Rocchio based | 0.3254 | 0.3330 | 0.2724 | 0.2925 | 0.3058 |

Table 4.2: NDCG@5 of unweighted user profile, context model with ranking method

None of these combinations in this method did as well as Unweighted Profile model with cosine similarity base ranking method described in Section 4.1. We use the same method described before to tune parameter $\alpha, \beta, \gamma$.

## 4.3 User profile, POI relevancy with Ranking

In this approach, candidate suggestions are divided into two groups based on POI relevance model. Group 1 contains all the relevant candidate suggestion classified by POI relevance classifier and the remaining are put in group 2. Individual group are ranked separately, and group 1 candidates are ranked before group 2 candidates.

For creating user profile we try both weighted as well as unweighted method with true rocchio based ranking as well as cosine similarity based ranking.

The result of unweighted user profile, POI relevancy model with ranking method are following:

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | Avg. NDCG@5 |
|---|---|---|---|---|---|
| Cosine Similarity based | 0.2348 | 0.4490 | 0.2787 | 0.3493 | 0.3279 |
| Rocchio based | 0.2453 | 0.2917 | 0.2325 | 0.3230 | 0.2731 |

Table 4.3: NDCG@5 of unweighted user profile, POI relevancy model with ranking method

None of these combinations in this method did as well as Unweighted Profile model with cosine similarity based ranking method described in Section 4.1. We use the same method described before to tune parameter $\alpha, \beta, \gamma$.

## 4.4 User profile, POI relevancy, context model with Ranking

The candidate suggestions were divided into three groups based on the context model output and POI relevance model output define in Chapter 3. Group 1 contains candidate suggestions for which the context classifier and POI relevance classifier both output 1. Group 2 contains candidate suggestions for which exactly one of the classifiers (context/POI relevance) gives output 1. Group 3 contains candidates for which no classifier gives output 1.

Individual groups are ranked separately. For creating user profile we try both weighted as well as unweighted method. For ranking, we use true rocchio based ranking as well as cosine similarity based ranking. The ranked list contains group 1 candidates then group 2 candidates followed by group 3 candidates.

The result of unweighted user profile, POI relevancy, context model with ranking method are following:

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | Avg. NDCG@5 |
|---|---|---|---|---|---|
| Cosine Similarity based | 0.2962 | 0.3437 | 0.2959 | 0.2903 | 0.3067 |
| Rocchio based | 0.2864 | 0.3414 | 0.2817 | 0.2639 | 0.2928 |

Table 4.4: NDCG@5 of unweighted user profile, POI relevancy, context model with ranking method

## 4.5 Learning Parameters using Linear Regression with 4 Cross Validation

Learn parameter $\alpha, \beta, \gamma$ using linear regression. Three dimension feature vector is generated for train fold using the cosine similarity of positive profile and candidate suggestion, neutral profile and candidate suggestion, negative profile and candidate suggestion, while output is the rating given by user in qrel file. Then train linear regression on train fold. Then we calculate the ndcg@5 for test fold, using the parameter learn by linear regression.

We apply this method for all 4 approaches described before. We tried both weighted and unweigthed profile model.

The result are following.

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | AVG NDCG@5 |
|---|---|---|---|---|---|
| Profile Only | 0.3201 | 0.4845 | 0.2970 | 0.3585 | 0.3650 |
| Profile with Context | 0.3192 | 0.4238 | 0.3280 | 0.3038 | 0.3437 |
| Profile with Place Relevance | 0.3166 | 0.4795 | 0.3012 | 0.4015 | 0.3737 |
| Profile,Place Relevance and Context | 0.3084 | 0.4249 | 0.2774 | 0.2956 | 0.3265 |

Table 4.5: NDCG@5 of unweighted Cosine Similarity based method with learning parameter using Linear Regression

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | AVG NDCG@5 |
|---|---|---|---|---|---|
| Profile Only | 0.2744 | 0.4082 | 0.2687 | 0.3789 | 0.3523 |
| Profile with Context | 0.3111 | 0.4323 | 0.2664 | 0.3425 | 0.3380 |
| Profile with Place Relevance | 0.2963 | 0.4054 | 0.2502 | 0.4003 | 0.3330 |
| Profile,Place Relevance and Context | 0.3277 | 0.4232 | 0.2629 | 0.3624 | 0.3440 |

Table 4.6: NDCG@5 of weighted Cosine Similarity based method with learning parameter using linear regression

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | AVG. NDCG@5 |
|---|---|---|---|---|---|
| 2016 request word2vec | 0.3066 | 0.2291 | 0.2316 | 0.2887 | 0.2640 |
| 2015 and 2016 request word2vec | 0.2909 | 0.2745 | 0.2391 | 0.3301 | 0.3058 |

Table 4.7: NDCG@5 unweighted True Rocchio base method only consider profile

| | test fold 1 | test fold 2 | test fold 3 | test fold 4 | AVG. NDCG@5 |
|---|---|---|---|---|---|
| 2016 request word2vec | 0.2621 | 0.4752 | 0.2659 | 0.3068 | 0.3100 |
| 2015 and 2016 request word2vec | 0.2822 | 0.4412 | 0.2867 | 0.3818 | 0.3479 |

Table 4.8: NDCG@5 unweighted Cosine Similarity base method only consider profile

Our hypothesis is that more data of the tags corresponding to POIs will increase the performance. As per the result shown in the Table 4.7 and Table 4.8, NDCG@5 is lesser with Word2Vec trained on 2016 request file than other. So we can conclude that Word2Vec trained on 2015 and 2016 request file will works better and the result supports our hypothesis.

## 4.6 Tuning Parameters

In this case we use same parameter for all the user profile, this means every user is given same weights to positive, neutral and negative profile. It might also be possible that different users have different weights to positive, neutral and negative profile. This can be captured if we can tune parameter per profile.

We can also use some other ways to tune same parameters for all profile without cross validation. We will use the places rated by user in the request file to tune the parameter instead of train fold. We will set the parameter such that ndcg score of the places ranked by the contextual suggestion system will be maximize.

From the result of cross validation we conclude that unweighted profile model using Word2Vec trained on 2015, 2016 request file is better than any other model, so we will show the result of unweighted profile model with ranking method for further tuning.

### 4.6.1 Parameter Per User Profile

To capture the individual bias toward positive profile, neutral profile and negative profile, we will tune parameter per profile. The following are the result:

| | NDCG@5 | P@5 | MRR | NDCG | MAP | bpref | P@10 | Rpref |
|---|---|---|---|---|---|---|---|---|
| True Rocchio Base | 0.3323 | 0.4759 | 0.6440 | 0.6799 | 0.4597 | 0.4799 | 0.4586 | 0.4239 |
| Cosine Similarity Base | 0.3507 | 0.5000 | 0.6496 | 0.6826 | 0.4590 | 0.4687 | 0.4603 | 0.4203 |

Table 4.9: Parameter Per User Profile Model

### 4.6.2 Same parameter for all profiles

Result of unweighted user profile ranking method with same parameter for all profile with True Rocchio and cosine similarity base ranking method.

| | NDCG@5 | P@5 | MRR | NDCG | MAP | bpref | P@10 | Rpref |
|---|---|---|---|---|---|---|---|---|
| True Rocchio Base | 0.3908 | 0.5621 | 0.7248 | 0.7017 | 0.4808 | 0.4898 | 0.4948 | 0.4375 |
| Cosine Similarity Base | 0.3674 | 0.5241 | 0.6326 | 0.6912 | 0.4661 | 0.4862 | 0.4948 | 0.4265 |

Table 4.10: Same Parameter for All User Profile Model

## 4.7 Result Analysis

We try with many combinations of models and different ways to tune parameters. Methods using Word2Vec trained on 2015, 2016 request file works better than Word2Vec with 2015 request file. The Method using unweighted profile model performs better than weighed profile model. True Rocchio Base method considering same parameter for all query, works finest in all as shown in Table. This beat all the previous method in all measure by margin. And other methods also has better ndcg@5 than previous methods.

| | NDCG@5 | P@5 | MRR | NDCG | MAP | bpref | P@10 | Rpref |
|---|---|---|---|---|---|---|---|---|
| True Rocchio Base Same Parameter | 0.3908 | 0.5621 | 0.7248 | 0.7017 | 0.4808 | 0.4898 | 0.4948 | 0.4375 |
| Cosine Similarity Base Same Parameter | 0.3674 | 0.5241 | 0.6326 | 0.6912 | 0.4661 | 0.4862 | 0.4948 | 0.4265 |
| True Rocchio Base Parameter Per User | 0.3323 | 0.4759 | 0.6440 | 0.6799 | 0.4597 | 0.4799 | 0.4586 | 0.4239 |
| Cosine Similarity Base Parameter Per User | 0.3507 | 0.5000 | 0.6496 | 0.6826 | 0.4590 | 0.4687 | 0.4603 | 0.4203 |
| DUTH_rocchio | 0.3306 | 0.4724 | 0.6801 | 0.6835 | 0.4497 | 0.4704 | 0.4552 | 0.4245 |
| USI5 | 0.3265 | 0.5069 | 0.6796 | 0.6804 | 0.4590 | 0.4507 | 0.4603 | 0.4177 |
| UAmsterdamDL | 0.2824 | 0.4448 | 0.5924 | 0.6544 | 0.4168 | 0.4452 | 0.4310 | 0.3881 |

Table 4.11: Result Comparison of All Run

# Chapter 5

# Conclusion

In this technical report we present our methodologies for TREC Contextual Suggestion task. We showed our approaches to model users profile and POIs based on tags' word embedding, are very effective for reranking task. The result showed that tags' word embedding (specifically Word2Vec) is able to capture the relationship between tags. The result also indicates that increasing the data about POIs' tags will also increase the performance.

## 5.1 Future Work

Here we find the relationship between tags according to places. We can also use tags for modelling user's positive profile, negative profile and neutral profile using Word2Vec. In which three Word2Vec will be trained to capture tags relationship according to positive profile, negative profile and neutral profile.

We could also explore Phase 1 solution using tags by applying these approach. For doing it we need to find good way to extract tags from web pages.

We can also explore more technique to tune parameter so that it will not overfit with less number of training example and can work well while tuning parameter per profile.

# Bibliography


[1] Mohammad Aliannejadi, Ida Mele, and Fabio Crestani. Venue appropriateness prediction for contextual suggestion. In *TREC*, 2016.

[2] James Allan, Bruce Croft, Alistair Moffat, and Mark Sanderson. Frontiers, challenges, and opportunities for information retrieval: Report from swirl 2012 the second strategic workshop on information retrieval in lorne. In *ACM SIGIR Forum*, volume 46, pages 2–32. ACM, 2012.

[3] Seyyed Hadi Hashemi, Charles LA Clarke, Jaap Kamps, Julia Kiseleva, and Ellen Voorhees. Overview of the trec 2016 contextual suggestion track. In *Proceedings of TREC*, volume 2016, 2016.

[4] Seyyed Hadi Hashemi, Jaap Kamps, and Nawal Ould Amer. Neural endorsement based contextual suggestion. In *TREC*, 2016.

[5] Georgios Kalamatianos and Avi Arampatzis. Recommending points-of-interest via weighted knn, rated rocchio, and borda count fusion. In *TREC*, 2016.

[6] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[7] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Readings in information retrieval*, 24(5):355–363, 1997.