Hashtag Recommendation for Tweets using External Knowledge

PRESENTATION OUTLINE

- Introduction
- Data Extraction And Preprocessing
- Algorithms For Recommending Hashtags
- Learning to Rank Framework

INTRODUCTION

- Twitter is a popular platform for users to share personal activities to their friends/followers in form of tweets.
- Tweets are generally short texts, and usually contains hashtags to provide topical or contextual information. For this reason, hashtags are frequently used as queries to search for tweets about a topic or a specific event (e.g., #NBA and #sigir2014).
- However, many tweets do not contain any hashtags.

INTRODUCTION

- Many traditional techniques have been proposed to provide hashtags for tweets, which mostly relies on text similarity with other tweets.
- As short tweets do not provide sufficient term co-occurrence info, traditional text matching techniques have several limitations.

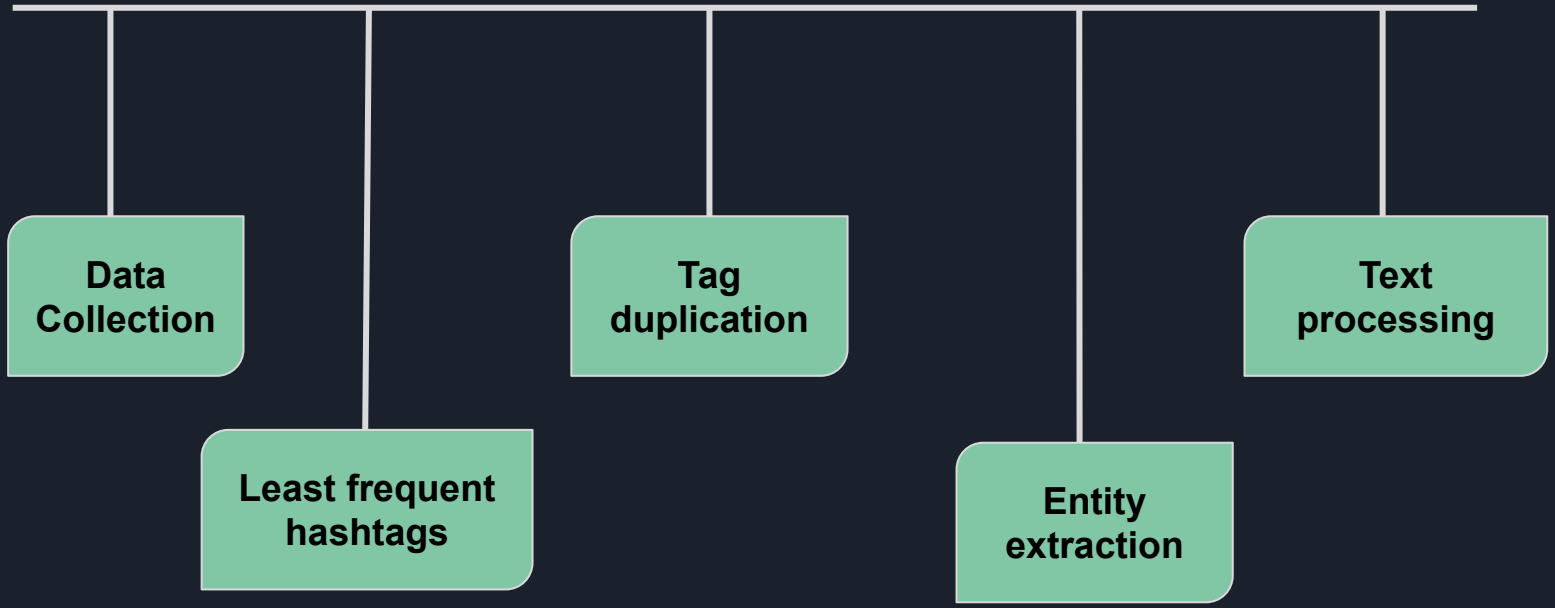
INTRODUCTION

- In this paper, we propose an effective technique to recommend hashtags for tweets by using external data.
- External data refers to information from external sources, which have similar contextual meaning to the tweet in consideration.
- We used several methods for finding these external sources which uses techniques like word embedding, similarity measures, Topical similarity, entity-hashtag relations etc.

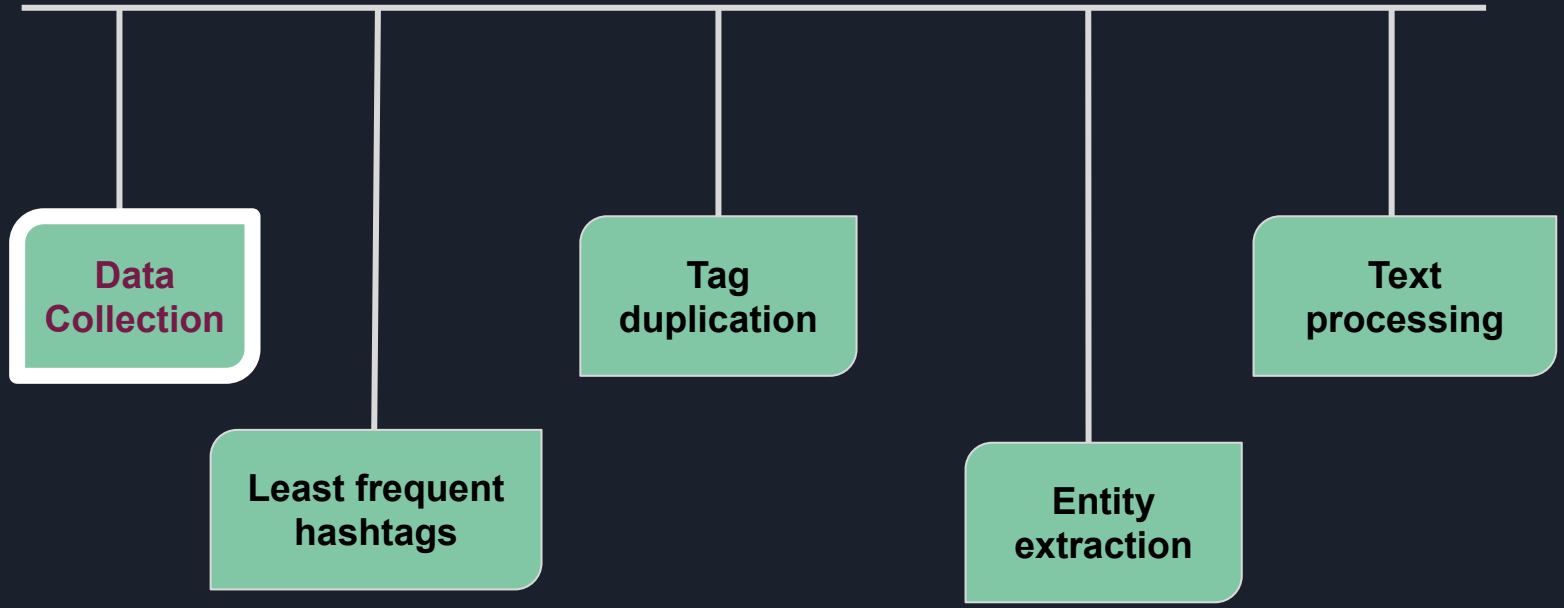
PRESENTATION OUTLINE

- Introduction
- **Data Extraction And Preprocessing**
- Algorithms For Recommending Hashtags
- Learning to Rank Framework

DATA EXTRACTION AND PREPROCESSING



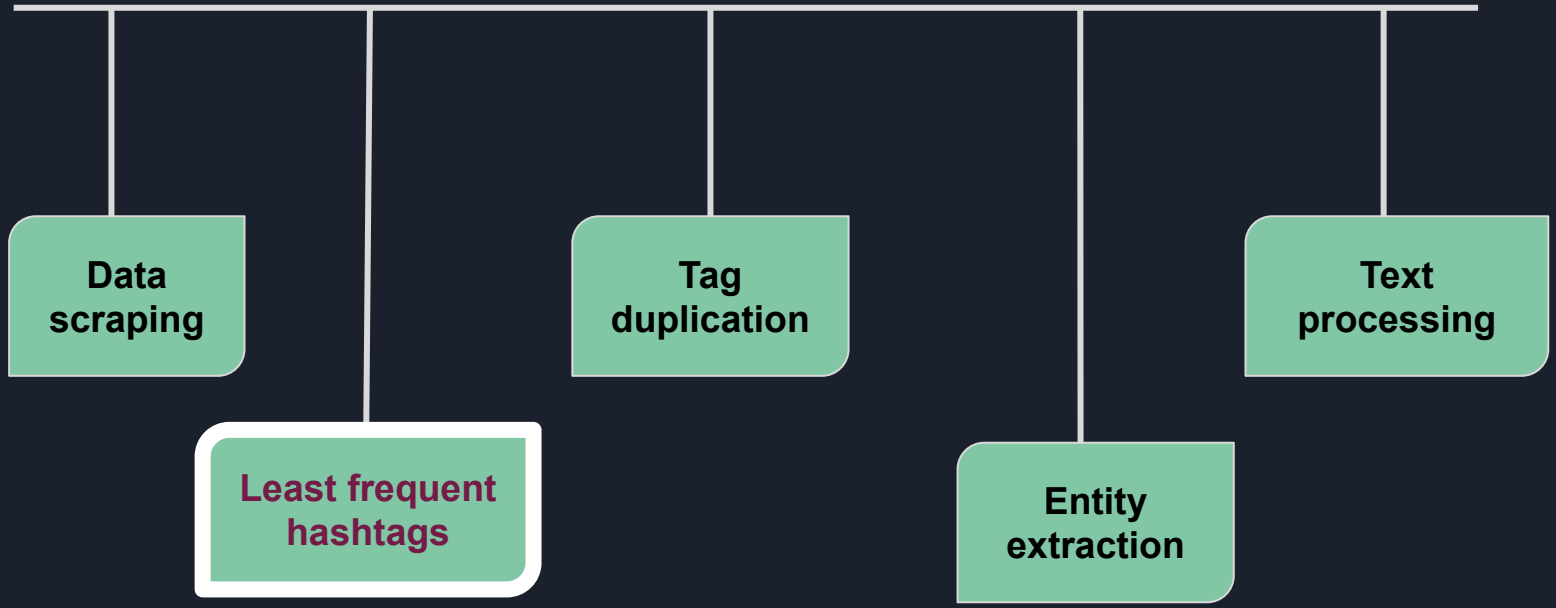
DATA EXTRACTION AND PREPROCESSING



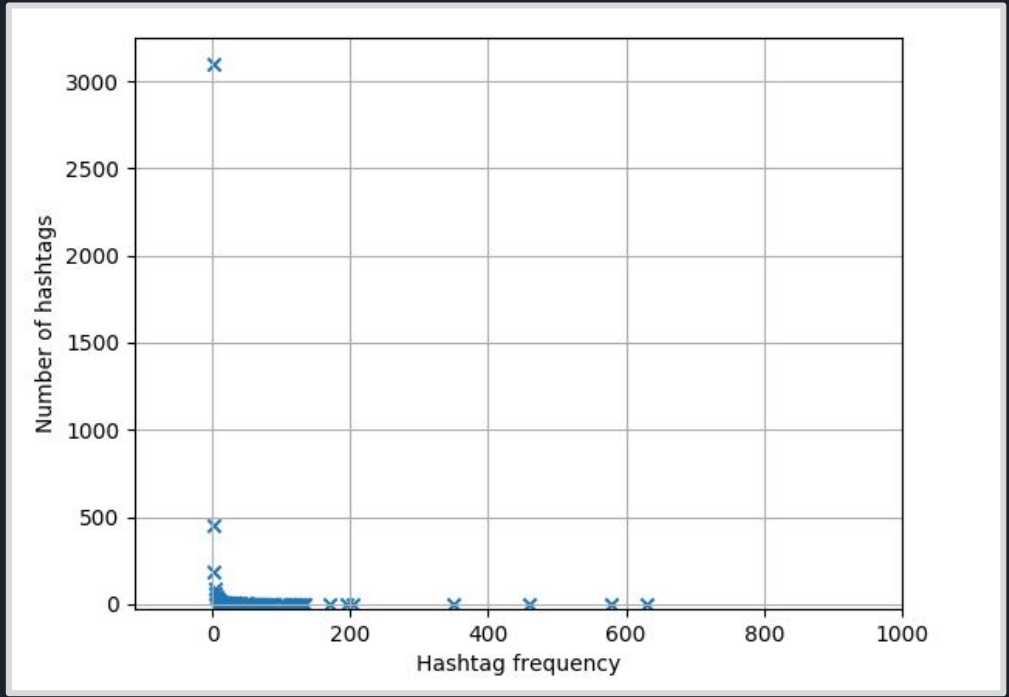
1. DATA COLLECTION

- We used Twitter stream API for collecting 2 million tweets.
  - Tweets were related to news events
- Collected using hashtags such as #news, #cnn, #foxnews etc.

DATA EXTRACTION AND PREPROCESSING

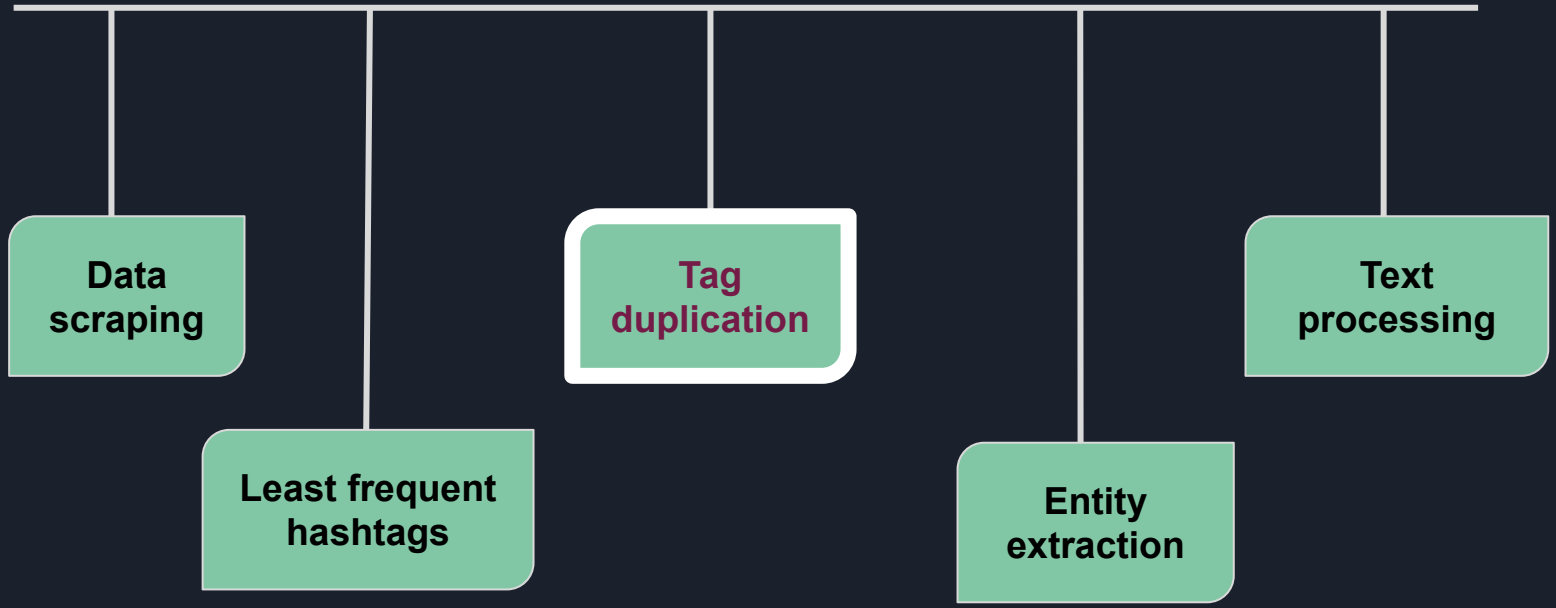


2. DATA DISTRIBUTION - HASHTAG FREQUENCY VS NUMBER OF HASHTAGS



- As seen from the graph, there are many hashtags which occurs just once throughout the dataset.
- So, removing these outlier hashtags can help in creating a much better generic hashtag recommendation system.

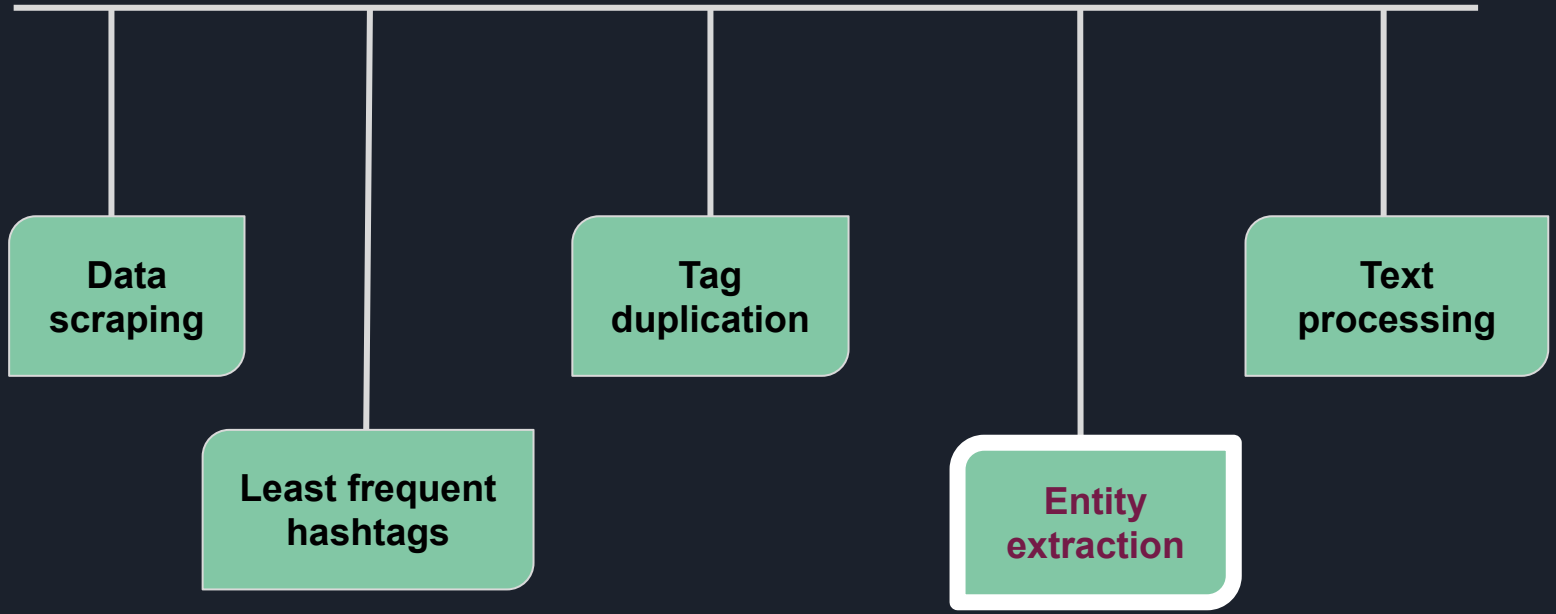
DATA EXTRACTION AND PREPROCESSING



3. DUPLICATE HASHTAGS ELIMINATION

- There were many hashtags which were duplicates but in different form.
- **Example**:
  - #DonaldTrump, #Trump, #PresidentTrump are 3 hashtags about the same person but in different forms.
- So, it's important to have one hashtag in place of duplicates for effective recommendation.
- **Method**:
  - **Fuzzy matching** followed by **word2Vec** was used to identify the duplicate hashtags.

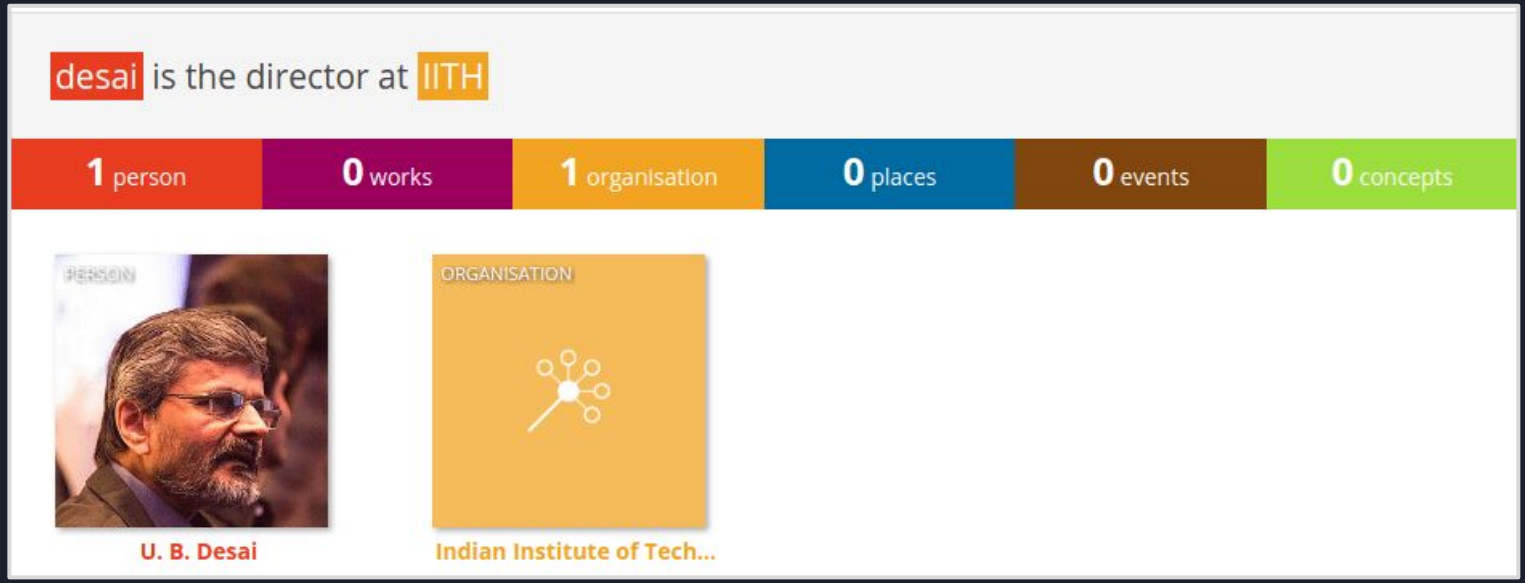
DATA EXTRACTION AND PREPROCESSING



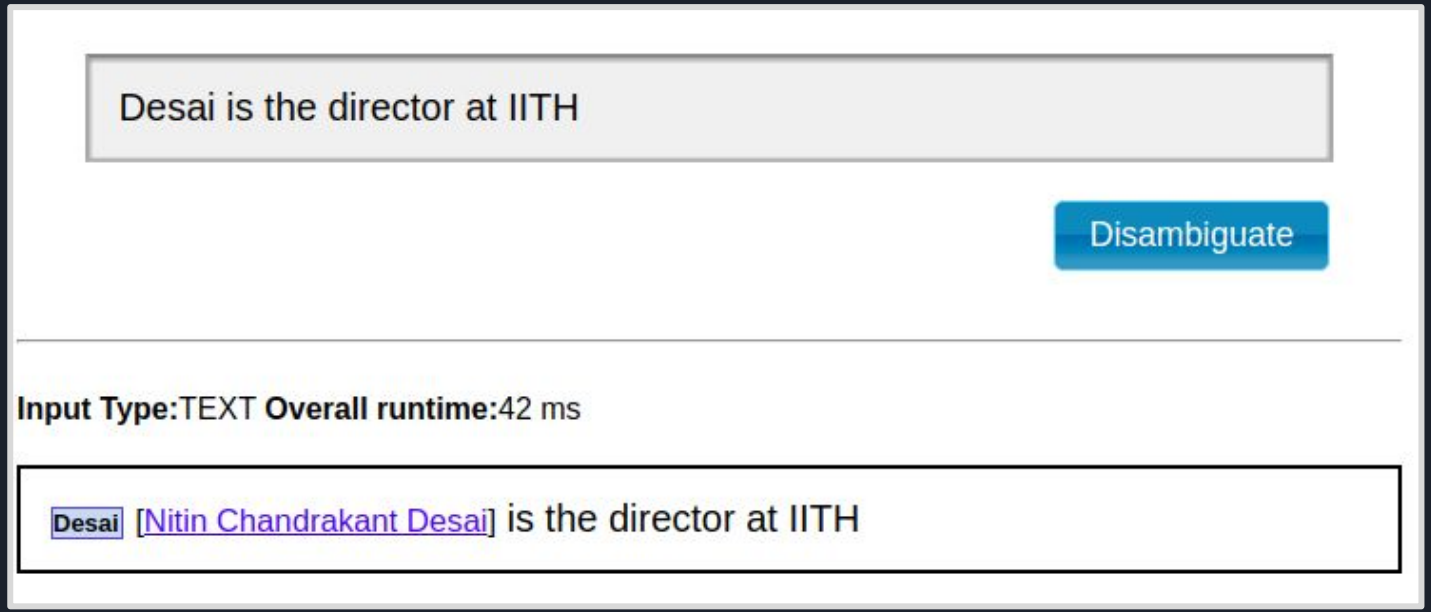
4. ENTITY EXTRACTION

- The entities in every tweet were extracted using some of the modules:
  - **Stanford NER**
    - A pretty standard entity recognition tool .
  - **AIDA / Dandelion API**
    - These take the sentence context into account to find entities.
  - **Watson NLP API**
    - The most accurate entity predictor among all others listed.
  - **Google Cloud Natural Understanding API**

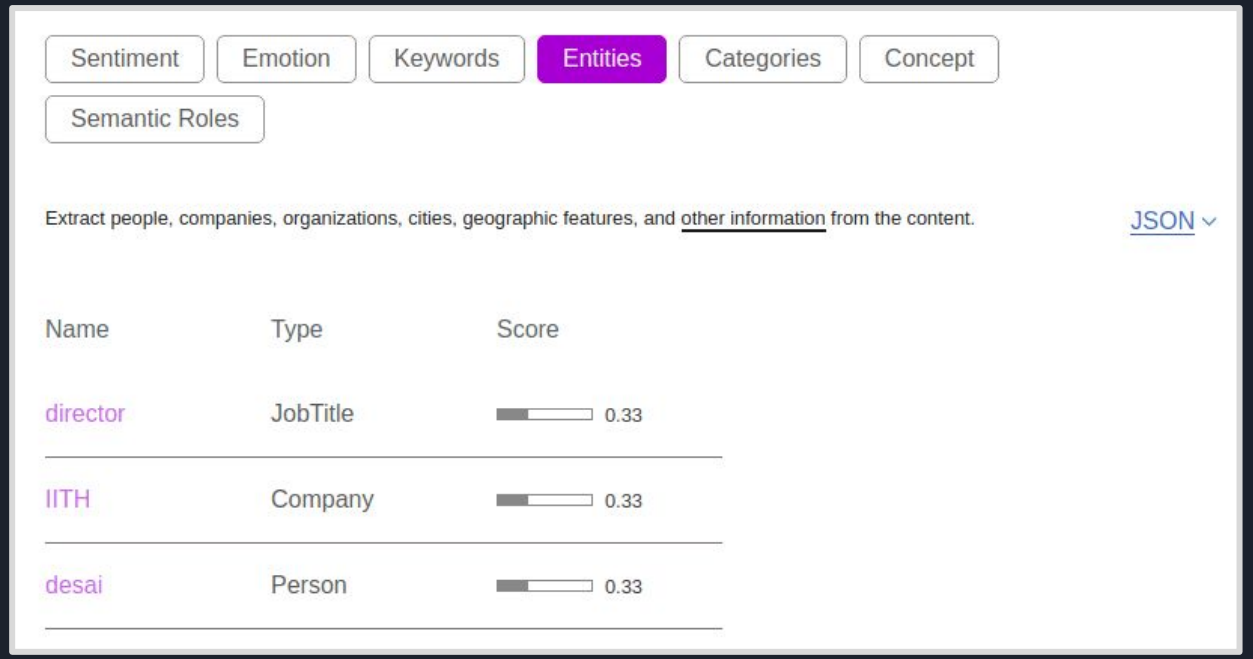
ENTITY EXTRACTION API RESULTS - DANDELION API



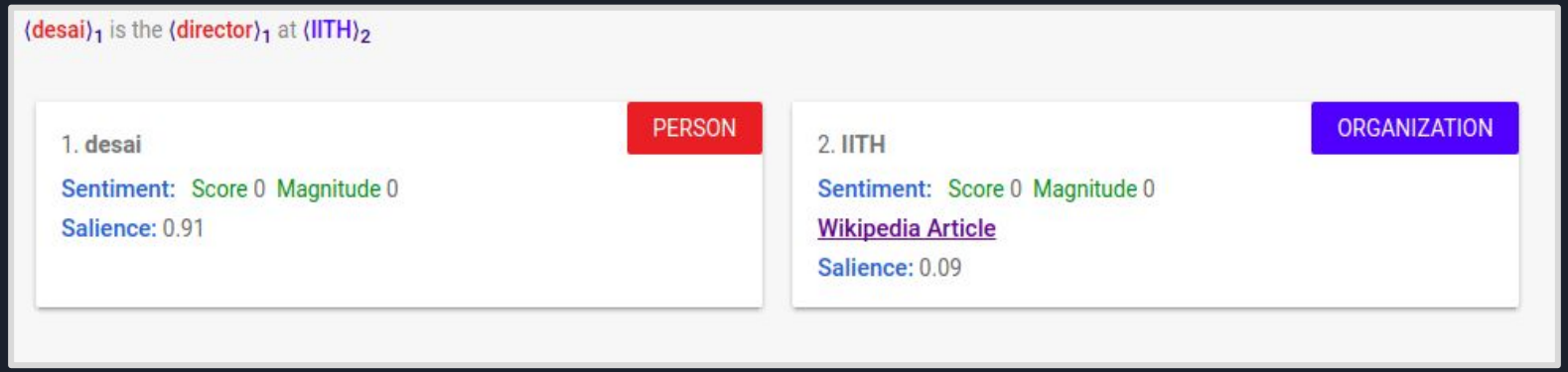
ENTITY EXTRACTION API RESULTS - AIDA



ENTITY EXTRACTION API RESULTS - IBM WATSON



ENTITY EXTRACTION API RESULTS - GCLOUD NLP



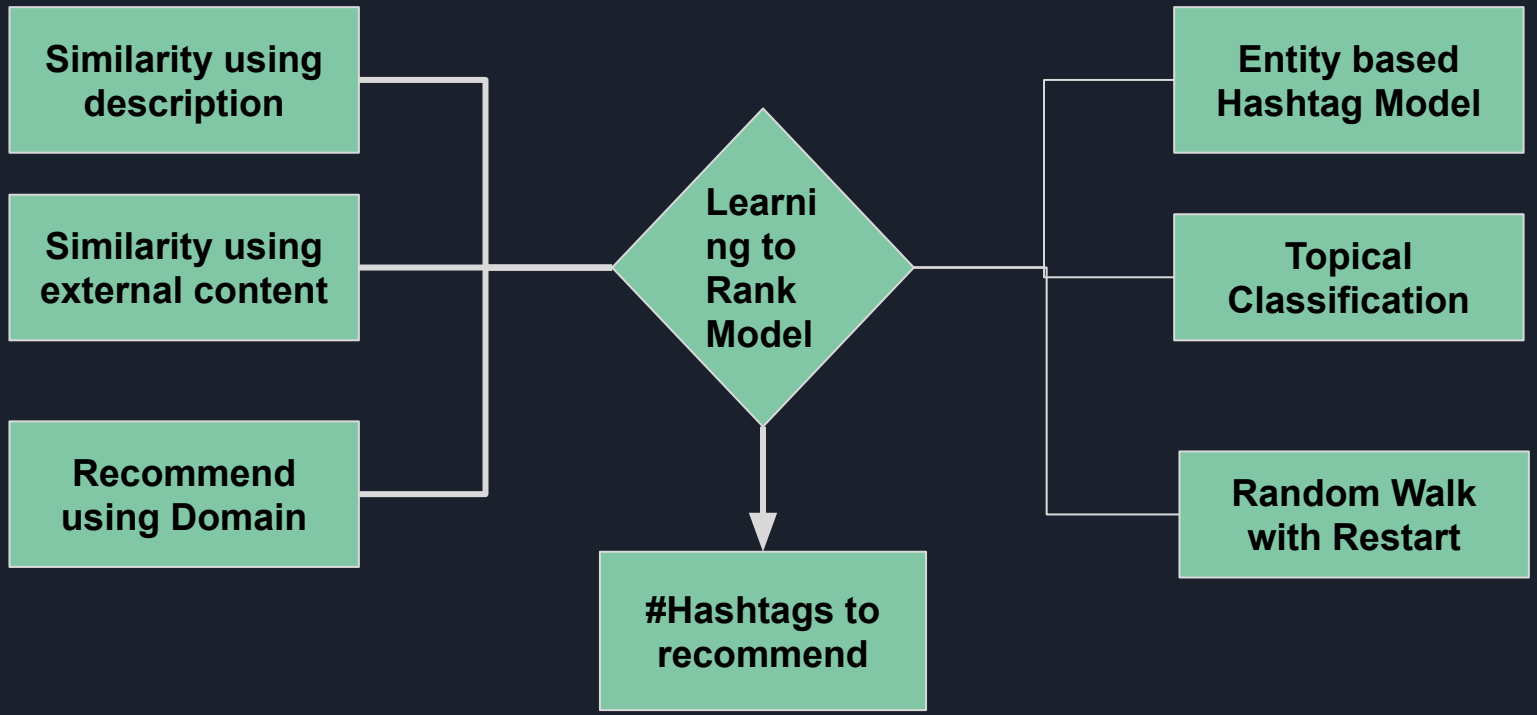
ENTITY EXTRACTION CONT..

- Since the Dandelion API seems to take into account the context of the sentence in addition to the sentence structure to find the entities, the Dandelion API along with Watson API was used to find the entities for all the tweets in the dataset.
- These extracted entities are then used by algorithms like Random Walk and Entity based Hashtag Model to recommend the hashtags for a given tweet.

PRESENTATION OUTLINE

- Introduction
- Data Extraction And Preprocessing
- **Algorithms For Recommending Hashtags**
- Learning to Rank Framework

ALGORITHMS FLOWCHART



RECOMMENDATION BASED ON SIMILAR DESCRIPTION

- It is more likely that similar tweets are annotated by similar hashtags
- We used Cosine similarity and TF IDF weighting scheme to find tweets with similar semantic meaning.
- Based on similarity score , we select top K tweets and recommend hashtags of those top K tweets to the new tweet.

RECOMMENDATION BASED ON SIMILAR EXTERNAL CONTENT

- Tweets often have hyperlinks which generally refers to related articles in form of webpages or docs.
- These articles can be used as additional contextual information for the tweet.
- We extend the previous method to the article content using Doc2Vec algorithm.
- We find the tweets with similar article contents and recommend hashtags of top K similar tweets obtained.

RECOMMENDATION BASED ON THE DOMAIN OF THE ARTICLE

- For all the tweets belonging to a particular domain, we calculate the frequency of all hashtags present
- We store the top K hashtags with high frequency
- When a new tweet is given which belongs to that domain, we recommend the top hashtags of that domain
- So, basically this method is query independent and only depends on domain of the tweet

Recommendation through Topical Classification

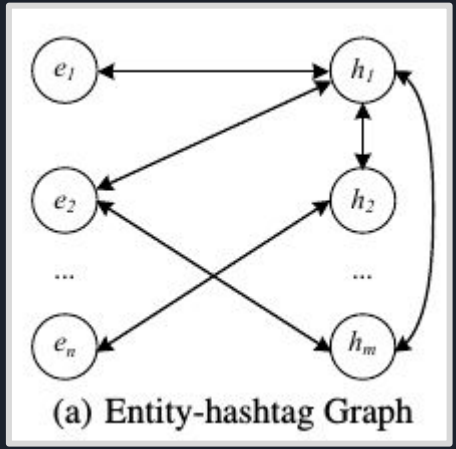
- Hashtags in tweets can be regarded as approximate indicators of tweet's context or topic.
- Following this idea, we used LDA (Latent Dirichlet Allocation) for forming a cluster of topics from the hashtags given.
- For a new tweet, we find the top k most relevant topics using LDA for the recommendation purposes.

RANDOM WALK WITH RESTART (RWR)

- The RWR algorithm works by constructing an entity-hashtag graph and traversing that graph randomly (random walk) with some restart probability.
- A node's score is incremented every time it's visited. This means that the node is reachable from many other nodes (which means higher significance).
- After the random walk, the nodes with top-K scores are the recommended hashtags for the given tweet.

RANDOM WALK WITH RESTART CONT...

- e_i is the set of entities and h_i is the set of hashtags in the graph.
- All the entities of the given article is added to the graph and a hashtag from the training set is added to the graph if the hashtag occurred together with an entity in the given article.
- 3 types of edges are added to the graph:
  - **e_i --> h_j || h_i --> e_j || h_j --> h_k**
- Weights of the edges:
  - **e_i to h_j** : $P(e_i/h_j)$ - The number of times a hashtag h_j is used to annotate a tweet linking to a document containing an named entity e_i, w.r.t. the frequency of h_j.
  - **h_i to e_j** : $P(h_i/e_j)$ - Similar to above, but w.r.t entity e_i , with respect to the frequency of e_j.
  - **h_j to h_k** : P(hk/hj) - co-occurrence count w.r.t. frequency of h_j.



(a) Entity-hashtag Graph

RANDOM WALK - USING VARIOUS PARAMETERS

Entity based Hashtags

- When given a tweet, we get the entities in the tweet and find hashtags connected to those entities from the train data.
- Let N_e be the set of named entities present in the tweet
- Score of a hashtag h_j is computed by

$$Score(h_j) = \sum_{e_i \in N_e} P(h_j|e_i)$$

where, $P(h_j|e_i)$ is the conditional probability of occurrence of hashtag h_j given entity e_i is present.

Entity Based Hashtags CONT..

- We find the scores for all the hashtags linked to the entities of the tweet.
- We select the top K hashtags with highest score and recommend.

Data Structures for RWR and Entity Hashtag

- Our Random Walk and Entity hashtag method leverage from these two Data Structures.
- The first is a Term to Hashtag-frequency-map (THFM)
- The second one is Hashtag to Frequency Map(HFM)

THFM AND HFM

- Term to Hashtag Frequency
  - Primary Keys are the Terms or Entities, observed in the tweets .
  - The value associated with each primary key is a map from hashtag to a frequency count .
  - It indicates how often that hashtag has occurred with the term specified .
- Hashtag Frequency Map
  - Analogous to THFM , by using hashtags as primary key and term frequencies as the final value .

THFM AND HFM

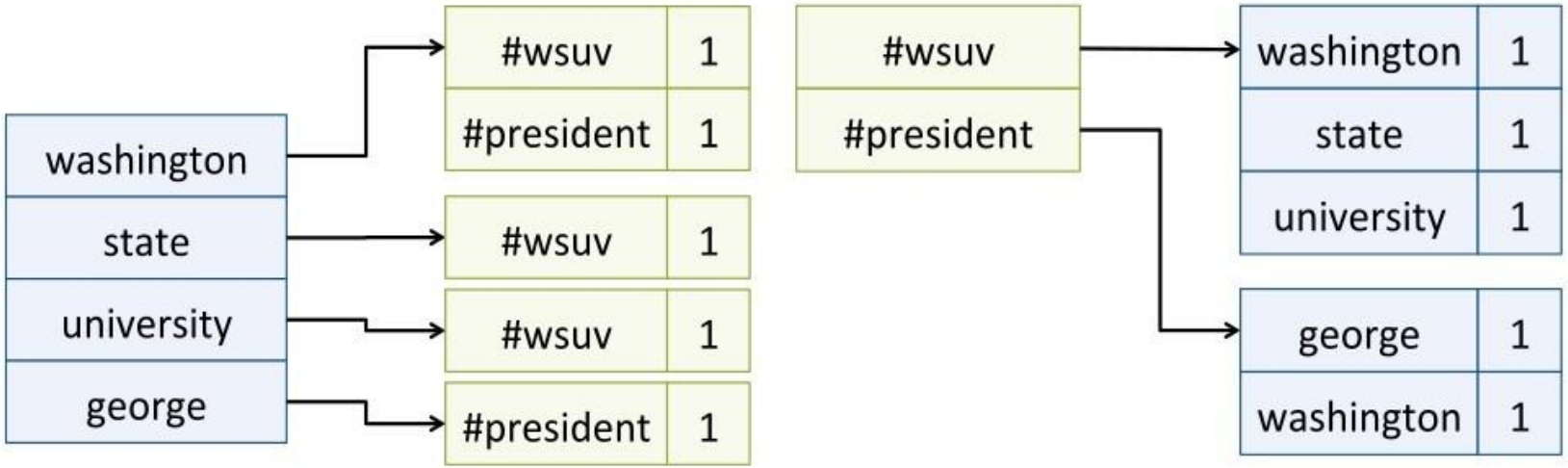


Fig. 1 THFM (*left*) and HFM (*right*)

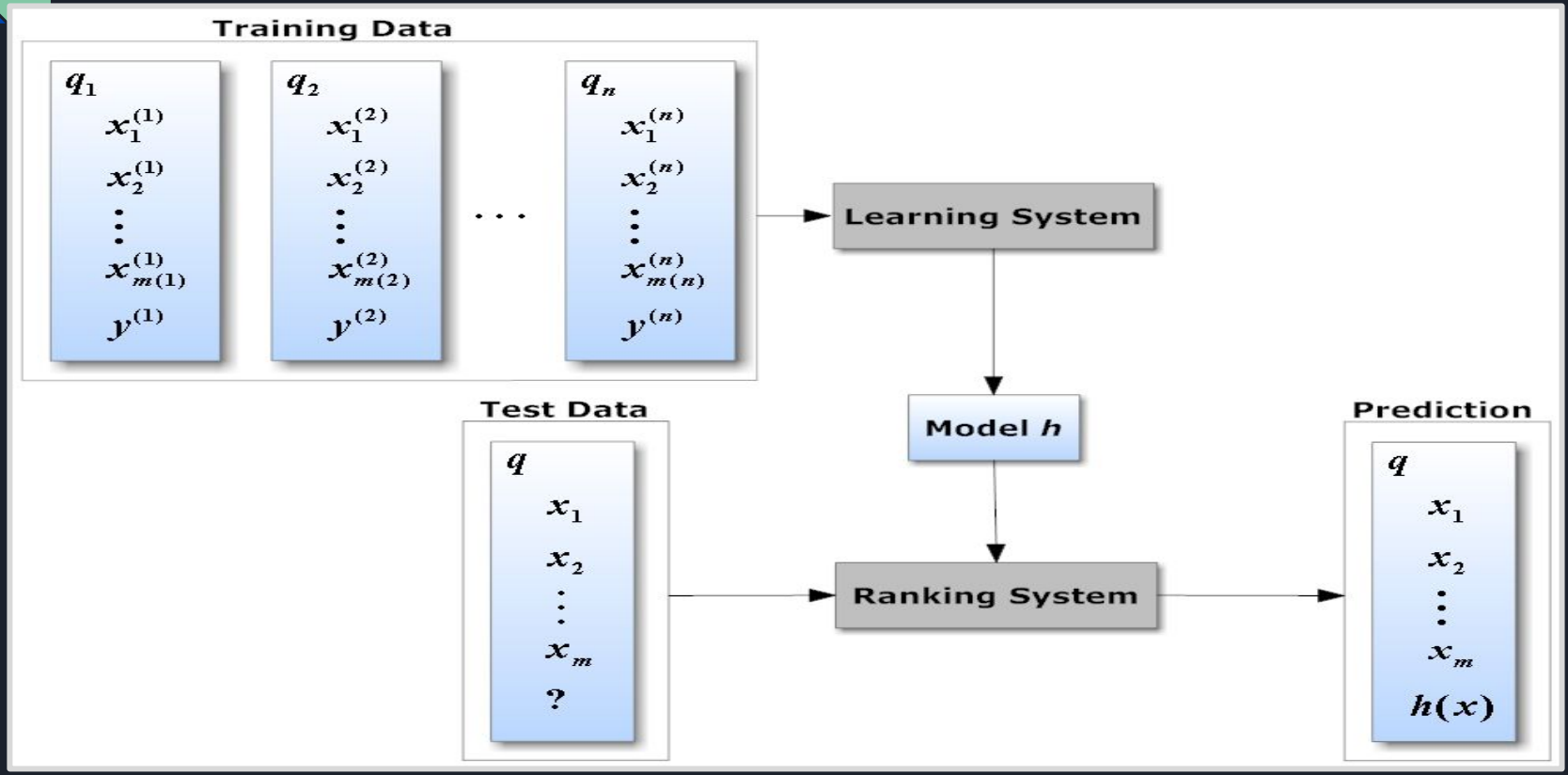
PRESENTATION OUTLINE

- Introduction
- Data Extraction And Preprocessing
- Algorithms for Recommending Hashtags
- **Learning to Rank Framework**

RECOMMENDATION BY LEARNING TO RANK

- Main purpose is to learn a function to automatically learn to rank results effectively .
- Pairwise Learning to Rank - Representing each hashtag as a feature vector and use Pairwise LR for ranking hashtags .
- Each of the candidate hashtag selection discussed above,can be used as a standalone hashtag recommendation method .
- However, to get benefit from all methods, we have aggregated all and ranked them , to get candidate hashtags .

LEARNING TO RANK



Feature and Label Generations

- For a candidate hashtag, a six dimensional feature vector is generated for ranking as follows :
  - All the feature elements are binary (0 or 1).
  - Each feature element corresponds to a candidate hashtag generation method used.
  - The feature element is set to 1 if candidate hashtag is predicted by that method, otherwise 0.
- Label is also required for each candidate hashtag for ranking purposes .
- For each candidate hashtag for a tweet, a 2-dim. label is generated as
  - First element is set to 1 if it is positive hashtag (explained in next slide) else -1 .
  - Second element is the unique tweet id , later used to avoid inter-tweet comparisons while learning the ranking function.

Recommendation by Learning to Rank

- We are using Pairwise Approach to Learning to Rank for our purposes .
- Methodology used for Training
  - Given a tweet t, any candidate hashtag is considered positive(h^+_i) or negative(h^-_i). h^+_i is used to annotate tweet in reality while h^-_i isn't.
  - Instead of taking a single hashtag, a pair of hashtags is taken as a training instance for pairwise learning .
  - Pairs are made with same tweet-id and only when label are different i.e only two pairs are possible which are <positive hash,negative hash> and vice-versa .
  - For training instance <h1,h2> , the feature vector and label are formed from the difference of feature vector and labels of h1 and h2 respectively(in short (h1 - h2)) .
  - In short , the positive pair < positive hash,negative hash> will have +ve label while negative pair <negative hash,positive hash> will have -ve label .
- The goal is to manifest these pairwise preferences to train a ranking model that learns to identify positive pairs accurately and simultaneously neglects negative pairs

Recommendation by Learning to Rank

- Testing phase ,
  - For a test tweet , we first club all the candidate hashtags from individual methods .
  - Feature generation process is done for each of the candidate hashtag .
  - The feature matrix is fed to the ranking model .
  - While testing, each candidate hashtag is paired with the rest of the candidate hashtags and then ordered by using the learned model.
  - The ranking model finally gives an ordering on the candidate hashtags.
  - The top-K candidate hashtags in order are recommended for the test tweet
- We used RankSVM for learning the Ranking model for above task.

THANK YOU