2015

CSE 535: Information Retrieval

PROJECT C REPORT

Team: Fantastic Four

Naved Merchant	50168637
Khushbu Mittal	50169099
Prithvi Shah	50169672
Bhavika Jain	50170168

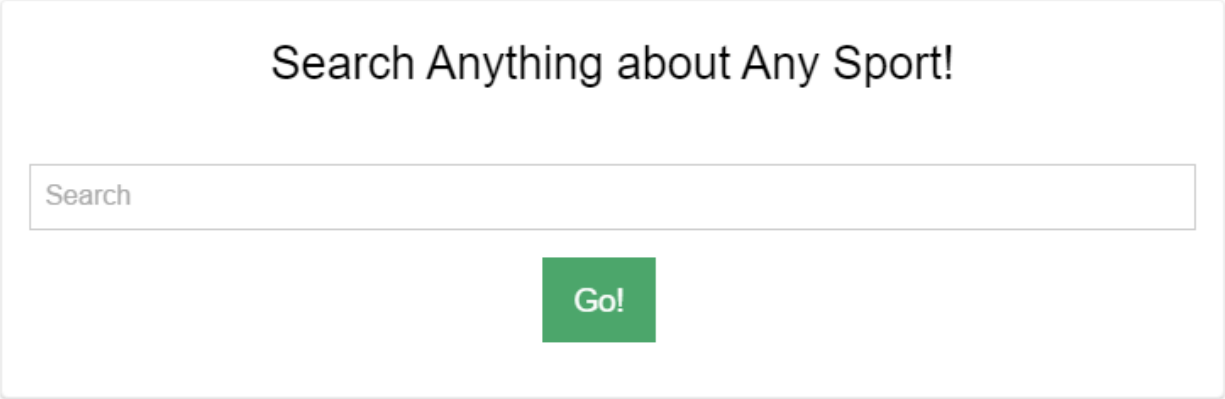
Introduction

In this project, we present a multilingual faceted search system: **Sportify.** As the name suggests, Sportify is a search system for social media data on sports. The user can query the database with a simple and easy-to-use interface and interact with the fetched results in an intuitive manner.

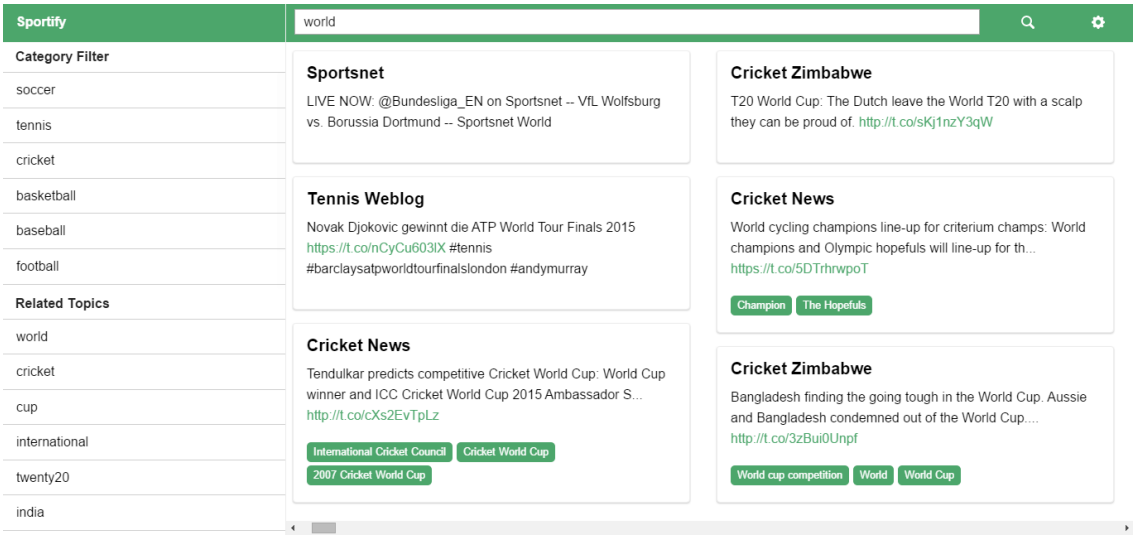
The system scours a database of over 10000 tweets on various sports ranging from Baseball, Basketball, Cricket, Formula 1, NFL, Soccer and Tennis in multiple languages namely English, French, German, Hindi and Spanish. Sportify includes several intelligent features for extracting and presenting results to the user beyond merely the simple query.

User Interface

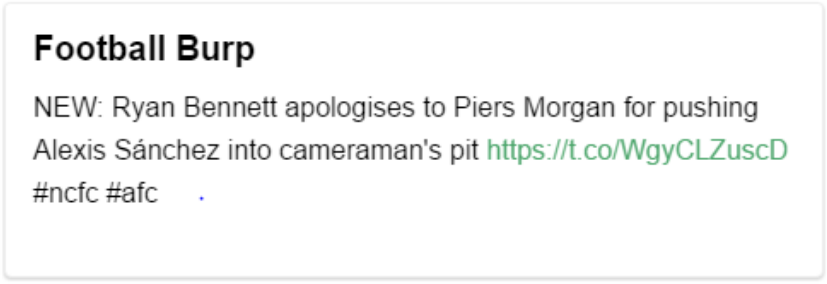
Sportify has a very clean and simple UI. The homepage directly presents a search bar so that the user can start off querying instantly.

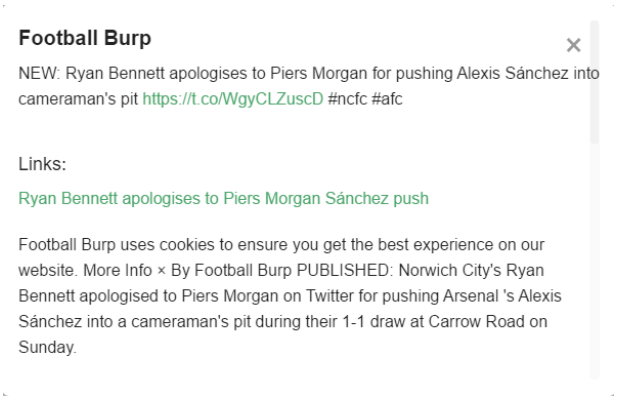


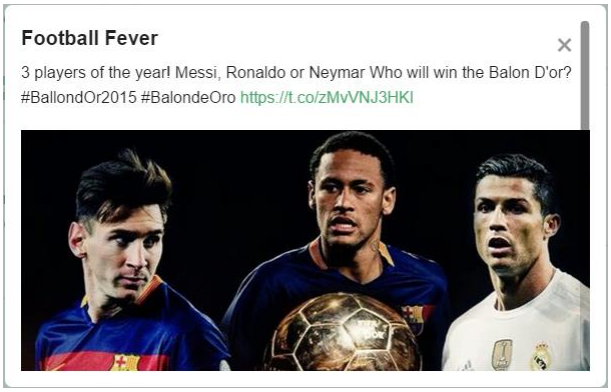
The result page is partitioned into a search bar and an analytics option on the top, a filter column on the left, and a scrollable section on the remainder of the page. The tweets are presented in a card style format which are scrollable sideways.



The tweet card has the User Name as the header followed by the tweet. Each card can be clicked on to expand and view the additional content such as images or URL link summaries present in the tweet.







The column on the left has two portions. The top portion has a category filter to search within specific sports and the bottom portion consists of topics related to the query.

Framework

Foundation-for-Apps framework is a powerful framework that can be used to make single page web applications. It has various things built in features like cards, and a beautiful animation framework called MotionUI that makes animation of components of the web framework extremely easy.

The following are some features of the Foundation framework:

1. UI elements.
   Foundation provides some UI elements that were used in out project. For example: The cards for showing the tweets, and the blocklist used on the sidebar. Both of these are UI elements that come with foundation for Apps framework, and can be customized using the Sass mixins provided to us.

2. State Based Flow
   Foundation provides an easy way for making states and providing navigation between them. For example in our project we used 2 states, a home state for the welcome screen and the other called the search state that takes a parameter to take the item for the search.

3. Animation
   Foundation also provides animations for the UI elements. We can apply an animation for the state transitions, so that when a state enters or exits, it shows a nice animation. Animation can also be applied to other elements such as cards or lists to give a smoother effect to the UI.

4. Block Lists
   Block Lists are a component that can be provided to show various types of lists. It can be shown with check boxes and various other components.

5. Modals
   Modals are dialog boxes that can be used to show dialog boxes on top of the current UI. We have used dialog boxes for showing a loading screen as well as showing more information about the tweet.

   Interaction with the Backend

   The backend is a Solr instance running on an AWS server, which is accessible via http. The interaction has some problems like cross origin resource sharing (CORS), where the app could not access the database since it was hosted in a different place than the current server. This was resolved by allowing CORS by configuring it in solrconfig.xml.

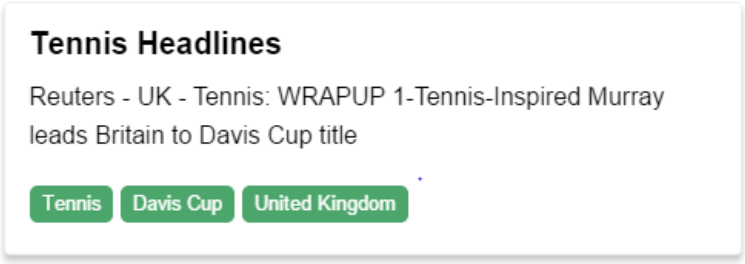
   The interaction was as simple as an AJAX request, and it is made even simpler using Angular JS, since it is used by our application.

   The $http service provided by angular lets us perform HTTP GET and POST requests, that can send the search request to the Solr instance. We had to set it a GET type request and also added wt=JSON so that we could process the JSON natively in javascript. The $http.get() method returns a promise that can be used to get the response of the query.

Content Tagging

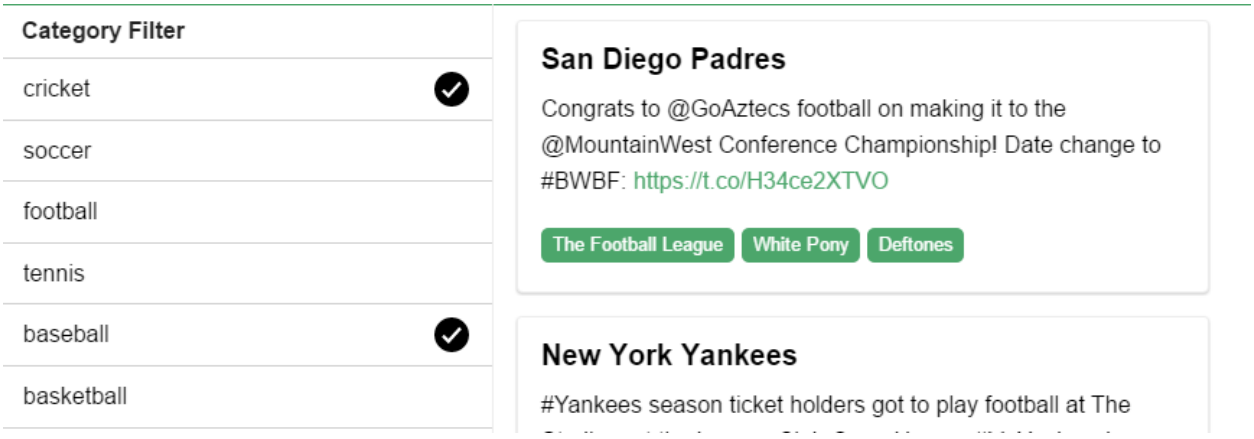
In order to highlight key topics or important entities within a result card, we use the Alchemy concept tagging API. The service utilizes sophisticated NLP techniques to analyze the data and tag information in a manner similar to human-based tagging. AlchemyAPI's concept tagging API is capable of abstraction, understanding how concepts relate and tagging accordingly (e.g. "My favorite brands are BMW, Ferrari, and Porsche." = "Automotive Industry").

The tags appear in green labels at the bottom of each result card (subject to them having any relevant content to be tagged). Each of these tags are clickable and fire a query to fetch all the results pertaining to that particular tag. In addition to this, we have provided a special syntax to explicitly search for a content tag directly from the search bar using: “!tag”, where ‘tag’ is the particular content tag to be searched.

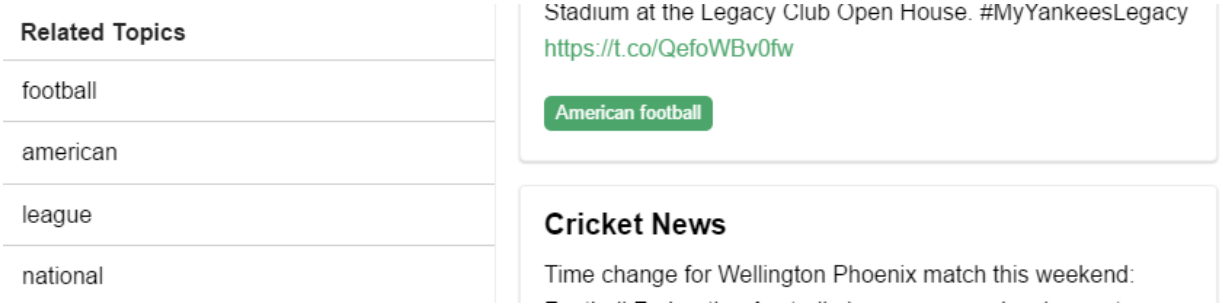


Faceted Search

Category Filters: The indexed data has been categorized on the basis of sport type. After performing the search operation, our system fetches the category from the retrieved results and dynamically displays the categories on the left side of the UI. The user is provided with the feature of further filtering the result set by selecting one or more categories from the panel. By de-selecting the categories, previously fetched results can be restored.

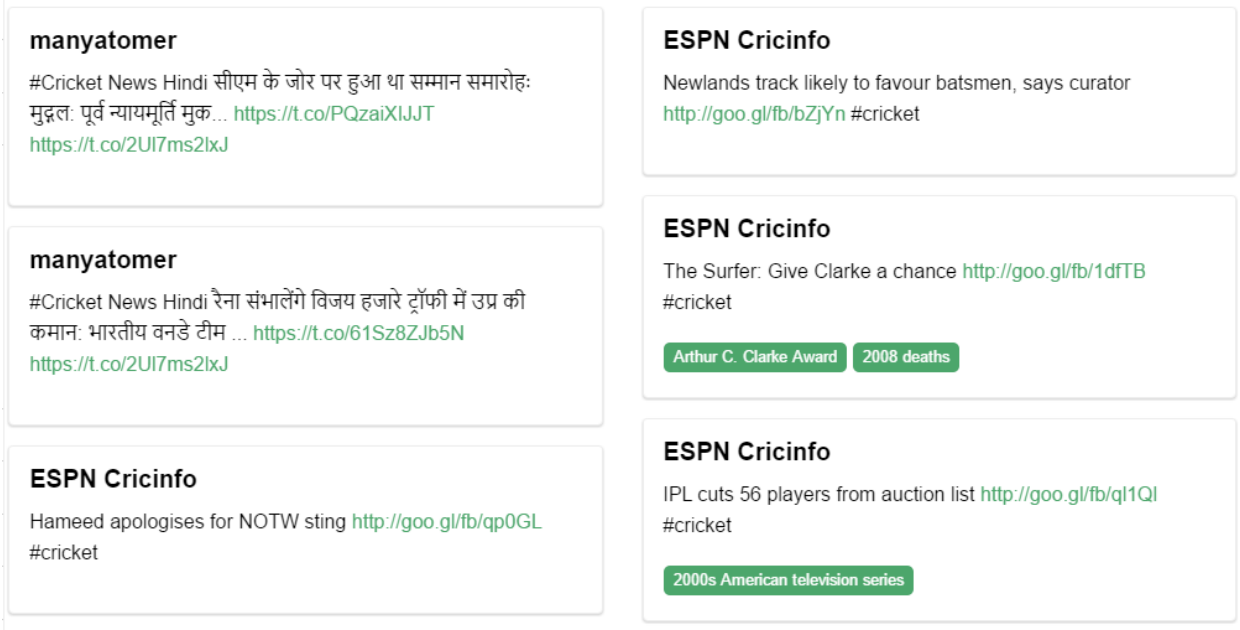


Related Topics: The lower left panel in the user interface corresponds to the topics that can be of great significance to the users. These topics are high frequency topics appearing in the result set that may be closely related to queried term. When a user clicks on any of those terms, the search box gets populated with the corresponding term and a new search operation is performed.



Ranking & Cross Lingual IR

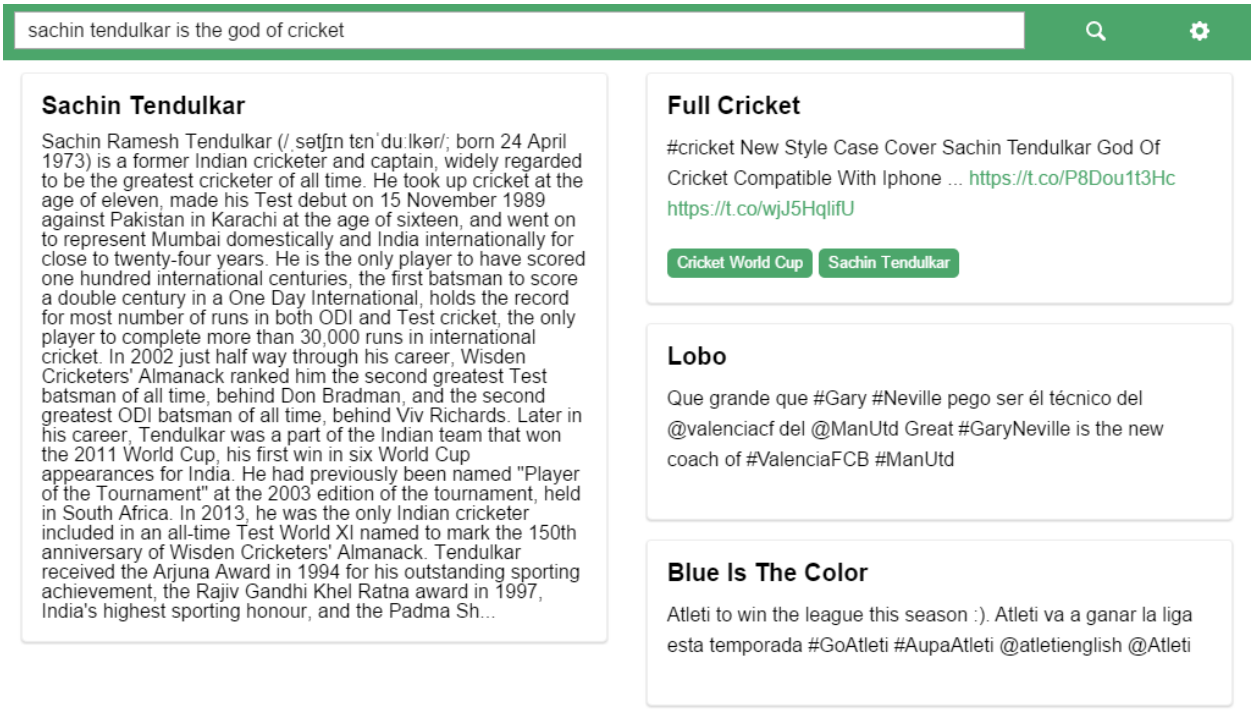
For processing Search Queries, the primary Request Handler provided with Solr is "SearchHandler". We have created a custom handler: “trinity”, for which the default values for query parameters have been specified in solrconfig.xml. We use the phrase slop and minimal match parameters to score and rank the documents and display them in order of relevance to the query. The search model plus the inclusion of content tags and hashtags in the queried fields enables us to retrieve results across multiple languages for a particular query.



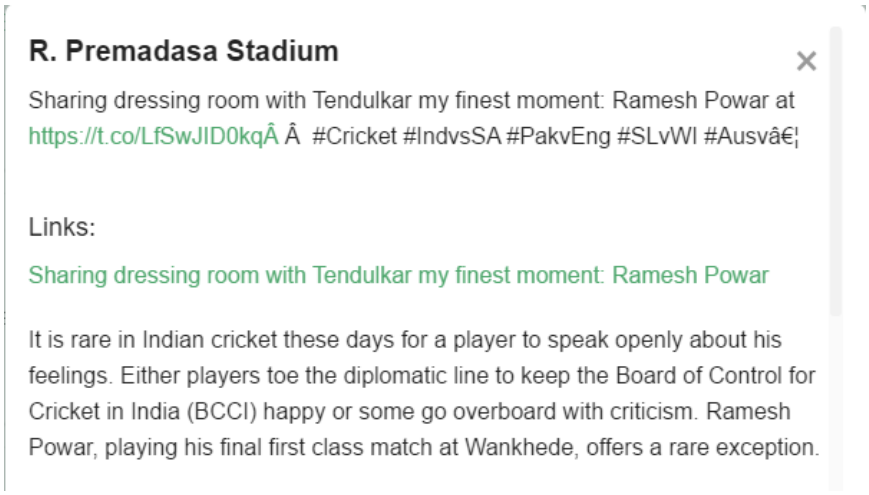
Summarization

Query Summarization:

We have implemented summarization in our system for the given query using excerpts from Wikipedia articles. To identify a topic for generating a summary for a query, we used a Natural Language Processing library -**nlp_compromise** that analyses the tweet and provides a named entity. Now, using the MediaWiki API, we generate and display a summary for the derived entity at the beginning of the search results.



Summarizing Information from Embedded Links:
Sportify allows users to view important information corresponding to the links embedded in any tweet. Embed.ly's Extract API has been used to summarize the contents of the embedded url. Now, when a user selects any result card, he can view the summarized url content along with the tweet-text.



Graphical Analysis

Analysis of the retrieved results has been performed on the basis of languages. The count of retrieved tweets in different languages helps in analyzing the demographics. This can tell us about the popularity of any sport or a team in certain part of the world. For example, high number of Hindi tweets on Cricket can signify the popularity of Cricket in India. On clicking the button at the top-right corner of the UI, a panel displaying the charts pops-up.