Personalized Voice Search for Internet TV

Joaquin A. Delgado, PhD.
Verizon Corporation
San Jose, CA
USA
joaquin.a.delgado@verizon.com

Ravi Kalluri
Verizon Corporation
San Jose, CA
USA
ravi.kalluri@verizon.com

Krishnaja Gutta
Verizon Corporation
San Jose, CA
USA
krishnaja.gutta@verizon.com

Arun B. Krishna
Verizon Corporation
San Jose, CA
USA
arun.krishna@verizon.com

Devon Turner
Verizon Corporation
San Jose, CA
USA
devon.g.turner@verizon.com

ABSTRACT

In this paper, we discuss various strategies that have helped address the unique set of challenges we have faced in the attempt to provide highly relevant and personalized voice search results to users of our Internet TV (a.k.a. IPTV) system. While movie recommender systems have been heavily studied in the academia [1] as well as in the industry [2], full TV recommender systems are less prevalent and require a deeper understanding of real-world complex scenarios, such as using voice search as a mechanism for providing an easy-to-use interface for content search and discovery in IPTV platforms. It also requires the generation of fresh, domain-specific, relevant and highly contextual search results and recommendations within the constraints of what is playable and what is not; whether the suggested programs come from airings currently available from live/linear channels, time-shifted (a.k.a. catch-up) TV, digital video recordings (DVR) or video-on-demand (VOD), or from future airings that may not yet be available but may still be of interest to users to subsequently follow and/or record them.

KEYWORDS

Internet TV, Recommender Systems, Voice Search, Constraint-based recommendation, Query-driven recommendation

1 INTRODUCTION

Internet TV or IPTV systems are platforms that deliver high quality and reliable video streaming of live/linear channels, time-shifted and recorded TV, as well as streaming of video-on-demand (VOD) over Internet protocol (IP). Examples of these platforms include over-the-top (OTT) providers such as Sony's *PlayStation Vue*, *Sling TV*, *Hulu TV*, and the recently announced *YouTube TV*, as well as advanced video IP network providers such as Verizon's *FiOS IPTV* and *Google Fiber*.

This is in contrast, to pure IP-based video-on-demand (VOD) only streaming services (e.g. *Netflix*, *Amazon Prime Video*, *iTunes*, *Google Play*, etc.) and other TV systems that use quadrature amplitude modulation (QAM) for video delivery, a standard used by most traditional digital cable television providers such as Cox, Cablevision, Time Warner Cable and Comcast[1].

Verizon FiOS is a bundled Internet access, telephone, and television service that operates over a fiber-optic communications network with over 6 million customers in nine U.S. States. FiOS is in the process of upgrading its customers to the new FiOS IPTV platform.

Perhaps, the key benefit of having all information come through IP is that it will allow providers to deliver more content to a wider variety of devices, all with an improved user experience (through better analytics and more relevant content), usually accompanied with an improved customer interface and hardware. Navigating channels and programs now feels more like surfing the web, and system upgrades are easily performed, ensuring the experience can be regularly updated with new features as easily as apps are updated on mobile devices.

Even with the advances of video IP, one aspect of the user interface that is seemingly difficult to overcome is the cumbersome typing that is often necessary to perform a search, typically done by selecting letters on a screen using the remote or other pointing devices. Instead, companies have developed advanced voice and natural language understanding user interfaces such as Apple's *Siri*, Amazon's *Alexa* or Google's *Assistant*. These kinds of interfaces are tremendously useful when performing search and discovery on TV due its simplicity vs. the use of an on-screen keyboard. Along the same lines, Verizon has developed and deployed a system for voice command and control as well as for search and discovery for its *FiOS* TV platform, currently available only in its Mobile App.

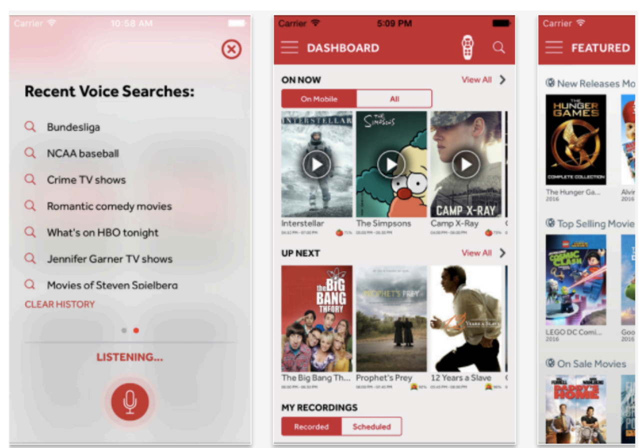

Figure 1: Recent Voice Searches on FiOS TV Mobile App

[1] Technically, Comcast's Xfinity X1 is a hybrid QAM+IP based system.
[2] One could argue that surfacing "free" content is equally important.

Users are always interested in receiving contextually relevant and personalized search results, which may include recommendations based on usage. These results can help improve users' satisfaction and can increase the likelihood that a user finds something enjoyable to watch.

The remainder of this paper describes in more detail the types of queries and the strategies we have developed to cope with the unique challenges regarding relevancy and personalization our voice search users now expect.

2 VOICE QUERIES AND USER INTENT

2.1 Constraint-based Query Fabrication

The simplified diagram in Fig. 2 illustrates the query fabrication sequence of a typical voice search platform, starting with the original speech utterance by the user, automatic speech recognition (*ASR*) module, natural language understanding (*NLU*) processing and formulation of a final search query that ultimately runs against the metadata DB to produce results.

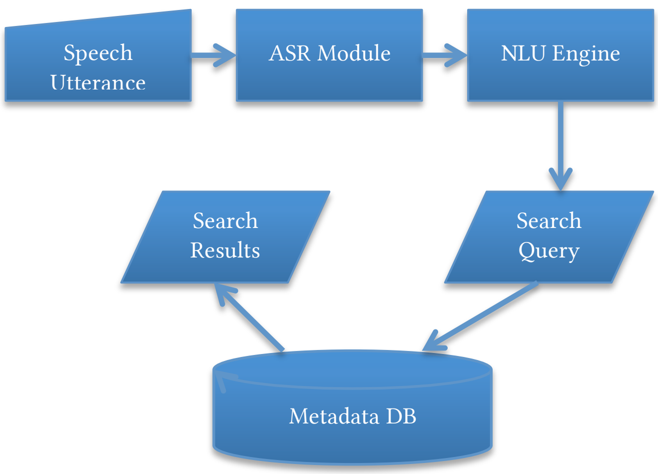

Figure 2: A Typical Voice Search System Architecture

The query generated by the NLU engine is, in general, a constraint-based search query that can be represented using a SQL-like syntax. For example, an utterance converted into text that says: "Show me Brad Pit movies from the 90's" will result in a constraint-based query of the following form:

```
SELECT items FROM MetadataDB
WHERE Person = "Brad Pitt"
AND Decade = "90"
AND ProgramType="Movie"
```

Figure 3: Constraint-based Query

So far this model assumes that constraints in the search query only determine membership in the result set. There is no reference to sorting parameters and/or relevance ranking, which we will discuss in greater details later.

2.2 Program Types

The user, through voice, might be trying to issue a TV control query, like "Tune into channel X" or "lower volume". But when it comes to content discovery the user intent goes hand-in-hand with the type of programs that the user is interested in retrieving. Examples of such program types are:

- Episodic TV Series (e.g. "The Americans")
- Single Programming Event (e.g. "The Grammys")
- Movie (e.g. "Rogue One")
- Music Video (e.g. Maroon 5, "Sugar" – 2015)
- Sports/Game (e.g. "NBA Finals")
  - Event (Actual game/match, e.g. "Golden State Warriors vs. Cavaliers @ Oracle Arena")
  - Non-event (Commentaries, pre-game shows, etc., e.g. "NFL Pre-game Show")

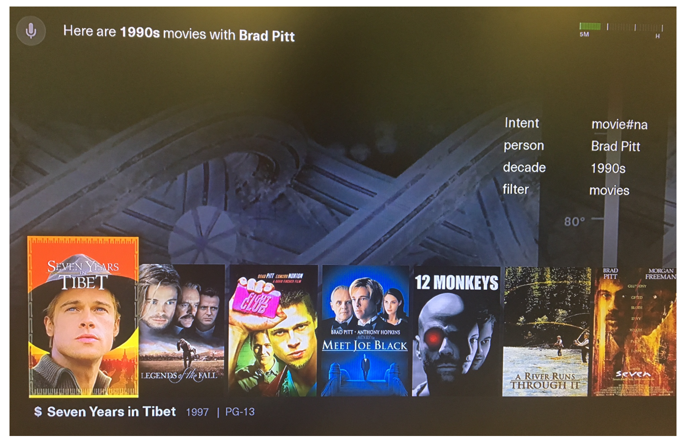

Figure 4: Movies with Brad Pitt from the 90's

2.3 Constraints and Query Specificity

In Fig. 4, the user did not specify a particular movie title he/she wants to watch but rather inquired about all the movies in which a particular person (e.g. "Brad Pitt") was a cast of from the decade of the 1990's. Prior to sorting this query yields many results without any particular order. Depending on the entities used to build the constraints, the number of results and the sorting order, a query could be classified as *specific* or *generic*, with a full spectrum in-between.

Some low to medium cardinality entities, when specified as constraints, lead to more generic queries. For example: *People* (Cast & Crew, Singer), *Genre*, *Sports League*, etc.

Others entities, of very high cardinality when specified as constraints tend to narrow down to a lesser number of results, thus generating a more specific kind of queries. Such entities are: *Title* (movie, music video or TV program title), *Sports Team* or *Sports Tournament*.

On the other hand qualifiers, such as time and quality, augment the specificity of the query by narrowing down the number or results and/or predetermining a sorting order. Examples of such qualifiers are:

- Specific period (year or decade): e.g. "from the 90's".

- Relative time: e.g. "Latest", "Oldest".
- Qualitative sorting: e.g. "Top Rated", "Best".

3 SEARCH STRATEGIES & PERSONALIZATION

3.1 Search Dimensions

Our metadata content contains several attributes that represent various dimensions with which our search applications must work in order to build query-based constraints like the one shown in Fig. 3 When searching TV programs, these attributes include:

- Prose text (including overviews, synopsis, and user reviews).
- Shorter text (such as director and actor names, and titles).
- Text labels (such as moods, keywords, sports league, sports team).
- Numerical attributes (user ratings, movie revenue, the number of awards, Rotten Tomato scores [4], IMDb ratings [5]).
- Programming schedule (airing-date), Release dates and other attributes important in search.

In theory, any of these dimensions can be used to construct hard constraints (filters) or soft constraints (ranking) as part of the search query and sorting strategy. Some of these dimensions could be used to derive newly computed values that can also be used in the ranking function, such as:

- Popularity and Trending: Shows that are popular or trending based on viewership and recording events.
- New, Live and *On Now*: Shows that are airing for the first time, air "live" and/or are currently airing *now*.

3.2 User and Item Taste Vectors

Besides these content dimensions, we have modeled users and items in the same *latent space* using *taste* vectors. Item taste vectors item taste vectors are the result of Matrix Factorization [3] on the user-item DVR recording matrix from the FiOS TV legacy system that *decomposes users and movies into a set of latent factors* (which we can think of as categories like "Fantasy" or "Violence").

For Users, who are relatively new in the system, we are inferring taste vectors from the top items present in the user's current viewing history.

For any item in the result set for which we have an item taste vector we are able to compute a score as the dot product of the user taste vector and the item taste vector:

$$Score(U, I) = \vec{\boldsymbol{U}} \cdot \vec{\boldsymbol{I}} \qquad (1)$$

This score can then be used to personalize any search results set by including it in the function score to determine the final ranking.

We also store and use personalized lists of entities inferred from the user's viewing history, such as:

- Most Watched Channels (*MWC*).
- Most Watched Teams (*MWT*).

to improve our intent-based relevance functions.

3.3 Strategies Selection and Relevance Functions

Based on Program Type that was inferred from the user intent mentioned in Section 2.1 we can decide to apply different strategies to further refine the query output from the NLU Engine.

A *strategy* is actually a query + a relevance function which may be used to sort the search results.

In general, any ranking will happen after all constraints specified in the query are applied. What we refer to, as *recommendation* (*personalization*) *ranking* is actually a function of:

1. Text query (TF×IDF) [6] relevance score, Popularity/trending score plus *Score* (*U, I*) shown in equation 1.
2. In the case of TV Series we weigh higher shows aired on channels in *MWC*, and for sports, we weigh higher items associated to the users' favorite teams/sports in the *MWT* list.

Here is a list of the strategies we implemented.

3.3.1 TV Series Strategy

This includes episodic content airing at various times or available as VOD.

- In general, airing time is not relevant (e.g. we should not give preference to a specific airing window), however it is important to surface first playable[2] assets.
- If the user searches for a series with an exact match, or matches, only return those specific results (e.g. "homeland" should return one result – the TV series "Homeland").
- If the user's title search matches multiple titles, sort titles based on text query relevance; e.g.: "Family" should yield "Modern Family" which is a currently airing show before "Family Ties" and "All in the Family."
- If the user performs a more generic search (e.g. "Dramas on HBO"), apply the filter and then rank by recommendation, including possible bias towards shows from channels/providers from the *MWC* list.
- If the user mentions certain qualifiers, the defined data point should be used:
    - "New episodes": Return series with "new" episode aired in the last week sorted by personalization.
    - "Latest TV airings": Return airings sorted by original airing date and then by personalization.
    - "Top rated series: Return series sorted by IMDB/Rotten Tomatoes rating. If multiple series have the same rating, then sort by personalization.

3.3.2 Single Title Strategy

This includes movies, single programming event and music titles.

[2] One could argue that surfacing "free" content is equally important.

- If the user searches for a title with an exact match, or matches, only return those specific results (e.g. "James Bond movies" or "Star Wars".
- If the user's title search matches multiple titles, sort titles based on text query relevance.
- If the user performs a more generic search based on genre: "comedy movies" or "action thrillers" then rank by recommendations after applying all constraints.
- Generic search with a single filter (e.g. "movies with Brad Pitt"), rank by recommendations after applying all required constraints.
- Generic search with multiple filters (e.g. "movies with Brad Pitt & Angelina Jolie"), same as the case with one filter.
- If no taste vector exists for the user (new profile, no activity) a generic search result should be sorted by original *airing-date* or *release-date*, ascending.
- If a user performs a generic search with the following sort qualifier as interpret by the NLU engine, the defined data point should be used to sort:
    - "Latest comedy movies": Sort based on theatrical release date.
    - "Top rated comedy movies": Sort based on critics rating and then by recommendation..

3.3.3 Game (Sports) Strategy

This applies to both sports events (e.g. "Golden State Warriors vs. Cleveland Cavaliers" NBA match) as well as sports non-events (e.g. "NBA Pre-game Show", "Inside the NBA", etc.). It is perhaps the trickiest strategy to implement.

We classify sports searches into the following 5 categories that range from more specific to the more generic:

- Team Search: A user does a search for a specific team (e.g. "Golden State Warriors").
- Tournament Search: A user does a search for a specific tournament or event (e.g. "Kentucky Derby", "The Masters", "Indianapolis 500", "Super Bowl").
- League Search: A user does a search for a league (e.g. "NBA").
- Sport Genre Search: If the user does a search for a generic genre of sports (e.g. "Basketball").
- Sports On Now: The user does a search for "Sports on now" to figure out what is being shown currently in on TV based on the guide/schedule.

For all these use cases we factor the following elements into the ranking:

- Program Type: Sports Events will always weigh higher than Sports Non-Events.
- *New* or *Live* programs will always weigh higher than repeated programs or "re-runs".
- Airing start-time: While events that are airing now are weighed the highest, past and upcoming events decrease their score based on Gaussian decay [7] based on airing start-time.
- League Bias: We bias results towards popular/ professional leagues in the U.S. (e.g. NFL, NBA, NHL, NCAA, etc.) when a generic search are performed.
- Personal Team Bias: Extra boost is given to teams in the user's *MWT* list in generic searches as well. We found that for the sports use case users are more interested in live sports results, and upcoming schedule of their favorite teams rather than popular sports shows.

4 CONCLUSIONS

In Voice Search for Internet TV queries with high specificity tend to be very precise and have little need for additional sorting and/or relevance ranking to satisfy the user's request. On the other hand, the more generic a query is, the more results the user has to sift through, thus requiring some type of relevance ranking to bubble up results that are expected to be most relevant to the user.

But, what is relevance in this context? Is relevancy universal or does it depend on the user that asking?

Relevancy tuning is a hard problem — it's usually misunderstood, and it's often not immediately obvious when something is wrong. It usually requires seeing many bad examples to identify problematic patterns, and it's often challenging to know what better results would look like without actually seeing them show up. Unfortunately, it's often not until well after a search system is deployed into production that organizations begin to realize the gap between out-of-the-box relevancy defaults and true domain-driven, personalized matching.

This paper describes some promising strategies that have been used to implement personalized voice search for Internet TV to mitigate the relevancy problem. We will report the evaluation of this approach in subsequent reports.

DISCLAIMER

This paper makes does not describe any specific product feature nor does it promise the delivery of one. It bares no influence on the development roadmap of FiOS IPTV or any other Verizon product for that matter. It *is* a research paper, exploratory in nature, that represents the discussions and ideas solely attributed to the authors and does *not* represent any company plan and/or position.

REFERENCES

[1] Jonathan L. Herlocker , Joseph A. Konstan , Loren G. Terveen, John T. Riedl, Evaluating collaborative filtering recommender systems, *ACM Transactions on Information Systems (TOIS)*, v.22 n.1, p.5-53, January 2004
[2] The Netflix Prize [http://www.netflixprize.com/]
[3] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.
[4] Rotten Tomatoes [https://www.rottentomatoes.com/]
[5] Internet Movie Database (IMDb) [http://www.imdb.com/]
[6] Ricardo Baeza-Yates, Berthier Ribeiro-Neto. Modern Information Retrieval: The Concepts and Technology Behind Search, Addison Wesley, 2011
[7] The Closer, The Better [https://www.elastic.co/guide/en/elasticsearch/guide/current/decay functions.html]