G+1 44 更多 下一个博客» leeshuqing@gmail.com 信息中心 退出

NETFLIX

The Netflix Tech Blog

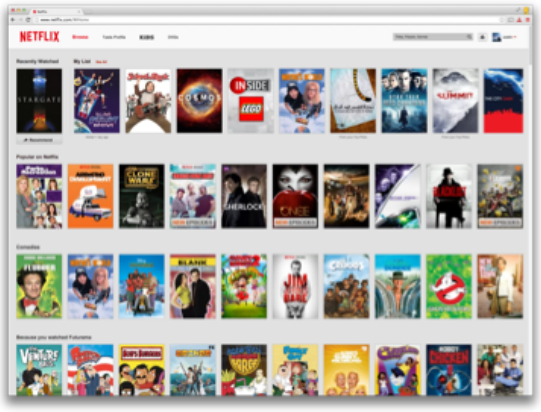
Thursday, April 9, 2015

Learning a Personalized Homepage

by Chris Alvino and Justin Basilico

As we've described in our previous blog posts, at Netflix we use personalization extensively and treat every situation as an opportunity to present the right content to each of our over 57 million members. The main way a member interacts with our recommendations is via the homepage, which they see when they log into Netflix on any supported device. The primary function of the homepage is to help each member easily find something to watch that they will enjoy. A problem we face is that our catalog contains many more videos than can be displayed on a single page and each member comes with their own unique set of interests. Thus, a general algorithmic challenge becomes how to best tailor each member's homepage to make it relevant, cover their interests and intents, and still allow for exploration of our catalog.

This type of problem is not unique to Netflix, it is faced by others such as news sites, search engines, and online stores. Any site that needs to choose items from a large number of available possibilities and then present them in a coherent and easy-to-navigate manner will face the same general challenges. Of course, the problem of optimizing Netflix homepages has its own unique aspects, ranging from interface constraints to differences with how movies and TV are consumed compared to other media.



An example of a personalized Netflix homepage on our website.

Currently, the Netflix homepage on most devices is structured with videos (movies and TV shows) organized into thematically coherent rows presented in a two-dimensional layout. Members can scroll either horizontally on a row to see more videos in that row or vertically to see other rows. Thus, a key part of our personalization approach is how we choose rows to display on the homepage. This involves figuring out how to select the rows most relevant to each member, how to populate those rows with videos, and how to arrange them on the limited page area such that selecting a video to watch is intuitive. In the rest of this post, we will highlight what we think are the most relevant and interesting aspects of this problem and how we can go about solving some of them.

Before going on, it is worth mentioning that at Netflix we have a multitude of algorithms for doing personalization and recommendation including: how to predict the rating that a member will give a video, how to rank videos in each row, and how to create meaningful groupings of videos. Thus, in some sense, personalized page generation represents the next logical step in the evolution of our recommendation system that started with rating prediction and subsequently evolved into personalized ranking of our entire catalog. It involves solving a more general problem: how best to populate a personalized two-dimensional page of content, including recommendations.

Links

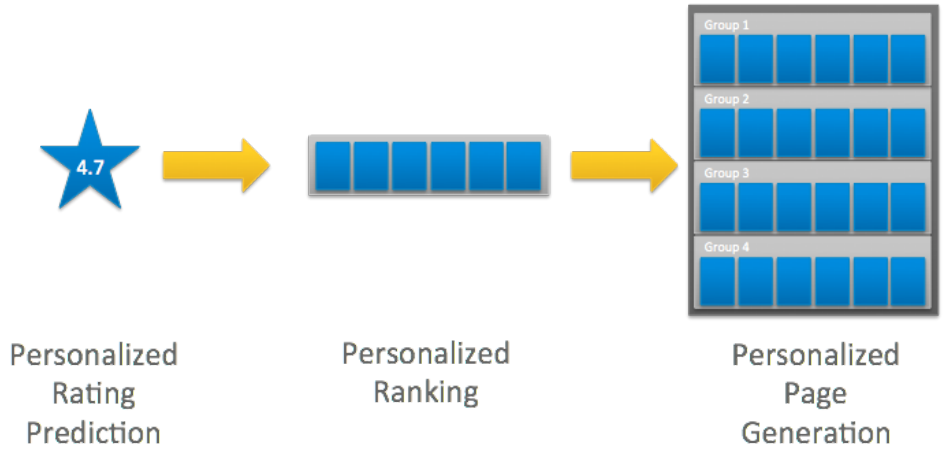
- Netflix US & Canada Blog
- Netflix America Latina Blog
- Netflix Brasil Blog
- Netflix Benelux Blog
- Netflix DACH Blog
- Netflix France Blog
- Netflix Nordics Blog
- Netflix UK & Ireland Blog
- Netflix ISP Speed Index
- Open positions at Netflix
- Netflix Website
- Facebook Netflix Page
- Netflix UI Engineering
- RSS Feed

About the Netflix Tech Blog

This is a Netflix blog focused on technology and technology issues. We'll share our perspectives, decisions and challenges regarding the software we build and use to create the Netflix service.

Blog Archive

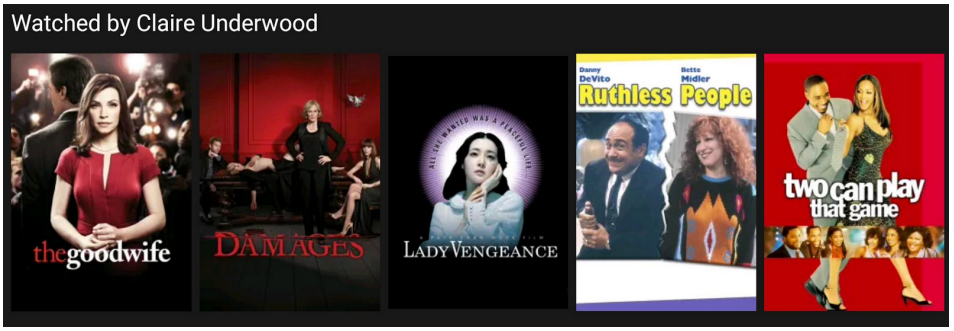
- ► 2016 (25)
- ▼ 2015 (50)
  - ► December (7)
  - ► November (5)
  - ► October (5)
  - ► September (6)
  - ► August (6)
  - ► July (3)
  - ► June (2)
  - ► May (2)
  - ▼ April (3)
    - Learning a Personalized Homepage
    - Introducing Vector: Netflix's On-Host Performance ...
    - Extracting contextual information from video asset...
  - ► March (3)
  - ► February (5)
  - ► January (3)
- ► 2014 (37)
- ► 2013 (52)
- ► 2012 (37)
- ► 2011 (17)



Evolution of our personalization approach.

Why Rows Anyway?

We organize our homepage into a series of rows to make it easy for members to navigate through a large portion of our catalog. By presenting coherent groups of videos in a row, providing a meaningful name for each row, and presenting rows in a useful order, members can quickly decide whether a whole set of videos in a row is likely to contain something that they are interested in watching. This allows members to either dive deeper and look for more videos in the theme or to skip them and look at another row. This would not be the case if, for example, the page contained a large, unorganized collection of relevant videos.



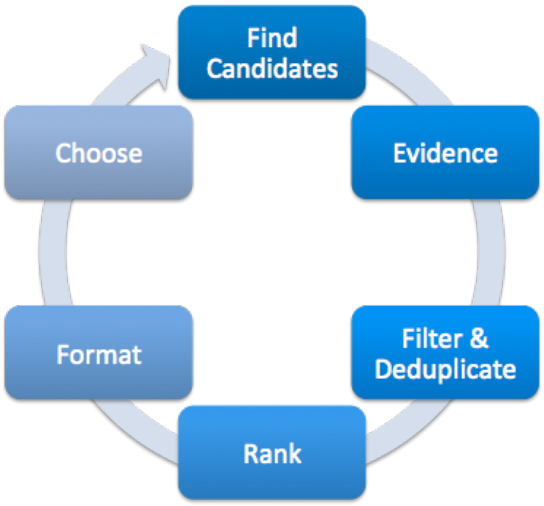
A possible row of titles that might be watched by one of our Netflix original characters.

One natural way to group videos is by genre or sub-genre or other video metadata dimensions like release date. Of course, the relationship between videos in a row does not have to be due to metadata alone, but can also be formed from behavioral information (for example from collaborative filtering algorithms), videos we think a member is likely to watch, or even groups of videos watched by a friend. Thus, each row can offer a unique and personalized slice of the catalog for a member to navigate. Part of the challenge and fun of creating a personalized homepage is figuring out new ways to create useful groupings of videos, which we are constantly experimenting with (e.g., rows of titles that might be watched by one of our Netflix original characters shown above).

► 2010 (8)

Labels

"cloud architecture" (3)
A/B Testing (2)
accelerated compositing (2)
adwords (1)
Aegisthus (1)
algorithms (4)
aminator (2)
analytics (5)
Android (3)
angular (1)
animations (1)
api (16)
appender (1)
AppsAndDevices (3)
Archaius (2)
architectural design (1)
architecture (1)
artwork (1)
Asgard (1)
Astyanax (4)
authentication (1)
automation (2)
autoscaling (3)
availability (4)
AWS (31)
bake (1)
benchmark (2)
big data (11)
billing (1)
Blitz4j (1)
build (4)
Cable (1)
caching (5)
Cassandra (14)
chaos engineering (1)
chaos monkey (5)
chukwa (1)
ci (1)
classloaders (1)
Clojure (1)
cloud (26)
cloud architecture (17)
cloud prize (3)
CO2 (1)
collection (1)
complex event processing (1)



Process for creating and choosing rows.

Once we have a set of possible video groups to consider for a page, we can begin to assemble the homepage from them. To do this, we start by finding candidate groupings that are likely relevant for a member based on the information we know about them. This also involves coming up with the evidence (or explanations) to support the presentation of a row, for example the movies that the member has previously watched in a genre. Next, we filter each group to handle concerns like maturity rating or to remove some previously watched videos. After filtering, we rank the videos in each group according to a row-appropriate ranking algorithm, which produces an ordering of videos such that the most relevant videos for the member in a group are at the front of the row. From this set of row candidates we can then apply a row selection algorithm to assemble the full page. As the page is assembled, we do additional filtering like deduplication to remove repeat videos and format rows to the appropriate size for the device.

Page-level algorithmic challenge



To algorithmically create a good personalized homepage means assembling one page per member profile and device from thousands of videos that may be relevant for a member and from easily tens of thousands of potential rows, each with a variable number of videos. On top of that, we need to balance several factors that often compete for precious screen real estate. Our approach to personalization and recommendation largely focuses on helping our members find something new to watch, which we call discovery. However, we also want to make it easy for a member to watch the next episode of a show or re-watch something that they watched in the past, which normally falls outside the realm of recommendation. We want our recommendations to be accurate in that they are relevant to the tastes of our members, but they also need to be diverse so that we can address the spectrum of a member's interests versus only focusing on one. We want to be able to highlight the depth in the catalog we have in those interests and also the breadth we have across other areas to help our members explore and even find new interests. We want our recommendations to be fresh and responsive to the actions a member takes, such as watching a show, adding to their list, or rating; but we also want some stability so that people are familiar with their homepage and can easily find videos they've been

computer vision (2)
concurrency (1)
configuration (2)
configuration management (2)
conformity monkey (1)
content delivery (1)
content metadata (1)
content platform engineering (2)
content quality (1)
continuous delivery (4)
coordination (2)
cost management (1)
crypto (1)
Cryptography (2)
CSS (2)
CUDA (1)
dart (1)
data (1)
data migration (1)
data pipeline (5)
data science (8)
data visualization (1)
database (5)
DataStax (2)
deadlock (1)
deep learning (1)
Denominator (2)
dependency injection (1)
device (3)
device proliferation (1)
devops (2)
distributed (11)
DNS (1)
Docker (1)
Dockerhub (1)
DSL (1)
Dyn (1)
DynECT (1)
efficiency (2)
Elastic Load Balancer (1)
elasticsearch (3)
ELB (1)
EMR (2)
encoding (4)
energy (1)
eucalyptus (1)
eureka (2)
evcache (2)
Experimentation (2)

recommended in the recent past. Finally, we need to be able to place task-oriented rows, such as “My List,” in amongst the more discovery-oriented rows.

Each device has different hardware capabilities that can limit the number of videos or rows displayed at any one time and how big the whole page can be. As such, the page generation process must be aware of the constraints of the device for which it is creating the page, including the number of rows, the minimum and maximum length of a row, the size of the visible portion of the page, and whether or not certain rows are required or are not applicable for a certain device.

While there are many challenges to page generation, tackling recommendation problems at this level also opens up new solutions. As mentioned before, selecting a diverse set of items is important in a recommendation system. However, it can be challenging to navigate a diverse ranking since the relevant items may be blended with other items that do not match someone’s current intent. However, by presenting a two-dimensional navigation layout, a member can scroll vertically to easily skip over entire groups of content that may not match their current intent and then find a more relevant set, which they can then scroll horizontally to see more recommendations in that set. This allows for coherent, meaningful individual rows to be selected while maintaining the diversity of the videos shown on the whole page, and thus lets the member have both relevance and diversity.

Building a page algorithmically

There are several approaches for how we can build our homepage algorithmically. The most basic is a rule-based approach, which we used for a long time. Here a set of rules define a template that dictates for all members what types of rows can go in certain positions on the page. For example, the rules could specify that the first row would be Continue Watching (if any), then Top Picks (if any), then Popular on Netflix, then 5 personalized genre rows, and so on. The only personalization in this approach was from selecting candidate rows in a personalized way, such as including “Because you watched <video>” rows for videos someone has watched in the past and genre rows based on known genre preferences. To choose specific rows within each type, simple heuristics and sampling were used. We evolved this template using A/B testing to understand where to place rows for all members.

This approach served us well, but it ignored many aspects we consider important for the quality of the page, such as the quality of the videos in the row, the amount of diversity on the page, the affinity of members for specific kinds of rows, and the quality of the evidence we can surface for each video. It also made it hard to add new types of rows, because for a new row to succeed it would need to not only contain a relevant set of videos in a good order but also be placed appropriately in the template. Because of this, the rules for the template grew over time and became too complex to handle the variety of rows and how they should all be placed, which represented a local optimum for the member experience.

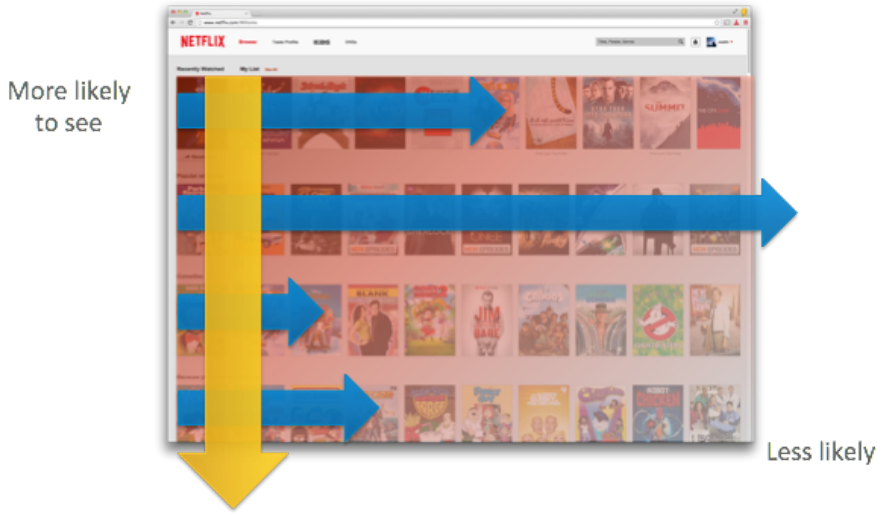
To address these issues, we can instead think of personalizing the ordering of rows on the homepage. The simplest approach for doing this is to treat rows as items in a ranking problem, which we call a row-ranking approach. For this approach, we could leverage a lot of existing recommendation or learning-to-rank approaches by developing a scoring function for rows, applying it to all the candidate rows independently, sorting by that function, and then picking the top ones to fill the page. Even though the space of rows may be relatively big, this type of approach could be relatively fast and may result in reasonable accuracy. However, doing this would lack any notion of diversity, so someone could easily get a page full of slight variations of their interests, such as many rows each with different variants of comedies: late-night, family, romantic, action, etc.

A simple way to add in diversity is to switch from a row-ranking approach to a stage-wise approach using a scoring function that considers both a row as well as its relationship to both the previous rows and the previous videos already chosen for the page. In this case, one can take a simple greedy approach and pick the row that maximizes this function as the next row to use and then re-score all the rows for the next position taking that selection into account. Depending on the diversity function, this greedy selection may not lead to an optimal page. Using a stage-wise approach with k-row lookahead could result in a more optimal page than greedy selection, but it comes with increased computational cost. Other approaches to greedily add diversity based on submodular function maximization can also be used.

failover (2)
falcor (2)
fault-tolerance (12)
flamegraphs (2)
Flow (1)
footprint (1)
FRP (1)
functional reactive (1)
garbage (1)
garbage collection (1)
gc (1)
Genie (4)
git (1)
Governator (1)
GPU (2)
gradle (1)
green (1)
Groovy (1)
Hack Day (3)
Hadoop (12)
HBase (1)
high volume (4)
high volume distributed systems (11)
Hive (2)
HTML5 (8)
https (1)
Hystrix (5)
IBM (1)
ice (1)
images (1)
IMF (3)
IMSC (2)
infrastructure (2)
initialization (1)
innovation (3)
insights (1)
inter process communication (1)
Interoperable Master Format (3)
iOS (1)
Ipv6 (2)
isolation (1)
ISP (1)
java (5)
JavaScript (19)
jclouds (1)
jenkins (2)
kafka (4)
Karyon (2)
keystone (2)

However, even the stage-wise algorithm is not guaranteed to produce an optimal page because a fixed horizon may limit the ability to fill in better rows further down the page. Thus, if we can instead take a page-wise approach by defining a full-page scoring function, we can try to optimize it by choosing rows and videos appropriately to fill the page. Of course, the space of possible pages is huge, even larger than the space of possible rows. Since a page layout is defined in a discrete space, directly optimizing a function that defines the quality of the whole page is a computationally prohibitive integer programming problem.

When solving a page optimization problem with any of these approaches, there are also various constraints that need to be taken into account that were mentioned before, like deduping, filtering, and device-specific constraints. Each of these constraints add to the complexity of the optimization problem.



Notional importance of navigation modeling. Members are more likely to vertically than horizontally, which means videos presented in the upper left are much more likely to be seen than those in the lower right.

When forming the homepage it is also important to consider how members navigate the page, i.e., to consider which positions on the page they are likely to pay attention to and interact with in a session. Placing the most relevant videos in the positions that are most likely to be seen, which tends to be the upper-left corner, should reduce the time for a member to find something relevant to watch. However, modeling navigation on a two-dimensional page is difficult, especially taking into account that different people may navigate differently, people's navigation patterns may change over time, there are differences in navigation across different device types based on the interaction design, and that navigation is clearly dependent on the relevance of the content shown. With an accurate navigation model, we can inform better placement of videos and rows and where on the page to focus on relevance as opposed to diversity.

Machine Learning for page generation

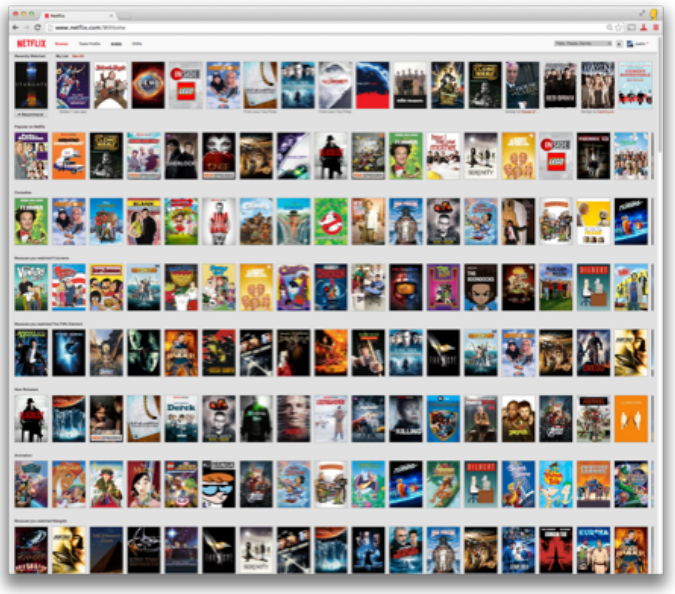
At the core of building a personalized page is a scoring function that can evaluate the quality of a row or a page. While we could use heuristics or intuition for building such a scoring function and tune it using A/B testing, we prefer to learn a good function from the data so that we can easily incorporate new data sources and balance the various different aspects of a homepage. To do this, we can use a machine learning approach to create the scoring function by training it using historical information of which homepages we have created for our members, what they actually see, how they interact, and what they play.

There is a large set of features that we could potentially use to represent a row for our learning algorithms. Since rows contain a set of videos, we can use any features of those videos in the row representation, either by aggregating across the row or indexing them by position. These features can be simple metadata or more useful model-based features that represent how good of a recommendation we believe a specific video is for a member. Of course, we have many different recommendation approaches, so we can include them as different features to learn an

lifecycle (1)
linux (2)
lipstick (2)
load balancing (3)
localization (1)
localization platform engineering (1)
locking (1)
locks (1)
log4j (1)
logging (2)
machine learning (7)
Map-Reduce (1)
media pipeline (1)
meetup (3)
memcache (2)
memcached (1)
Meson (1)
message security layer (1)
Mobile (3)
modules (1)
monitoring (1)
msl (1)
nebula (1)
negative keywords (1)
Netflix (17)
Netflix API (8)
netflix graph (1)
Netflix OSS (14)
NetflixOSS (12)
neural networks (1)
node.js (4)
NoSQL (5)
observability (1)
Open source (10)
operational excellence (1)
operational insight (2)
operational visibility (1)
optimization (2)
OSS (4)
outage (1)
page generation (1)
payments (1)
Paypal (1)
performance (24)
personalization (7)
personalization infrastructure (1)
phone (1)
Pig (4)
pipeline (1)

ensemble of them at the page level. We can also look at the quality of the evidence associated with the row, such as how much support there is for a member being interested in a specific genre. We can also look at past interactions with the row to see if that row or similar such rows have been consumed in the past by the member. We can also add simple descriptive features like how many videos are in a row, in what position a row is being placed on a page, or how often we've shown the row in the past. Diversity can also be additionally incorporated into the scoring model when considering the features of a row compared to the rest of the page by looking at how similar the row is to the rest of the rows or the videos in the row to the videos on the rest of the page.

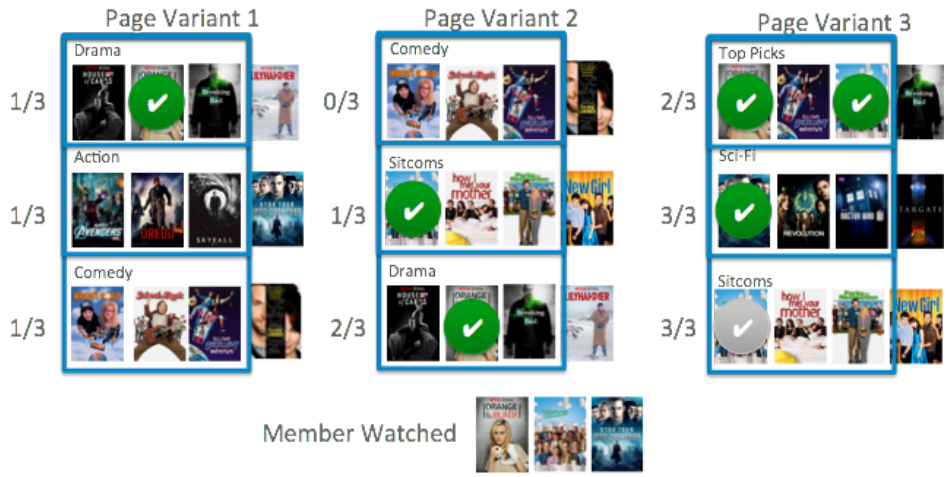
While the space of potentially useful features is quite large, there are several challenges with training machine learning models for scoring rows. One challenge is dealing with presentation bias, where a member can only play from a row on the homepage that we've chosen to display, which can have a huge impact on the training data. To further complicate things, the position of a row on the page can greatly affect whether a member actually sees the row and then chooses to play from it. To handle these presentation and position biases, we need to be extremely careful about how we select training data for our algorithms. There is also a challenge around how attribution is allowed in the model; a video may have been played in a certain row in the past, but does that mean the member would have chosen that same video if it was placed in a different row but in the first position? Perhaps the title of a row being "Critically Acclaimed Documentaries" was responsible for play where it may not have been selected without that additional evidence, for example, in a "New Releases" row, even if it was in a better position. Learning over features to represent diversity can also be challenging because while the space of potential rows at different positions on the page is large, when the rest of the page (or the already chosen rows) is taken into account for diversity, the space of possible pages is even larger.



Page-level metrics

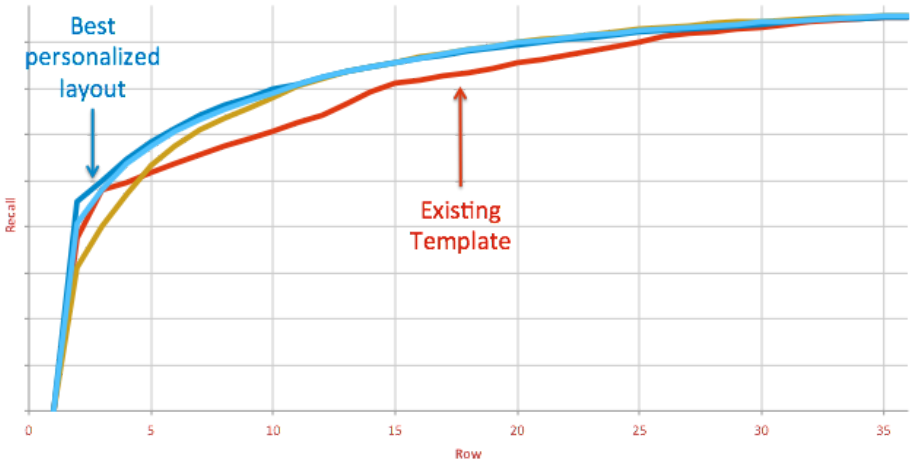
To deal with these challenges, as with any algorithmic approach, choosing a good metric is important. Of fundamental importance in page generation is how to evaluate the quality of the pages produced by a specific algorithm during offline experimentation. While we ultimately will test any potential algorithmic improvement online in an A/B test, we would like to be able to focus our precious A/B testing resources on algorithms that we have evidence are likely to improve the quality of the pages. We also need to be able to tune the parameters of those algorithms before A/B testing. To do this, we can use historical data to generate hypothetical pages from new algorithmic approaches, provided we can choose a good metric for page quality.

pki (1)
Playback (2)
prediction (2)
predictive modeling (3)
Presto (2)
prize (1)
prs (1)
pubsub (1)
pytheas (1)
python (3)
Quality (1)
quality control (1)
rca (2)
React (3)
Reactive Programming (2)
real-time insights (2)
real-time streaming (3)
Recipe (1)
recommendations (9)
Redis (3)
reinvent (2)
reliability (7)
remote procedure calls (1)
renewable (1)
research (2)
resiliency (7)
REST (2)
Ribbon (2)
Riot Games (1)
root-cause analysis (2)
Route53 (1)
rule engine (1)
Rx (2)
Samza (1)
scalability (12)
scale (1)
scripting library (1)
search (4)
security (8)
Servo (1)
shared libraries (1)
simian army (5)
SimpleDB (3)
site reliability (1)
spark (2)
spinnaker (1)
sqoop (1)
ssd (2)
ssl (2)



Example of two-dimensional recall metrics. For each page variant, the fractions on the side represent the recall at 1-by-3, 2-by-3, and 3-by-3 metrics, respectively.

To come up with page-level quality metrics, we took inspiration from ranking metrics that are common in information retrieval (many of which exist in the literature) for a one-dimensional list and created ones that work over a two-dimensional layout. For instance, consider a simple metric like Recall@*n*, which measures the number of relevant items in the top *n* divided by the total number of relevant items. We can extend it in two dimensions to be Recall@*m*-by-*n*, where now we count the number of relevant items in first *m* rows and *n* columns on the page divided by the total number of relevant items. Thus, Recall@3-by-4 may represent quality of videos displayed in the viewport on a device that initially can show 3 rows and 4 videos at a time. One nice property of recall defined this way is that it automatically can handle corner-cases like duplicate videos or short rows. We can also hold one of the values *n* (or *m*) fixed and sweep across the other to calculate, for instance, how the recall increases in the viewport as the member would scroll down the page.



Comparison of four page algorithms in recall up to a fixed column position while sweeping the row position. The red line is the previous rule-based approach and the blue is a personalized layout.

Of course, Recall is a basic metric and requires choosing values for *m* and *n*, but we can likewise extend metrics that assign a score or likelihood for a member seeing a position, like NDCG or MRR, to the two-dimensional case. We can also adapt navigation models like Expected Reciprocal Rank to incorporate two-dimensional navigation through the page and take into account the cascading aspect of browsing. With such page-level metrics defined, we can use them to evaluate changes in any of the algorithmic approaches used to generate the page, not just the algorithms for ordering the rows, but also the selection, filtering, and ranking algorithms, or any of the input data that they use.

Other challenges

STAASH (1)
Stamos (1)
stream processing (4)
streaming (2)
suro (1)
SWF (1)
synchronization (1)
tablet (2)
testability (1)
Timed Text (1)
tls (2)
traffic optimization (1)
TTML (2)
TV (5)
UI (15)
UltraDNS (1)
unit test (2)
uptime (2)
user interface (5)
Velocity (1)
video quality (2)
visualization (1)
WebKit (3)
websockets (1)
Wii U (1)
windows (1)
winner (1)
winners (1)
workflow (1)
workshop (1)
ZeroToDocker (1)
ZooKeeper (1)
zuul (1)

There is no shortage of challenging questions that come up in engineering the homepage. For example: When is it appropriate to take into account other context variables such as the time of the day or device, in how we populate the homepages? How do we find the appropriate trade-off between finding the optimal page and computational cost? How do we form the home pages during the critical first few sessions of a member, precisely at the time when we have the least information about them? We need to think about and weigh the importance of each of these questions every day in order to continually improve the Netflix homepages.

Conclusion

While Netflix may be most famous in the recommendations community for the Netflix prize, we think of personalized page generation as the next step in the evolution of our personalization approach from rating prediction to video ranking to now page generation. We have taken the initial step of coming up with our first algorithm for personalized page generation that showed significantly better online performance than our existing template, and deployed it last year. However, personalized page generation is a challenging problem that involves balancing a multitude of factors, and we think that this is just the beginning. There is a lot of potential to improve the homepages for all of our members and help them easily find content they will love.

We are always looking for talented researchers and engineers to join our team. So if you are interested in helping us solve these types of problems and increasing global happiness, please take a look at some of our open positions on the Netflix jobs page.

3 Comments Sort by **Oldest**

Add a comment...



Sachi Mohanty
This is somewhat like YouTube home page personalization?
Like · Reply · 1 · Apr 9, 2015 9:59pm



Alex Addison
One of the basic things is that in this discovery-oriented page, I might find the same thing I have already watched and rated appearing, even appearing more than once.
Like · Reply · 1 · Apr 22, 2015 2:53pm



Angus Mezick
The biggest problem I have with all of this is that sometimes I have to go hunting for "my list" within the PS3 app. I want that list to be pinned to a known location (first row) or be accessible from some sort of shortcut.
Like · Reply · 2 · Jun 16, 2015 6:42am

Facebook Comments Plugin

Posted by Justin Basilico at 9:24 AM

G+1 +44 Recommend this on Google

Labels: algorithms, machine learning, Netflix, page generation, personalization, recommendations

Newer Post Home Older Post

Terms of Use | Privacy | Cookie Preferences

Awesome Inc. template. Powered by Blogger.