Targeted Content for a Real-Time Activity Feed: For First Time Visitors to Power Users

Diane Hu
Etsy, Inc
Brooklyn, NY, USA
dhu@etsy.com

Tristan Schneiter
Etsy, Inc
Brooklyn, NY, USA
tschneiter@etsy.com

ABSTRACT

The Activity Feed (AF) is Etsy's take on the ubiquitous "web feed" - a continuous stream of aggregated content, personalized for each user. These streams have become the de facto means of serving advertisements in the context of social media. Any visitor to Facebook or Twitter has seen advertisements placed on their web feed. For Etsy,[1] an online marketplace for handmade and vintage goods with over 29 million unique items, the AF makes the marketplace feel a bit smaller for users. It enables discovery of relevant content, including activities from their social graph, recommended shops and items, and new listings from favorite shops. At the same time, Etsy's AF provides a platform for presenting users with targeted content, as well as advertisements, served alongside relevant and timely content.

One of the biggest challenges for building such a feed is providing an engaging experience for all users across Etsy. Some users are first-time visitors who may find Etsy to be overwhelming. Others are long-time power users who already know what they're looking for and how to find it. In this work, we describe solutions to the challenges encountered while delivering targeted content to our tens of million of users. We also cover our means of adapting to each user's actions, evolving our targeted content offerings as the user's familiarity with Etsy grows. Finally, we discuss the impact of our system through extensive experimentation on live traffic, and show how these improvements have led to increased user engagement.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining

Keywords

Recommender Systems; Hadoop; Large-scale Systems

[1] `http://www.etsy.com`

WWW 2015 Companion, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2742002.

1. INTRODUCTION

As any shop owner would say, one-time buyers are great, but repeat buyers are priceless. For an e-commerce site like Etsy - which specializes in one-of-a-kind, handmade goods - getting users to stay and understand the unique economy of the Etsy marketplace is invaluable. For years, Etsy thrived on a strong network of buyers and sellers from all over the world who intimately understood Etsy's function, style, and brand. These users repeatedly returned, not just to make purchases, but also to engage in Etsy's social and inspirational features [7]. They are the foundation on which Etsy's success stands on today, and in order to strengthen and expand our user base, it is important for Etsy to continue to engage more users in this capacity. For this reason, the Activity Feed (Figure 1) was launched several years ago, with the intent of providing a space for users to share and engage with other users and shops that they take interest in.

The Activity Feed is a frequently updated stream of content in the form of *stories* that flow from the top to the bottom of the page as time goes on (see Figure 1). Earlier versions of the feed were composed entirely of activities from the user's social graph. There were several issues with this approach. First, it provided us very little control when user experiences were sub-optimal. For example, users who followed few people had extremely sparse feeds, leading to a less-than-engaging experience. Other users followed too many people, leading to an overload of content flowing by. Still, more pressing, was the lack of a content delivery mechanism: there was no way of providing targeted content to users, including personalized advertisements and recommendations, that could greatly assist in users' discovery of relevant content.

In this paper, we describe how we addressed these issues in a new version of the Activity Feed that was built and deployed to all of our tens of millions of users. These users cover the full gamut of Etsy engagement levels, ranging from the first-time visitor to the long-time power user. As described in Section 4, our solution revolved around moving away from a purely social graph and introducing a new class of targeted content that can include anything from advertisements to personalized recommendations. In the remaining sections, we show how we balanced targeted content with user's pre-existing content on the feed, while providing helpful recommendations depending on the user's familiarity with the site. The result is a diverse and well-paced feed that interests users and encourages them to move toward the next level of user engagement and loyalty on Etsy.

2. ACTIVITY FEEDS

Activity feeds are loosely time-ordered sets of varied content, and are commonly found on popular social networks such as Twitter, Facebook, and LinkedIn [1, 10, 5]. The content in feeds is drawn from a set of *content producers*. Typically, content producers are nodes in a user's *social graph*, which represents the user's interests based on explicit user action, for example, "following" another user. We call content generated from a user's social graph *organic content*. We also note that the time-ordered nature of the AF makes it ideal for keeping up with the actions of your *social graph*, the users with whom one shares commonalities. Be,cause AFs are a singular source for "what is new" amongst a peer group, they often serve as a primary landing page for users.

Because Etsy is such a large and eclectic marketplace, with over 29 million unique items, one difficulty users often have is in item discovery. If a user must actively search for a type of item, they are limited by their own knowledge of what exists on Etsy. In Etsy's previous AF, the content pulled from the user's social graph included stories on listings, shops, and users that were favorited and followed. Thus, users are shown items that are hand-selected by others in their social graph who most likely have similar taste, enabling a wider array of high-quality items to be found.

The primary lever for improving feeds on other sites has traditionally been to strengthen the social graph [4, 2]: since organic content comes from the user's social graph, the more connected a user, the more high quality the user's feed will be. For websites that are primarily social networks, users are naturally more motivated to build their social graph. However, Etsy finds a unique challenge when it comes to strengthening its social graph: being primarily an e-commerce site, users don't visit with making connections in mind. This means that the majority of Etsy users have relatively poorly connected social graph, and as a result, sparse feeds. Since sparse feeds leads to a generally uninteresting and poor experience, users are less likely to interact with their AF. In the past, several products were launched to encourage user connections: a friend finder, user recommendations, and referrals. Unfortunately, none of these succeeded in significantly strengthening our social graph. It was clear that a new approach to improving feeds was required.

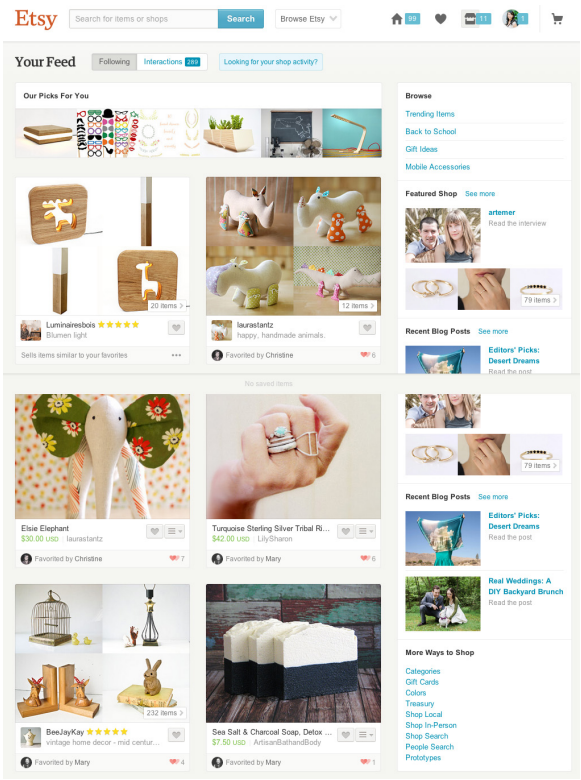

Figure 1: A screenshot of Etsy's Activity Feed. It currently lives on the Etsy homepage for a signed-in user. Each box is a *story*, and each story showcases a single listing, shop, or user.

3. GOALS & CONSIDERATIONS

Our approach to addressing the shortcomings of the previous Activity Feed (Section 2) is to introduce a new class of content: *generated content*. Similar to organic content, generated content also showcases users, listings, and shops in the form of stories (Figure 1). The difference is that the source of the generated content will not come from a user's social graph. Instead, it will come from recommender systems, advertising engines, and other sources from Etsy. This approach has several merits, namely the ability to: 1) to "pad" a user's feeds with interesting content when there is not enough organic activity to fill a full page, and 2) to expose users to more targeted content that should increase overall engagement and conversation rates.

Moreover, introducing generated content provides us with a content delivery mechanism that we have complete control over. We can determine what kind of stories users see, as well as the amount and frequency with which they see them. This not only makes for a better user experience, but also allows us to cater to a huge user base with a wide range of organic activity, from first-time visitors to long-time power users. In this section, we enumerate some of the various constraints and goals we have for this approach:

1. **Respect Users' Organic Activity:** While most users do not have a strong social graph, some more experienced users do - they have a strong aesthetic style and have invested time in finding other users with similar styles and interests to follow. In this case, we want to respect the good Activity Feed experience that they have cultivated and avoid burying user's self-selected organic content with our generated content.

2. **Recommend Relevant & Diverse Content:** Generated content should add value to anyone's Activity Feed experience. A brand new visitor to Etsy should see targeted content that introduces them to the different kinds of items sold on Etsy, and teach them about social features, such as favoriting and following. A more experienced user with a well-cultivated Activity Feed should receive recommendations and advertisements similar to their self-selected organic content, while still providing a glimpse of new styles and trends.

3. **Provide Fresh Content on Each Visit:** Users are more likely to engage and return to the feed if there is promise of fresh content on each visit. Thus, we aim to provide at least one page (10 stories) of fresh content

every time the user visits their Activity Feed:

$$\#\text{ organic stories} + \#\text{ generated stories} >= 10$$

We chose the number 10 as it is roughly the number of stories that can be seen on a user's screen at one time.

4. **Acknowledge User Feedback:** Not all story types should be treated equally: users should receive more of what they like and less of what they don't like. Based on implicit signals, such as click-through rates, we can determine whether a user responds better to one story type over another (e.g. shop advertisements might get a better response than user recommendations). Using this information, we should have the ability to quickly and easily adjust the amount of each generated story type that is included in the user's Activity Feed.

5. **Create a Pleasant Browsing Experience:** Generated content should integrate seamlessly with a user's organic content. For example, we do not want generated content to appear in large clumps: if the user experience consists of an endless supply of new advertisements and recommendations, the Etsy experience quickly becomes impersonal. A user should be able to recognize content in their feed to give them some sense of familiarity. There are also several different story types, each with its own visual experience. We want to avoid showing the same type of content repeatedly in order to prevent users from feeling as if they are "missing out" on other content types.

4. SYSTEM OVERVIEW

In this section, we describe a solution that easily adapts to users with a wide range of organic activity and acts as a platform for delivering interesting, targeted content in a timely fashion. Before delving into the details of our proposed solution however, we first introduce an important concept that the entire system revolves around: the use of pre-computed *content streams*. A content stream is a data construct that includes all of the information needed to render generated stories on the user's feed, be it a recommendation or a targeted advertisement. More specifically, the content stream is represented as a list of time-ordered *story objects*, where each story object includes all information necessary to render a particular story card (e.g. the type and id of the generated content, as well as what time that story should show up on a user's feed). Figure 1 shows an example screenshot of the Etsy AF with rendered story cards. One content stream is stored for each user. For efficiency, all content streams are computed in advance, thus giving us the ability to perform more complex computations such as estimating a user's organic activity level before determining the amount of generated content and at what pace it should be served. It also gives us flexibility in controlling the ratios of different types of targeted content ahead of time, while still being robust to intermittent failures. More broadly, the overarching system consists of four main components:

1. **Generate targeted content**: Dozens of different algorithms are used behind-the-scenes to compute different types of targeted content. These are generated for each of our users, and stored on our Hadoop cluster.

2. **Build content streams**: This process involves aggregating all of the different types of targeted content available for a single user. These currently include a variety of personalized and generic shop, item, and user recommendations. This step also includes estimating a user's organic activity level in order to determine the number of stories that should appear on the user's AF, and at what time they should be displayed.

3. **Store content streams**: Content streams must then be stored so that relevant story objects can be quickly and efficiently retrieved by the Etsy website.

4. **Integrate into the AF**: Retrieved story objects must be integrated with the rest of the user's organic content in a process called aggregation, and are rendered for display.

These components are illustrated in Figure 2 and are described in turn in the following sections.

4.1 Generating Targeted Content

The generated portion of the AF relies solely on automatically generated content sources. Currently, the entirety of our targeted content is composed of shop, user, and listing recommendations. However, this system can be easily extended to advertised content, including advertised shops and listings. Because we must provide feeds to 100% of our user base, we cannot assume that we will have explicit feedback (e.g. favorites, purchases, etc.) from all users to base personalized recommendations on. Thus, we divide our targeted content into two categories: generic and personalized recommendations. We describe both in turn below:

4.1.1 Generic Recommendations

Generic recommendations currently do not use any input signals from the user, and thus addresses our cold-start problem. Instead, they consist of generally popular and trending content that should appeal to a wide range of users, and introduce them to the breadth of the Etsy marketplace. The following is a short summary of the algorithms that power generic recommendations that are used in the AF:

User Recs uses an in-house adaptation of the HITS algorithm [8] to find the most influential users on Etsy. This link-analysis algorithm discovers "authoritative" users as those with many followers.

Shop Recs identifies the most influential users (using HITS), and finds the top favorited shops by these users.

Listing Recs identifies the most influential users (using HITS), and finds the top favorited listings by these users.

4.1.2 Personalized Recommendations

Personalized recommendations are based on explicit user feedback and are only computed for users with a certain amount of prior activity. In Section 4.2, we discuss the requirements for receiving personalized recommendations in more detail. Currently, most of our personalized recommendation systems are based on the "favoriting" signal: users can *favorite* a listing (or shop) that they are interested in, and re-visit them later. Favoriting has been shown to be one of the least noisy signals and best reflects a user's aspirational preferences and style. Since most recommender systems aim to show content that will increase user engagement (not just purchases), using favoriting data as an underlying signal is very desirable. The following is a short summary of algorithms that power personalized recommendations that are used in the AF:

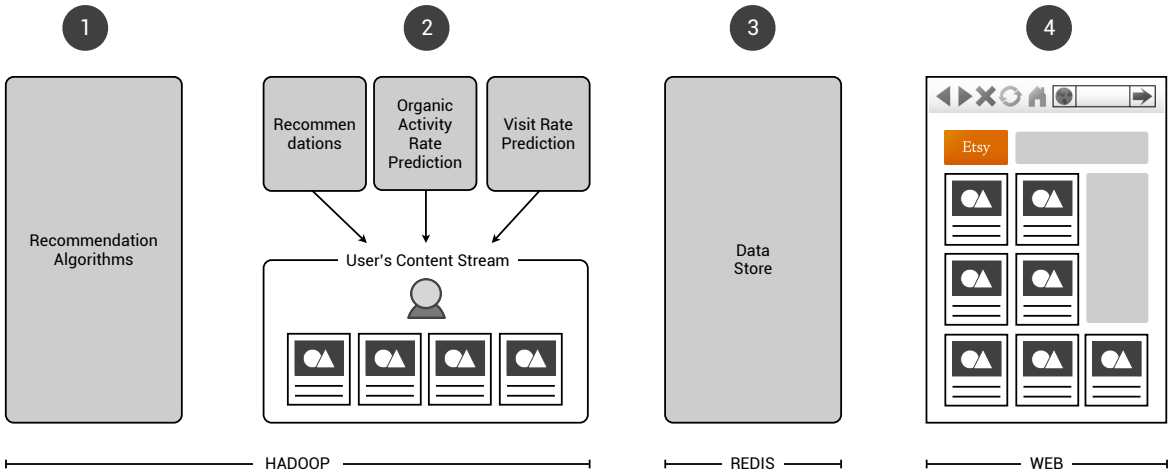

Figure 2: An overview of the four main components underlying the Activity Feed, as described in Section 4

User Recs obtained through factorizing the user and favorited listings matrix [9], using Stochastic SVD [6].

Shop Recs uses Latent Dirichlet Allocation [3] to discover latent topics that are represented as distributions over shops, and represents users as a distribution over topics, and recommends the highest weighted shops in each user's highest weighted topics. See [7] for implementation details.

Listing Recs ranks the favorited listings from a user's user recommendations.

As shown in Figure 2, the entire process of generating recommendations is implemented in a MapReduce framework that runs on our Hadoop cluster. This is necessary in order to scale the recommendation algorithm computation to tens of millions of users. In the next section, we discuss how the resulting recommendations are integrated into each user's content stream.

4.2 Building Content Streams

Building the content stream itself is perhaps the most central and distinguishing component of this system. As described earlier, a content stream is simply a list of time-ordered *story objects* that includes all the information needed to render the generated stories in a user's feed. Each story object contains the information necessary to render a single story card that recommends one item on the AF. They include information such as the *id* of the user that the feed belongs to, the *id* of the item being recommended, the type of recommendation it is (shop, listing, or user), and the time at which the story card should appear on the user's feed.

Content streams are computed regularly by a set of MapReduce jobs and determine the generated stories that each user will see within the next 24 hours. We now face two important questions about what actually goes into each content stream: 1) Content: *What* kind of stories should we include in each user's content stream? 2) Pacing: *When* and *how many* stories should we show on the user's AF? We discuss both questions below.

4.2.1 Content

Targeted content in a user's feed comes from the algorithms discussed in Section 4.1. However, depending on the availability of explicit feedback we have from a user, he or she may not be eligible for all of them. Generic user, shop, and listing recommendations, by nature, are available for all users, but personalized recommendations require users to have a certain number of favorited shops and/or listings. These requirements, though not desirable, are one way to increase our confidence in our recommendations: if we have too little information on a user, it is difficult to generate good recommendations. Improving the coverage of our recommender algorithms is a topic of ongoing research.

The flexible nature of the content stream also allows us to easily adjust the amount of recommendations that we issue from each algorithm. If a user is not eligible for a particular kind of personalized recommendation, we can fall back on the generic version of that recommendation. If we find out that users likes shop recommendations better than user recommendations (Section 5), we can easily manipulate the ratio of shop to user recommendations by simply adjusting some parameters in a config file. These features help satisfy our need for a configurable and modular approach.

After the recommendation stories are determined, they are aggregated into a list that is then shuffled, so that stories of the same type do not show up numerous times in a row. The output is a user-specific list of stories that contain a variety of recommendations for shops, listings and users, both generic and personalized.

4.2.2 Pacing

Once it is clear which types of listing, shop, and user recommendations each user will get, we must determine the number of stories to show, as well as the frequency with which to show them. Recall that our goal is to provide at least 10 new, unseen stories on each user's visit to the AF. This is challenging because some users visit their feed 20 times a day, while others visit their feed once a year. Furthermore, it is nearly impossible to know how much organic activity one will have ahead of time: it is entirely dependent on the amount of users they follow, as well as how active those users happen to be that day. Our approach estimates unknown factors using historical data from each user:

1. **Predict organic activity rate**: Here, we are estimating the user's organic activity per visit. This is done by computing the number of organic stories they have had in the past, normalized by the number of times they have visited the feed. Intuitively, this should be proportional to the number of followed users: the more users one follows, the more organic activity they will have, on average.

2. **Predict visit rate**: This is a simple calculation that averages the number of times a user has visited the AF over a set date range.

With these predicted rates, we can easily estimate 1) the number of times the user will visit the AF in the next day (which is how far out our nightly content stream computations go, keeping in mind that this number will be less than 1 for most users), and 2) the number of organic stories they would see on each visit. From these numbers, we can determine the number of generated stories that must be issued per visit, in order to hit the 10 new story minimum. For each of the generated stories, we can also assign timestamps such that they are evenly issued throughout the relevant time range, beginning from the time of the user's last visit to the AF to the end of the next day (which is as far into the future that the content stream job is responsible for computing). Figure 3 illustrates how, in most cases, issuing generated stories at uniform time intervals intersperses the generated content within the organic stories naturally. As a result, we have populated content streams with various generated stories that are ready to be delivered.

4.3 Storing Content Streams

After generating content streams on Hadoop, we need a way to bring them to the wider web. For this purpose, we use an intermediate datastore. Datastores for content streams have two primary constraints. We are adding an additional step to the feed creation process on the web side, and thus a slow content stream fetch negatively impacts user experience as they would be expected to wait. This gives rise to our first constraint: the datastore must be performant. Content streams contain data that span a wide range of time. We only care about a small portion of this data at any point, which gives rise to our second constraint: the datastore must be able to efficiently filter content streams before passing them to the client.

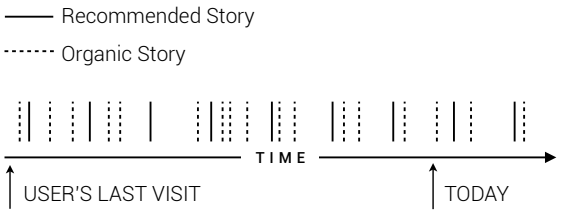

Figure 3: A content stream is computed daily for each user, and is responsible for serving generated stories that are timestamped from the user's last visit to the end of the next day. This figure shows a hypothetical timeline along which a user's stories occur. Note that the generated content (which are timestamped at uniform intervals) is nicely interspersed with their organic content.

Redis is one datastore that meets these constraints. Redis is an in-memory datastore, optimized for speed. It acts much like a key-value store, but boasts support for various data structures. Of particular interest are ZSets, Redis' implementation of a scored, sorted set. Content streams' structure - a timestamp and associated value - lend themselves naturally to ZSets. Redis also allows range queries on ZSets, allowing us to select values by score (which, in our case, is the timestamp for which the story should appear on a user's feed). These queries are performant: 90% of queries return in under 250 microseconds. Compared to the overall AF render, which is around 300 milliseconds, this is quite fast. Thus, Redis provides an efficient go-between for our data generation on the Hadoop side, and our data consumption on the web side.

4.4 Integrating with the Activity Feed

With easy access to our content streams, all that remains is integration with the AF on the web. This step happens when the user visits Etsy, and is responsible for delivering content to the user. Since Etsy's AFis a pull-based architecture, a feed is rendered by first "pulling" (or querying) content from all sources relevant to a particular user. This step is called *aggregation*. Every node in a user's immediate social graph is treated as a content producer. Each node is queried for new content, which is then appended to the user's feed. This is where content streams are integrated.

Content streams are treated like any node in a user's social graph, and queried for new content. Redis's ZSet query support is helpful here: `zrangebyscore` queries are used to find all content with a score greater than the timestamp of last aggregation, but less than our current timestamp. This means that generated content will be delivered only at its prescribed timestamp, and not before. This content is then appended to the feed (sorted by timestamp) like any other content. As noted, this usage of pre-assigned timestamps allows us to naturally intersperse generated stories amongst a user's organic stories, avoiding large bursts of unfamiliar content that might overwhelm the user.

Once a feed has been aggregated, it must be rendered. Prior research has shown that recommendations perform best when presented with context. An example of context may be "based on items you viewed, you may also like this." Here, context is tied to the system that is responsible for generating the targeted content - this may be a recommender system or an advertising engine. The render step consists of linking each piece of targeted content with its source, and passing it off to the feed's standard render process for display on the screen. At this point, our generated content and organic content have been woven together to form a coherent and vibrant feed experience, much like the one shown in Figure 1. AF content can now be consumed on any device, be it through desktop, mobile web, or mobile apps.

5. EXPERIMENTS

Adding content to a feed necessarily affects all other content surrounding it - generated content could overwhelm organic content, making high-value organic content difficult to find. Generated content may also be perceived as low-quality and mis-targeted. Because of the wide variability of feeds in both content sources and quantity of content,

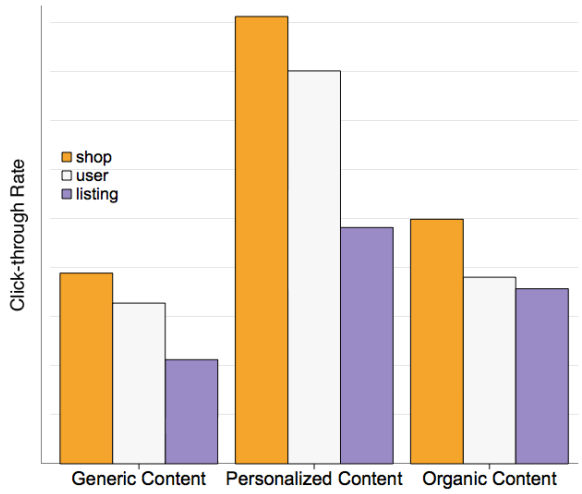

Figure 4: A break-down of CTR for different types of generic, personalized, and organic stories. It is clear that personalized content best engages users.

designing a heuristic to evaluate the quality of all feeds is difficult. Thus, we turn to experimentation to understand the effects content streams have on users.

Etsy's experimental framework is based on *A/B experimentation*, where we select key metrics to monitor as groups of users interact within control or treatment variants. The metrics we chose were: listing view rate, listing favorite rate, and shop favorite rate, all of which are measured on a per-visit basis. These are all core actions that not only lead to purchases, but also allow improvement in the generation of our targeted content. We also monitored the rate of engagement by story type to determine what kind of content engaged users the most.

Because of the configurability of our system, we have introduced a number of parameters to tweak the new AF experience with generated content. For our first experiment, we started with some sane defaults, and it proved to be a success: users were favoriting more listings (+1.07%), favoriting more shops (+8.56%), and visiting more listings (+0.32%). We also noticed that users were more likely to engage with targeted content than organic content by a wide margin (Figure 4). This gave us confidence in our changes and encouraged further experimentation.

With the knowledge that generated content was valuable in the context of the AF, we set out to optimize the blend of content. We launched a 3-variant experiment where each variant had a different ratio of story types: 1) Listing-heavy: received significantly more listing recommendations, 2) Shop-heavy: received significantly more shop recommendations, and 3) Control: received an equal amount of all recommendation types. We found that both listing-heavy and shop-heavy outperformed our control. The shop-heavy variant resulted in lifts of +0.96% to item favorites, +14.94% to shop favorites, and +0.44% to listing views. Compare this to listing-heavy's results of +1.77%, +7.05%, and +0.26% respectively. While we saw significant engagement lifts across all of our metrics in both variants, shop-heavy resulted in the best results overall, and was launched to 100% of users.

6. CONCLUSION

In this paper, we presented an overview of the combined engineering and data science effort that led to a substantial improvement of the existing Etsy Activity Feed, which now acts as a platform to deliver any kind of content - ranging from recommendations to advertisements. From a usability perspective, the Activity Feed is now the homepage to tens of millions of users, and proves to be a pleasant experience, be it their first or 100th visit to Etsy. Users can continue to shape the content of their feed by choosing who to follow, while also receiving targeted content for discovering new items and trends. From an engineering standpoint, we have devised a scalable, modular, and configurable content delivery system that gives us tremendous control over the timing and type of general content that users see everyday. We have shared our framework for experimentation and demonstrated how it guided our decision making. The system has been deployed to 100% of our user base and we have seen a substantial increase in engagement metrics all across the board as a result.

7. ACKNOWLEDGMENTS

The authors would like to thank: Josh Attenberg and Rob Hall for providing insight and work on recommendation algorithms; James Lee and Vernon Thommeret for work on content balance, product requirements, and managing and directing the Activity Feed product development and launch; Junghoon Park and Rachel Nash for providing design input and a flexible home for our content.

8. REFERENCES

[1] D. Agarwal, B.-C. Chen, R. Gupta, J. Hartman, Q. He, A. Iyer, S. Kolar, Y. Ma, P. Shivaswamy, A. Singh, and L. Zhang. Activity ranking in linkedin feed. KDD '14, 2014.

[2] L. Backstrom and J. Leskovec. Supervised random walks: Predicting and recommending links in social networks. WSDM '11, 2011.

[3] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty. Latent dirichlet allocation. *JMLR*, 2003.

[4] M. Burke, C. Marlow, and T. Lento. Feed me: Motivating newcomer contribution in social network sites. CHI '09, 2009.

[5] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. Wtf: The who to follow service at twitter. WWW '13, 2013.

[6] N. Halko, P. Martinsson, and J. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2), 2011.

[7] D. Hu, R. Hall, and J. Attenberg. Style in the long tail: Discovering unique interests with latent variable models in large scale social e-commerce. *KDD*, 2014.

[8] J. Kleinberg. Hubs, authorities, and communities. In *ACM Computer Survey*, 1999.

[9] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[10] M. Zuckerberg, R. Sanghvi, A. Bosworth, C. Cox, A. Sittig, C. Hughes, K. Geminder, and D. Corson. Dynamically providing a news feed about a user of a social network, Feb. 23 2010. US Patent 7,669,123.