Personalizing Natural Language Understanding using Multi-armed Bandits and Implicit Feedback

Fabian Moerchen
Patrick Ernst
Giovanni Zappella
moerchen@amazon.com
peernst@amazon.com
zappella@amazon.de
Amazon Music

ABSTRACT

Natural Language Understanding (NLU) models on voice-controlled speakers face several challenges. In particular, music streaming services have large catalogs, often containing millions of songs, artists, and albums and several thousands of custom playlists and stations. In many cases there is ambiguity and little structural difference between carrier phrases and entity names. In this work, we describe how we leveraged multi-armed bandits in combination with implicit customer feedback to improve accuracy and personalization of responses to voice request in the music domain. Our models are tested in a large-scale industrial system containing several other components. In particular, we focused on using this technology to correct errors made by upstream NLU models and personalize responses based on customer preferences and music provider functionality. The models resulted in significant improvement of playback rate for Amazon Music and are deployed in systems serving several countries and languages. We further used the implicit feedback of the customers to generate weakly labeled training data for the NLU models. This improved the experience for customers using other music providers on all Alexa devices.

CCS CONCEPTS

• **Computing methodologies** → **Online learning settings**; *Learning from implicit feedback*; • **Information systems** → *Personalization.*

KEYWORDS

natural language understanding, multi-armed bandits, personalization, music

ACM Reference Format:
Fabian Moerchen, Patrick Ernst, and Giovanni Zappella. 2020. Personalizing Natural Language Understanding using Multi-armed Bandits and Implicit Feedback. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3340531.3412736

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10.
https://doi.org/10.1145/3340531.3412736

1 INTRODUCTION

Music playback is among the most popular use cases on voice controlled speakers such as Amazon Echo, Google Home, or Apple HomePod. When a customer requests music by voice, a chain of Machine Learning (ML) systems processes the request and returns a response. Automated Speech Recognition (ASR) models transcribe the recorded audio to text, Natural Language Understanding (NLU) models detect the domain (e.g. Music), the intent (e.g. play music), and any named entities (e.g. artist) in the text [28]. Information retrieval models resolve the entity names (e.g. artist = *Kendrick Lamar*) to identifiers (e.g. *B0098PHJSO*). Finally, recommender models and learn-to-rank models [11] are used to provide personalized playback .

There are several challenges with named entity recognition and ranking for the music domain. The music catalogs of streaming services contain millions of songs, artists, albums and thousands of playlists and stations. There is little structural difference between entity names and often ambiguity. For example *Play Dark Side Of The Moon* may refer to an album by *Pink Floyd* or a song by *Lil Wayne*). Short voice commands offer the language model little evidence to distinguish between entity types such as song and album. For example, for *Play Let's Rock by the Black Keys* the top 1 interpretation may be incorrect (artist = *Black Keys*, song = *Let's Rock*}) instead of (artist = *Black Keys*, album = *Let's Rock*). The carrier phrase *play ? by ?* is identical, the album title is also a valid song title (by a much lesser known artist *E-Trax*) and if the training data has a lot more examples for song names than album names it may be more confident about the (incorrect) song interpretation. Finally, the correct answer may depend on the preferred music provider (e.g. Amazon Music, Spotify, Apple Music). For example, *Play Top Country* could be referring to a station for one provider or a playlist for another. Commands like *Play something I haven't heard in a while* may only be supported by some providers.

NLU models are typically trained on manually annotated and synthetically generated data and evaluated on their top k predictions with k=1 for voice applications. Using methods such as Conditional Random Fields (CRF) [17], nowadays often as the last layer in a Deep Learning language model [18], the top k interpretations can be returned along with a confidence score. As explained above, the best interpretation may depend on the music catalog, music provider, and customer preferences. If we wanted to collect manual annotations for the best interpretation, it would require presenting the annotators with rich contextual data about musical preferences

of the customer (sparking privacy concerns), the music catalogs, and supported functionality for each providers.

We present a more scalable approach using online machine learning models to pick the best NLU interpretation among several candidates. A contextual multi-armed bandit models is given features not available to the upstream NLU model and trained with implicit customer feedback. This improves both accuracy and personalization of the NLU selection as measured by the playback rate[1]. in multiple A/B experiments. The systems has passed the test of time by running production for many months without significant escalations.

2 RELATED WORK

Re-ranking algorithms have been studied for different NLU tasks. Entity retrieval is a related line of work, which aims to retrieve specific entities or entity properties to answer short free-text search queries. For example, for a query such as *Rock bands from Seattle* the result would be a list, which ideally include entities like *Nirvana*, *Pearl Jam*, *Foo Fighters*, etc. A set of candidate entities is usually ranked based on term overlap of query terms with entity descriptions as well as semantic similarities defined on types and relationships of entities. To derive this information entity linking is incorporated as a prior step to link mentions from queries to knowledge base entries [9, 13]. Traditional information retrieval methods, such as probabilistic retrieval models [5, 16] as well as learning to rank approaches [12, 22], have been applied for retrieving relevant entities. [27] performs reranking for entity search using random walks over a knowledge graph. Furthermore, the problem was also addressed by various tasks defined in the INEX Entity Ranking (XER) [10] and the TREC 2010 Entity Track [6]. In contrast, we rerank NLU interpretations after each of them has been resolved to an entity. In addition these approaches did not consider personalizing query results in an online learning setting.

The work by [14], applies a Thompson Sampling based multi armed bandit for personalized query suggestions, but does not consider personalizing search results. Methodological related is Sokolov et al. [24], who use bandits to make structured prediction from partial information for individual NLU tasks. They show the applicability of their approach for tasks like sequence labeling, machine translation, but do not consider more complex problems such as query understanding or entity retrieval. Reranking methods based on various algorithms, such as boosting and perceptron methods [8], kernel-based approaches [20], or deep neural networks [29], have been applied to improve the output of baseline NER systems. Beyond NER, Sil and Yates [23] takes a large set of candidate mentions from NER and a large set of candidate entities, and reranks mention-entity pairs. The aforementioned approaches only consider individual tasks within NLU pipelines, contrary to our usecase where we globally rerank NLU interpretations across many tasks. Similarly, the work of Su et al. [25] applies a linear model which combines calibrated local ranking scores from domain-specific NLU modules, to globally rerank a list of NLU hypotheses. However, the approach does not consider personalization or the resolved entities.

[1]The fraction of voice requests where music played for at least K seconds.

3 PROPOSED APPROACH

Our input is a ranked list of NLU interpretations generated by an NLU model with confidence score and the identifiers of the musical entities these interpretations were resolved to. Selecting one interpretation and the associated musical content for playback on voice controlled devices is a multi-armed bandit problem as we can only observe the customer reaction for what we selected and the available historical data was limited to the ”most likely” interpretation ranked at the top by the NLU model confidence score. Many alternative and possibly better interpretations had never been exposed to customers. Learn-to-rank models trained offline would have been biased toward the interpretations with the highest NLU confidence scores. Such models may thus not surface alternatives without explicitly enforcing this. Instead we chose to use online learning of contextual multi-armed bandit models that offer a principled approach for exploration.

Over the years, several algorithms have been developed to solve this problem and its variants [7, 19]. Many bandit algorithms actively utilize the uncertainty of the model to efficiently trade-off exploration and exploitation: chosing to select an action known to have good performance when uncertainty is low vs. higher likelihood of exploring new options to see if they are possibly better when uncertainty is high. We use Thompson Sampling [3] drawing model parameters from the learned distribution leading to almost deterministic explotation when there is one clear winner in terms of predicted reward and exploration when there are several candidates with similar features and hence similar reward estimates.

In addition the capabilities of streaming services to respond to a certain voice requests and the content in the music catalog changes over time. An online learning model can adapt to such scenarios automatically.

We decorate the NLU interpretations with additional features that are potentially helpful in identifying the best interpretation and not available to the upstream NLU model such as popularity of the musical entity and customer preferences.

The models are trained with a binary reward (1 for at least K seconds playback, 0 otherwise) that matches the playback rate metric we want to optimize. This implicit signal is noisy, as some customers may change their mind within the first K seconds even of the response was correct and other may play an incorrect response play longer our of curiosity or because they are distracted. But machine learning models are robust to small amounts of label noise and we can collect the signal fully automatically for very large volumes of requests.

3.1 Impact estimation

To justify the investments into development and deployment of (additional) ML models we analyzed the potential impact at the start of the project. We analyzed the potential of utilizing the top-k NLU interpretations for improving the playback rate for Amazon Music customers, our targeted success metric. We observed that the playback rate was negatively correlated with the number of interpretations available. In other words, the higher the ambiguity of an request, the less likely the interpretation currently ranked at the top by the NLU model was working well. We found that for 72% of unfulfilled requests (no playback started) and 45% of

abandoned requests (playback stopped within K seconds) we had more than one NLU hypothesis available. Re-ranking the set of NLU interpretations to explore alternatives to the top 1 by NLU model confidence score seemed like a promising direction.

3.2 Online learning with explore/exploit

We formalize the problem described above as a sequential decision problem where in each time step $t = 1, \ldots, T$ a set of candidate entities in the form of actions $A_t = a_1, \ldots, a_k$ is presented to the learner. To increase readability, without loss of generality, we will assume k to be fixed. Each action is represented by a vector in $\mathbb{R}^d$ and are assumed to have unit length. This representation allows applying the model to use cases where the universe of actions is very large and not all actions are known at design time. In our case the action space consists of the top k NLU interpretations for any customer request that is identified by upstream models as a music request.

The learner is a linear Thompson Sampling [1, 4] implementation where for each prediction the parameters of a linear function are sampled from a multivariate Gaussian posterior and all the actions are scored with the same parameters of the linear function. The action with the highest score is then selected.

The action vectors contain information both from the context and the actions and are transformed in order to capture the interaction between the two. Namely, provided a vector x representing the context (e.g., information about the device) and a set of vectors $Z = z_1, \ldots, z_k$ representing the candidate actions (e.g. NLU confidence, entity popularity, etc.), the set of actions A is generated as the set containing the vectors where $\forall i \in 1, \ldots, ka_i = (x, z_i, \phi_{x,z_i})$, ϕ_{x,z_i} is a non-linear transformation of the vector obtained by the Kronecker product between z_i and x and a random projection to reduce the dimensionality.

In our experiments, we use Thompson Sampling with a linear assumption where the weight vector $\hat{\theta}_t$ used for the predictions at time t is sampled from the following posterior distribution:

$$\mathcal{N}(\theta_t, (\sigma^2 V_t + \lambda I)^{-1})$$

where V_t is the empirical covariance matrix of the actions played so far and it is defined as $\sum_{i=1}^{t} a_i a_i^T$. Defined $b_t = \sum_{i=1}^{t} r_i a_i$ where r_t is the observed reward at time t. The mean of the posterior is computed as $\theta_t = (\sigma^2 V_t + \lambda I)^{-1} b_t$.

As it can be observed in the equations above, the dimensionality reduction of the non-linear interaction features is necessary for many practical use-cases due to the $O(d^2)$ time complexity in both the prediction and update phase.

3.3 Offline ranking feature analysis

We performed an offline analysis of ranking features to de-risk the online learning and be cost-efficient in the production implementation. The upstream NLU model uses only the transcription of the voice request as input and ranks NLU interpretations by the model's confidence. We analyzed many candidate input features available at the re-ranking stage to identify the most useful features, de-skew their distributions and map the ranges to approximately $[0, 1]$. Well behaved and normalized input feature distributions aid the linear model in learning weights that indicate feature importance without also having to normalize ranges. The implementation of each ranking feature in the production system comes with a cost and delays the launch of the first experiment. We hence used predictive models trained offline to prioritize the features to be implemented in production. Our offline analyses was neccessarily based on data limited to NLU interpretation ranked by the NLU model at position 1, but this was the only data available in this stage of the project.

Boolean features: We initially generated about 50 boolean features representing each slot name we observed in at least one NLU interpretation and the most common values of the slot media type (e.g. Station) by coverage. We prioritized them by coverage and selected 12 features with at least 1% coverage, including *has song, has media type, has sort type*).

Categorical features: The categorical features included the NLU rank (1 through k) and customer-to-artist affinity (*weak*, *medium*, *strong*). The affinities are calculated by aggregating listening events by a customer to tracks from an artist with time decay and binning.

The numerical features we considered included:

NLU confidence, maximum confidence among the NLU interpretation for the same request, and confidence normalized by dividing by this maximum. NLU confidence had full coverage and a bi-modal distribution with most values either close to 0 or 1.

Entity resolution (ER) confidence for each entity type (e.g. song) with maximum and normalized version. We used clipping with different thresholds to remove outliers and normalized all scores to $[0, 1]$. ER confidences are sparse because they are only available if the NLU interpretation detected the corresponding entity. We generated a consolidated ER confidence corresponding to how playback use cases are triggered, in other words the confidence for the entity that would be played. Starting with song confidence we replace missing values with lyrics, then album, playlist, station, artist, genre. Remaining missing values and missing values in each individual ER confidence were replaced with 0.

Playback rate of an NLU interpretation represented by the lower bounds of the 95% confidence intervals over 5 different trailing time windows (7, 14, 28, 140 days and 1 year). The coverage of estimates for playback rates is naturally lower for shorter time windows, because fewer torso and tail queries are observed. We generated a consolidated playback rate feature using the maximum of the lower bounds across time windows, in other words the query works at least this well in some time period. For rare queries longer windows would have a higher lower bound because more observations contribute to the estimate. Frequent queries may have a higher lower bound in shorter time windows for example when problems with queries were fixed by business rules or other learning systems. The distribution was bi-modal with peaks toward 0 and 1, missing values were replaced with 0.

Query popularity: Popularity of the query with Amazon Music customers over the same time windows as playback rate. The popularity of voice queries had a very skewed distribution. We applied the transformation $log((\text{popularity} + 1)/\text{days in time window})/C$ with C chosen such that the distribution is approximately in $[0, 1]$ and +1 to account for queries observed only in long time windows, and normalization by number of days to make values comparable across time windows. The latter enables creation of a consolidated

version with the maximum of the transformed popularity score across time windows.

Entity popularity: Popularity of the resolved entity with Amazon Music customers over 2 different time windows (7, 140 days). The popularities available in production were already normalized to $[0, 1]$ by division with the maximum value, but still left skewed. We used clipping to remove outliers and roots with varying degrees to arrive at similar distributions centered in $[0, 1]$. We calculated a consolidated score across entity types for each time window analogous to ER confidence scores.

Utterance overlap: Any given NLU interpretation may capture only a subset of the tokens in the utterance, because tokens labeled as 'Other' by the upstream NLU model are dropped for privacy protection. We calculated the Jaccard similarity between the multi-set of tokens in the slots of an NLU interpretation with the tokens in the utterance. Since we did not actually have the utterance available, we approximated the multi-set of tokens in the utterance with a multi-set across the k available NLU interpretations keeping each token as many times as it's maximum frequency in any of the individual interpretations. This feature indicates how faithfully an NLU interpretation represents the (approximated) utterance. The distribution was heavily right skewed toward 1.0 as expected, because most NLU interpretations preserve most tokens.

Missing value flags: We generated boolean features indicating for each numerical feature, whether a missing value was replaced. Since missing values were replaced with zeroes, the weight a model learns for the indicator features acts as a learned replacement value.

Most of the features described above are action features, i.e. they can have different values for each NLU interpretation. Context features are limited to a few summary statistics like the number of NLU interpretations and the maximum NLU confidence.

Feature correlation: We analyzed pairwise scatter plots among the numerical features to check for highly correlated features. We observed high correlation (>0.9) among related features (e.g. among playback rates estimated over different time windows) but much lower (<0.6) across feature groups. Thus it seemed promising to use models to learn a ranking function combining many different signals.

Offline feature selection: We used an offline workflow to select and prioritize the implementation of features in production. We used 8M requests with at least two resolved NLU interpretations to be ranked resulting for a total of 18M NLU interpretations. We sampled 100K utterances and kept 30% as holdout evaluation. We split the rest into 70% for training and 30% for testing and generated a dataset with the ranking features and the outcome (1 for at least K seconds playback, 0 otherwise) for the NLU interpretation that was selected in production. In other words we are training a point-wise ranking model by trying to predict the success of the NLU interpretation neccessarily limited to the ones deemed most likely by the upstream NLU model. We trained Logistic Regression models as a proxy for the contextual bandit models we planed to use online. We used 10-fold cross-validation on the training data to optimize hyper-parameters and then compared different feature sets using the testing data. We found that using a small set of six features we expected to be critical (NLU confidence, consolidated playback rate, consolidated ER confidence, consolidated query popularity, consolidated entity popularity, and utterance overlap) performed well. However, adding any of the following helped improve accuracy further: binary flags, customer-to-artist affinity, contextual information, more granular playback rate/popularity/confidence features, and interaction features (pairwise products). The data was of course strongly biased toward NLU interpretations ranked at k=1 by NLU confidence. However lacking other less biased or randomized data, we used the results to prioritized the implementation of features production.

Offline simulation: We used offline simulation to get a sense of the risk of using an ML model to re-rank the NLU interpretations would be. We used both training and testing data with the identified hyper-parameters and feature set to train a final LR model and applied it to all NLU candidates of the holdout data to simulate ranking decisions. For 79% of the requests, the model ranked the same NLU interpretation at the top as the production system. These requests had a playback rate that was about 10% higher than the overall average. This indicated that the model was making good decisions on this subset. We simulated our triggering criteria on the remaining 21% where the model ranked an alternate NLU interpretation at the top. The lower bound of the playback rate of the current NLU interpretation needed to be $< 25\%$ and and the lower bound of the alternate NLU interpretation $> 25\%$ or unknown (i.e. never or not often tried). We found that the model would trigger on 3.8% of utterances. We sanity checked the changes the model would make on the most popular utterances and they looked promising, further increasing our confidence for the online experiment.

4 ONLINE EXPERIMENT

4.1 Baseline

In order to evaluate the strength of an ML model, it is common practice to compare it with a baseline to better understand if the investment and the complexity introduced in the system are justified. We thus started comparing with a simple heuristic: for voice requests where the top 1 interpretation had not resulted in playback in the past, we traversed the top-k NLU interpretations in order of the confidence score of the upstream NLU model until we would find one that would (if any). The heuristic was later refined introducing additional business rules to prevent certain fallbacks such as not selecting an interpretation k>1 with song if the k=1 interpretation had a genre. For example song = *ballerina* instead of genre = *ballerina* was surfacing a rock song with *ballerina* in the title and while playback rate was improved (now >0%) it was still very low. The A/B test[2] resulted in an increase of playback rate of +0.18% [3].

While the guardrails prevented many poor experiences, they also eliminated some good fixes. We further analyzed offline whether the estimated playback rate of an NLU interpretation based on historical data could be used for ranking *instead* of the confidence from the NLU model. We found a mix of good and poor decisions and concluded that there was merit in pursuing an ML approach where we provide the model with many different features and let it optimize the decision function based on implicit feedback.

[2] All A/B tests reported split customers into two equal groups and were evaluated on 1 or 2 weeks of data.

[3] All results from A/B tests are reported as absolute changes to playback rate and are statistically significant with p-values (much) smaller than 0.01

4.2 Model analysis

We ran an online experiment in order to verify the quality of our solution in November 2018. We monitored the online learning model by analyzing the time series of the model parameters and the decision consistency of the model. Recall that the bandit model maintains parameters describing a distribution for each model parameter and the Thompson Sampling draws a specific model parameter vector from these distributions for every request. We calculated the mean and variance of these sampled model vectors over 1h time windows. The time series of means is shown in Figure 1 (left). We observed the mean values diverging rapidly from their initial small random values and then stabilizing somewhat after only a few hours. They kept on changing gradually over the next days. On 11/13 some of the weights with medium size changed again within a few hours. This happened when we switched the US from a local to the global music catalog which may have caused changes in some of the input features. Another change happened on 11/15 and neither affected the playback rate. This demonstrates how an online learning algorithm can adapt to a changing environment. The trend of variances is shown in Figure 1 (right). Variances for large (important) weights quickly became small (< 1), variances of contextual features that only affect the decision through the separate random projection features had medium size (about 5). The remaining variances corresponded to sparse features with very low weights (no significant impact on decisions). They kept growing for a few days until leveling out at about 20.

Even if the model has uncertainty and keeps updating its parameters, this does not necessarily mean a lot of the decisions change. We calculated the consistency of decisions as follows: We draw a random sample of size 100 from each hour of requests available in the log data. Each row had the NLU interpretations with associated feature vectors, the sampled model vector, and the winning NLU interpretation. We simulated which interpretation would have been picked had we used model vectors from different time window and recorded how many times it would have been the same. Figure 2 (left) shows the consistency within each hour. It is almost immediately above 95% and around 99% after a few days. Figure 2 (right) shows the consistency with previous hours, measuring how much the model has converged. We can see that after about 10h of learning the decision consistency with the previous hours was above 95% and was near 100% after 3 days.

4.3 A/B experiment analysis

We measured an increase in playback rate of +0.37% in US and +0.41% in UK between control and treatment. This was a significant incremental improvement over the previously launched baseline described in Section 4.1. To put this into context, this largest increase in playback rate from a single project in Amazon Music that year.

To dive deep into the changes made by the model we grouped voice requests by the sorted list of tokens from all NLU slots as an approximation of the utterance and looked for the most improved and most degraded examples between control and treatment. A popular improved utterance was *Play Stir-fry* for which the model correctly selected song instead of album. Examples of improved utterances are shown in Table 1.

Table 1: Examples of improved utterances.

Request (play...)	Control	Treatment	Playback Δ
china	ArtistName	SongName	+71%
stir-fry	ArtistName	SongName	+70%
hopeless romantic	AlbumName	SongName	+49%

4.4 Model decision analysis

After launch we analyzed the playback requests where the bandit selected an NLU candidate other than the first one. The dataset consisted of 5M utterances for 1.2M distinct pairs of k=1 and k>1 NLU interpretations. We categorized the NLU changes by the slots present in both NLU interpretations and inspected the categories with at least 80% playback rate. Examples include:

Song to Album: The tokens labeled as song in the k=1 interpretation are labeled as album in the N>1 interpretation (e.g. {artist = *Lil Wayne*, song = *Carter Five*} vs. {artist = *Lil Wayne*, album = *Carter Five*}. These are clear NLU error corrections.

Remix: k=1 has genre *Remix* and an artist slot. k>1 has song ending in the artist and the token *Remix* (e.g. *Mic Drop Steve Aoki Remix*). This works better because the remixing artist is not the primary artist of the song but indexed in the song title field.

Greatest hits: The k=1 interpretation has album and artist and the album has values like *Greatest Hits*, *The Greatest Hits*, *Best Hits*. The artist frequently has a possessive suffix (e.g. *Led Zeppelin's Greatest Hits*. Instead of album, the N>1 interpretation has sort type *BEST*. The model learned that using the sort type, which on Amazon Music results in shuffling the most popular songs by an artist, works better than trying to find a specific album by name.

Genre to Playlist: Genre to playlist: Many of these could be playlist names (*Rock Anthems*, *Dance Cardio*, *Nineties One Hit Wonders*) others are multiple genres concatenated (*R and B Soul*). The model learned that for Amazon Music it is better to look for a playlist matching the keywords, possibly because playlists curated by musical experts are of higher quality when available.

We also analyzed categories where the NLU ranking decisions results in poor playback rates (less than 40%):

Song to Genre: The most frequent tokens assigned to song in k=1 and genre in k>1 were *easy*, *night*, *old*. It is hard to say whether customers wanted a song or a genre. The model may be over-eager in picking NLU interpretations with genre because they work really well for head genres like *Country*. It may help to provide the model with an explicit signal indicating which genres are supported.

Song to Device,Song: Keywords in the song for k=1 are assigned to device in k>1. For example song = *Burn the whole house down* becomes {device = *Whole house*, song = *Burn*}. We suspect that the model tends to pick NLU interpretations with Device present because most of the time the Device slot is correct and this works better than ignoring the request for a specific (likely different) device. But in these cases device is a false positive. Other tokens frequently in device were *deck*, *all*, *home*, *room*.

Genre to Artist: Many of these are long tail genre or instrumentation requests (*Beatbox*, *Saxophone*, *Tamil*, *Steel drums*). For some the artist interpretation actually worked surpisingly well (e.g. *Steel*

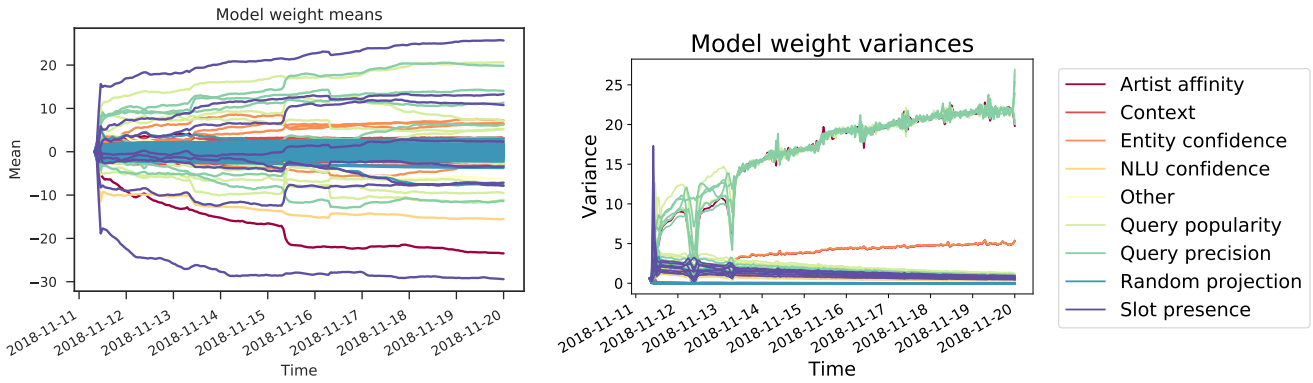

Figure 1: Mean and variance of model weights per feature during the initial learning phase of the bandit model.

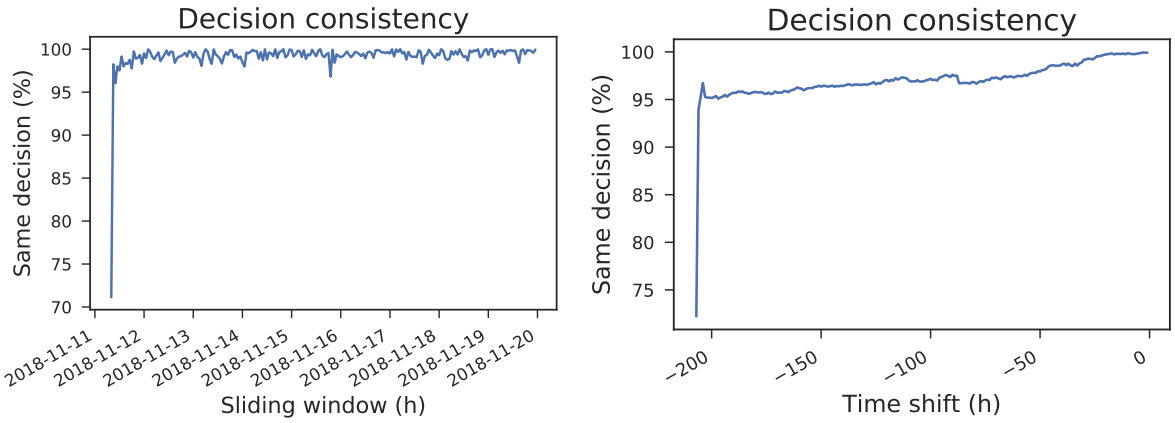

Figure 2: Decision consistency within each hour and with previous hours during the initial learning phase of the bandit model.

drums as artist had 89% success) but it seems we need to add support for these genre requests and try to first find matching playlists.

4.5 NLU model improvements

We have demonstrated that the bandit models correct many NLU ranking errors made by the upstream NLU models. Instead of fixing them only for Amazon Music customers, we investigated whether we can use successful decisions from the bandit to improve the upstream models and prevent ranking errors from occurring in the first place. We selected the over-performing categories of k>1 picks that would likely benefit all music streaming services and generated a dataset with 406K utterances for 53K under-performing NLU interpretations along with the well performing NLU interpretations picked by our ranker and considered this the ground truth annotation. We performed offline experiments augmenting the training and testing data. We observed a 55% reduction in errors on the newly added test set. In addition the data helped to identify and remove an under-performing post-processing rule that was changing the slot from artist to playlist, if the artist name was followed by 'playlist/hits/mixes' token. An updated NLU model was tested and released to production in March 2019 after an A/B test indicating significant improvement for two other music streaming services on Alexa enabled devices.

5 TEST OF TIME

The models launched in US (+0.37%) and UK (+0.41%) in 2018 strongly contributed to improving playback rate for Amazon Music customers. In 2019 we repeated this success by launching additional models in non-English locales with even higher impact (DE: +0.68%, FR:+0.15%, IT: +0.73%, ES: +0.83%, JP: +0.93%). This indicates that our international NLU models can benefit even more from utilizing implicit customer feedback.

There have been very few problems with these models running in production. Generalizing the benefit produced in the US to other countries and languages was straightforward and allowed us to scale quickly. While this may be considered a secondary aspect compared to the increase of performance, the ability to easily maintain and quickly scale to different locales is an important aspect for many industrial machine learning systems.

The only problems we encountered so far are related to the launch of new functionalities in the streaming service. Before launching a new functionality there is a significant amount of testing on live systems and the test traffic can negatively impact

the model, especially since the reward signal is constantly negative. In these cases it is enough to blacklist re-ranking for new features until enough real life performance data has been collected.

Overall, our approach has passed the test of time and has become a key mechanism for Amazon Music to improve the experience for our customers correcting upstream NLU ranking errors and personalizing responses. In addition we are contributing to upstream models and hence the success of Alexa devices overall also for customers who prefer other music streaming services.

6 DISCUSSION

Our experiments demonstrate that using a rich set of features, online learning models trained with implicit feedback, and the explore/exploit paradigm provides a significant added value over the existing NLU modeling and ranking solutions. The models have been deployed into production for many months and passed the test of time with no significant escalations. While our deployment is limited to the Music domain, the general approach is applicaple to other domains where there are multiple plausible NLU interpretation and the right one for a specific request depends on context and personalization.

There are several feedback loops in our system at different time scales. The implicit feedback from customers is used to update the bandit model every few minutes and allows it to adapt to concept drift in the features, changes in the catalog, and changes in the implemented functionality. In addition the implicit feedback is aggregated daily for each selected NLU interpretation over time and provided to the bandit model as an input feature. This reinforces up-ranking of well performing interpretations (the more often they are tried and work, the higher their playback rate) and down-ranking under-performing interpretations. The learnings in these features persist even when new bandit models with additional features are tested. Finally we derive weakly labeled training data for the upstream NLU models using over-performing picks aggregated over long time windows.

In addition to correcting upstream NLU ranking errors, we found cases where there are multiple plausible NLU interpretations and personalization of the response can help optimize the customer experience. Personalization to the music streaming service used can be beneficial when different services have different abilities or strategies to respond to a request, such interpreting a keyword as a genre vs. a playlist name. Personalization to the customer's musical preference is also needed when there are multiple plausible interpretations and the best one depends on the customer's music preferences. The affinity of the customer to the corresponding artists or genre provides evidence for picking the right interpretation. As a music streaming service, Amazon Music can calculate this preference using playback from voice and visual clients. Thus even when NLU errors from the upstream models will reduce there is a long term benefit to having a re-ranking for each music service.

7 CONCLUSIONS

We demonstrated how contextual multi-arm bandits trained with implicit feedback can be used to improve and personalize customer experience for voice controlled music playback. We described many practical aspects of real world problems including impact estimation, using data biased toward the top interpretation from the NLU model to prioritize feature implementation, monitoring of online learning models, and post-experiment analysis. We provided insights into both good and poor model decisions and measured significant positive impact overall using A/B experiments.

We believe there are opportunities for improvement activating the re-ranking on more traffic and adding additional input features. For example, ambiguous requests where a popular result competes with nich results may overall not have poor enough performance for the re-ranker to step in. Features describing personalized preferences and the current session may further boost performance. On the modelling side, we plan to experiment with models which can directly learn non-linear interactions and transformations (e.g., [15, 21]) to reduce the effort required for feature engineering and further increase scalability. We are also considering different approaches to leverage biased datasets (e.g., [26] [2]) which may decrease the amount of operations and maintenance needed for the service and can reduce cost to the customer experience during the first hours of online model learning when explore rates are high.

ACKNOWLEDGMENTS

We thank Kevin Sonnen for implementing the ML model in production, Prajiit Reddy Muppidi for incorporating our data into NLU models, and the Amazon Music ML team in Berlin for developing the bandit service used in these experiments.

REFERENCES

[1] Marc Abeille, Alessandro Lazaric, et al. 2017. Linear thompson sampling revisited. *Electronic Journal of Statistics* 11, 2 (2017), 5165–5197.
[2] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. 2015. A General Framework for Counterfactual Learning-to-Rank. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*.
[3] Shipra Agrawal and Navin Goyal. 2012. Analysis of Thompson Sampling for the multi-armed bandit problem. In *International Conference on Learning Theory*.
[4] Shipra Agrawal and Navin Goyal. 2013. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*. 127–135.
[5] Krisztian Balog, Marc Bron, and Maarten De Rijke. 2011. Query modeling for entity search based on terms, categories, and examples. *ACM Transactions on Information Systems (TOIS)* 29, 4 (2011), 22.
[6] Krisztian Balog, Pavel Serdyukov, and Arjen P. de Vries. 2011. Overview of the TREC 2011 Entity Track. In *TREC*. NIST Special Publication.
[7] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning* 5, 1 (2012), 1–122.
[8] Michael Collins. 2002. Ranking Algorithms for Named-entity Extraction: Boosting and the Voted Perceptron. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* (Philadelphia, Pennsylvania) *(ACL '02)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 489–496.
[9] Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rued, and Hinrich Schuetze. 2018. SMAPH: A Piggyback Approach for Entity-Linking in Web Queries. *ACM Trans. Inf. Syst.* 37, 1, Article 13 (2018), 42 pages.
[10] Gianluca Demartini, Tereza Iofciu, and Arjen P. de Vries. 2010. Overview of the INEX 2009 Entity Ranking Track. In *Focused Retrieval and Evaluation*, Shlomo Geva, Jaap Kamps, and Andrew Trotman (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 254–264.
[11] Chris Burges et al. 2005. Learning to Rank using Gradient Descent. In *22nd International Conference on Machine Learning*.
[12] Dario Garigliotti, Faegheh Hasibi, and Krisztian Balog. 2018. Identifying and exploiting target entity type information for ad hoc entity retrieval. *Information Retrieval Journal* (05 Dec 2018).
[13] Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. Exploiting Entity Linking in Queries for Entity Retrieval. In *Proceedings of the 2016 International Conference on the Theory of Information Retrieval* (Newark, Delaware, USA) *(ICTIR '16)*. ACM, New York, NY, USA, 209–218.

[14] Chu-Cheng Hsieh, James Neufeld, Tracy King, and Junghoo Cho. 2015. Efficient Approximate Thompson Sampling for Search Query Recommendation. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing* (Salamanca, Spain) *(SAC '15)*. ACM, New York, NY, USA, 740–746.

[15] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. 2018. Deep learning with logged bandit feedback. (2018).

[16] Rianne Kaptein, Pavel Serdyukov, Arjen De Vries, and Jaap Kamps. 2010. Entity Ranking Using Wikipedia As a Pivot. In *Proceedings of the 19th International Conference on Information and Knowledge Management* (Toronto, ON, Canada) *(CIKM '10)*. ACM, New York, NY, USA, 69–78.

[17] John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *18th International Conf. on Machine Learning*.

[18] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of NAACL 2016*.

[19] Tor Lattimore and Csaba Szepesvári. 2018. Bandit algorithms. (2018). https://tor-lattimore.com/downloads/book/book.pdf

[20] Truc-Vien T. Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2010. Kernel-based Reranking for Named-entity Extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics* (Beijing, China) *(COLING '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 901–909.

[21] Carlos Riquelme, George Tucker, and Jasper Snoek. 2018. Deep Bayesian Bandits Showdown: An Empirical Comparison of Bayesian Deep Networks for Thompson Sampling. In *ICLR 2018*.

[22] Uma Sawant and Soumen Chakrabarti. 2013. Learning Joint Query Interpretation and Response Ranking. In *Proceedings of the 22Nd International Conference on World Wide Web* (Rio de Janeiro, Brazil) *(WWW '13)*. ACM, New York, NY, USA, 1099–1110.

[23] Avirup Sil and Alexander Yates. 2013. Re-ranking for Joint Named-entity Recognition and Linking. In *Proceedings of the 22Nd International Conference on Information & Knowledge Management* (San Francisco, California, USA) *(CIKM '13)*. ACM, New York, NY, USA, 2369–2374.

[24] Artem Sokolov, Julia Kreutzer, Christopher Lo, and Stefan Riezler. 2016. Learning Structured Predictors from Bandit Feedback for Interactive NLP. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (Berlin, Germany). Association for Computational Linguistics, 1610–1620.

[25] Chengwei Su, Rahul Gupta, Shankar Ananthakrishnan, and Spyros Matsoukas. 2018. A Re-Ranker Scheme For Integrating Large Scale NLU Models. In *2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, December 18-21, 2018*. 670–676.

[26] Adith Swaminathan and Thorsten Joachims. 2015. The Self-Normalized Estimator for Counterfactual Learning. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*.

[27] P. Torres-Tramón, M. Timilsina, and C. Hayes. 2019. A Diffusion-Based Method for Entity Search. In *IEEE 13th Conference on Semantic Computing (ICSC)*. 16–23.

[28] Vikas Yadav and Steven Bethard. 2018. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. In *COLING 2018*.

[29] Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural Reranking for Named Entity Recognition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*. Varna, Bulgaria, 784–792.