Bootstrapping a Music Voice Assistant with Weak Supervision

Sergio Oramas* and **Massimo Quadrana*** and **Fabien Gouyon**
Pandora Media LLC.
Oakland, USA
{soramas, mquadrana, fgouyon}@pandora.com

Abstract

One of the first building blocks to create a voice assistant is the task of tagging entities or attributes in user queries. This can be particularly challenging when the number of entities are in the tenth of millions, as is the case of music catalogs. Training slot tagging models at an industrial scale requires large quantities of accurately labeled user queries, which are often hard and costly to gather. On the other hand, voice assistants typically collect plenty of unlabeled queries that often remain unexploited. This paper presents a weakly-supervised methodology to label large amounts of voice query logs, enhanced with a manual filtering step. Our experimental evaluations show that slot tagging models trained on weakly-supervised data outperform models trained on hand-annotated or synthetic data, at a lower cost. Further, manual filtering of weakly-supervised data leads to a very significant reduction in Sentence Error Rate, while allowing us to drastically reduce human curation efforts from weeks to hours, with respect to hand-annotation of queries. The method is applied to successfully bootstrap a slot tagging system for a major music streaming service that currently serves several tens of thousands of daily voice queries.

1 Introduction

Music listening is among the top-5 reasons of daily usage of voice assistants in the US.[1] Users can have different goals when formulating a music-related query to their home voice assistant or mobile phones. For instance, users may look for a specific entity, which can be either explicit (e.g., "play Led Zeppelin") or implicit (e.g., "play the latest album by Foo Fighters"). They may also ask queries without having a specific entity in mind (e.g., "play some reggae music"), or make open-ended requests like "play something that I like" (Ostuni, 2019; Volokhin and Agichtein, 2018).

Given a transcribed voice query, a fundamental task towards its understanding is to identify entities and musical attributes in it. However, this can be a non-trivial task, especially when the catalog is composed of possibly millions of different entities. In such situations, the chances that the name of one entity will overlap even partially with another entity are non-negligible. It is even more likely to find overlaps between entities and musical attributes, or between entities and other commonly-used natural language phrases in the query (Guy, 2018). For example, the word "happy" is at the same time a song by Pharrell Williams and an attribute belonging to the "Mood" category in our taxonomy of musical attributes. Mislabeling entities in a user query can potentially lead to awkward user experiences.

Slot tagging, or slot filling, is traditionally tackled as a supervised sequence labeling problem and it is often based on methods such as Recurrent Neural Networks (Goyal et al., 2018), Conditional Random Fields (Reimers and Gurevych, 2017) or pre-trained language models like BERT (Chen et al., 2019). In real-world industrial applications, however, the choice and optimization of the Machine Learning architecture is just the tip of the iceberg. Most of the time and cost are actually spent in gathering sufficient accurately-labeled training data. This process generally requires the manual annotation of up to millions of user queries, a process that should be routinely repeated to keep up with natural drifts in user queries due to, e.g., new interests from users or items that are added or removed from the catalog of searchable products. Manual annotation can be complemented, or even replaced, with synthetically generated training data based on patterns curated by experts (Goyal et al., 2018). While synthetic generation unlocks the possibility of gathering nearly infinite labeled training sam-

*Equal contribution.

[1]Source: https://www.pwc.com/us/en/advisory-services/publications/consumer-intelligence-series/voice-assistants.pdf

Proceedings of NAACL HLT 2021: IndustryTrack Papers, pages 49–55
June 6–11, 2021.

ples, it still requires solid domain expertise to create a sufficiently rich set of patterns to cover as many query variations as possible.

Both manual annotation and generation of training data requiring a significant financial and human resources; another line of thoughts is to exploit unlabeled query data, which is generally cheap and abundant, and to label it via weak supervision.

Weak supervision –or distant supervision– has demonstrated its suitability to a number of natural language processing tasks such as relation extraction (Mintz et al., 2009) or entity recognition (Lison et al., 2020). Moreover, it has been shown as a useful method to bootstrap conversational systems, being applied to intent detection (Mallinar et al., 2019) or slot tagging (Surdeanu et al., 2011) tasks. Given this success, flexible frameworks like Snorkel (Ratner et al., 2017) have been created to help on building weak supervision pipelines at scale. However, these frameworks are not easily adaptable to sequence labeling tasks (Lison et al., 2020).

In this paper, we present our own methodology inspired by weak supervision to label large sets of transcribed voice queries with entities and attributes from a catalog with millions of entries. The resulting labels are, albeit noisy, sufficiently accurate to be used for training slot tagging models. We show how our methodology allows us to control the amount of noisy labels injected in the training dataset by combining weak supervision and human filtering, and provide experimental evidence of how it was exploited to successfully bootstrap a slot tagging system that now serves tens of thousands of voice queries every day in a major music streaming platform. It is worth noticing that the proposed methodology, while defined and tested specifically for the music domain, is generic enough to be applied to other voice search applications that face similar challenges, like e.g. Video On-Demand (Rao et al., 2018) or online shopping.

2 Training Data Creation Methodology

Starting with large amounts of unlabeled voice query transcripts,[2] we automatically label selected terms with respect to both a set of music entities (i.e., artists, albums and tracks) and to attributes from an in-house taxonomy of musical attributes (e.g., genres, instruments, moods, etc.). Some of these annotated queries are then discarded, while the remainder are selected for training purposes (see Section 3). This methodology requires the following basic components:

- A large set of unlabeled queries (in the scale of 100k+).
- A large catalog of entities (10M+).
- A taxonomy of attributes (1k+) classified into semantic categories.

There are two main steps to our methodology. First, a heuristic labeling function makes use of corpus statistics and string matching rules to fully-automatically label queries, while discarding some queries whose annotations cannot be established with sufficient confidence. Then, query patterns are extracted from this first set of labeled queries, and leveraged in a human filtering task where erroneously-labeled queries are discarded.

2.1 Heuristic labeling

2.1.1 Categorizing Entities

A pre-processing step of the catalog of entities is necessary before processing the queries. Indeed, working with very large catalogs of entities implies potential ambiguities between entity surface forms and common natural language phrases or even attributes from the taxonomy. For example, in tens of millions of tracks, as those in our catalog, we can find almost any word or expression as a track name (see Table 1). Simple string matching cannot disambiguate whether a user is asking for a specific track, or saying anything else.

Our approach to tackle this issue is to separate entities into three distinct subsets: the *safe-set*, *ignore-set* and *unsure-set*, illustrated in Table 1. The first subset is for entities for which we have high confidence that, when appearing in a query, the user is in fact referring to the entity, regardless of the context (i.e, the other words present in the query). In opposition, the second subset is for entities for which we have high confidence that the user is in fact not asking for that specific entity, but saying something else. Finally, the third subset is for entities where our confidence to assess any of the two previous statements is low.

To decide on the subset of a given entity, we define the concepts of *corpus frequency* of an entity e as the number of times its surface form appears in the corpus of unlabeled queries, and *intrinsic*

[2] In this paper we do not deal with the aspect of Automatic Speech Recognition (ASR), i.e. transcribing voice audio signals to text. The terms "query" and "query transcript" are used interchangeably.

popularity of an entity as the number of times this entity has been interacted with in our product (e.g., by considering number of streams, their Click Through Rates, or through any other notion of popularity relevant to the product at hand). We empirically observed that whenever an entity has a very high corpus frequency but very low intrinsic popularity, it is highly likely that the user is not referring to the entity in their query, even if there is a perfect string matching between the surface form of the entity and a text span in the query. This observation led to the definition of simple rules for entity categorization, making use of the following concepts:

- *Frequency-popularity ratio*: is computed as follows:

$$r(e) = \frac{popularityRank(e)}{frequencyRank(e)} \quad (1)$$

  where $frequencyRank(e)$ is the ranking of entity e with respect to its corpus frequency,[3] and $popularityRank(e)$ is instead its rank with respect to its intrinsic popularity. We compute $r(e)$ for all entities in the catalog, and then normalize to the $[0, 1]$ range. Values close to 1 will reflect cases where the entity is very frequent in queries but not interacted very much with in our product.
- *Attribute overlap*: Given T the set of all attributes in the taxonomy we say that an entity e has an overlap with T if *every token* in the surface form of e pertains to T. For example, the entity "Spanish House" has attribute overlap, because "Spanish" and "House" are both attributes in our taxonomy.

We then use simple rules based on two thresholds τ and ϵ, with $\tau > \epsilon$, to assign entities to either the safe-set, ignore-set or unsure-set. Given the ratio $r(e)$ of an entity e:

- If $r(e) \geq \tau$, e is added to the ignore-set. This is likely a mismatch with a natural language phrase or an attribute.
- If $\tau > r(e) \geq \epsilon$, e is added to the ignore-set or to the unsure-set, depending on their attribute overlap: If there is attribute overlap, it is added to the ignore-set, as this is likely a mismatch with an attribute; otherwise they go to the unsure-set.

[3]Higher frequency means higher rank.

Entity	$r(e)$	T overlap	Category
Could You	0.99	no	ignore-set
Play Music	0.99	no	ignore-set
Xmas	0.99	yes	ignore-set
You Did Something	0.94	no	unsure-set
Country Joe	0.94	no	unsure-set
Acoustic Piano	0.92	yes	ignore-set
Little Snowflake	0.84	no	safe-set
Spanish House	0.47	yes	unsure-set
I am a Human	0.37	no	safe-list

Table 1: Illustration of ambiguities between entities, natural language sentences and taxonomy attributes: Examples of actual entities (music tracks, albums or artists) as found in our catalog, their $r(e)$ ratio, whether they overlap with our taxonomy C, and their corresponding category. The categorization was performed using $\tau = 0.99$, $\epsilon = 0.90$.

- If $r(e) < \epsilon$, e is added to the unsure-set or to the safe-set, depending also on their attribute overlap: If there is attribute overlap they go to the unsure-set; otherwise they go to the safe-set.

Table 1 shows some actual examples of entities from our catalog, with the corresponding ratio, attribute overlap and the resulting category. For instance, *Acoustic Piano* is the name of a rather unpopular album in our catalog which frequently appears in user queries. Since it completely overlaps with the attributes "acoustic" and "piano" of our taxonomy and its frequency-popularity ratio is between τ and ϵ, it is added the ignore-set.

2.1.2 Labeling Function

Once we have categorized all entities in the catalog into the aforementioned three sets, we use this information for disambiguation purposes in the heuristic labeling process. Given a query, we extract all n-grams and look for all the possible matches with the entities within the union of the safe and the unsure sets, and select the longest non-overlapping matches. Then, we apply the following rules:

- If *any* of the matched entities was categorized in the unsure-set, we discard the whole query;
- Otherwise, the matched n-grams in the query are labeled as the corresponding entity types of the matched entities from the safe-set (e.g., artist, album, track). In case of multiple matches (e.g., an artist and a song having the same name) we pick the entity type with the highest intrinsic popularity.

All entities classified in the ignore-set are simply ignored in this process. After the labeling of entities in the query, we look for matches in the list of attributes from the taxonomy among the words that were left unlabeled. The number of musical attributes in the taxonomy is orders of magnitude smaller than the number of entities, and the probability of a confusion between an attribute and a natural language phrase is very low, so we choose to rely on simple string matching to label the attributes, once the entities are labeled in the query.

To illustrate this process, consider the query "could you play the xmas song little snowflake". Our method finds three matches with the catalog of entities: "could you", "xmas" and "little snowflake". The first two matches belong to the ignore-set and the third one belongs to the safe-set (see Table 1). Since no matched entity is classified in the unsure-set, the query is not discarded. The words in the query corresponding to the entity in the safe-set "little snowflake" are labeled as an entity (specifically a music track); the word "xmas" belongs to our taxonomy (under the "theme" category) and, since it does not overlap with any entity annotation, is labeled as an attribute (specifically a theme), see the final annotation in Figure 1.

2.2 Pattern Filtering via Human Curation

2.2.1 Pattern Extraction

For each heuristically labeled query, we extract the corresponding pattern by substituting all the attributes and entities in the query with a placeholder that is assigned to its corresponding class. For example, the pattern corresponding to the query in Figure 1 is "could you play the [theme] song [track]", being [theme] and [track] the placeholders of any theme attribute and any track name. Following this process, we first extract the patterns of all queries annotated by the heuristic labeling stage, and then group queries belonging to the same pattern. For example, the query "could you play the Halloween song I want candy" belongs to the same pattern of the query in Figure 1.

2.2.2 Human Filtering

After pattern extraction, a filtering process is applied, as follows. Given the set of extracted patterns, we identify the vocabulary of words present in those patterns, and compute their respective frequency in the pattern corpus. This list of words is presented to a human curator who –starting from the most frequent words to the less frequent ones– must identify words that should *not* be part of a pattern, and discard them from the vocabulary. Optionally, if the vocabulary of words is very large, the curator can also select a frequency threshold below which all words are discarded.

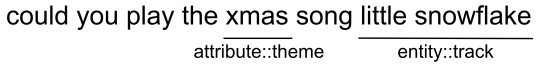

Figure 1: Example query

From the full set of extracted patterns, we then keep only those patterns for which *all* words are included in the cleaned-up vocabulary, and discard the remaining patterns. Finally, all queries from the remaining patterns will form the final set of filtered queries.

This process helps us to avoid labeling errors made by the heuristic labeling step, which can be caused either by limitations of the method itself, inconsistent queries made by users, ASR errors, multilingual queries or entities not present in our catalog. Removing queries with wrong labels is fundamental to avoid noisy patterns and have a clean training dataset. Take for example the query "can you play la modelo by osona." The extracted pattern is "can you play [track] by osona". The word "osona" is not in our catalog of entities (therefore not labeled as an artist). It also very rarely appears in the patterns. This word, and respective pattern, are therefore discarded. Note that artist "Ozuna" is in our catalog of entities. In this particular example, an error was probably introduced by the ASR system.

3 Experimental Setup

We evaluate different quantities of weakly-supervised labeled queries as training data for slot tagging models, from a few thousands up to millions, on a sample of actual voice traffic collected from our application. With the goal of showing the potential of weakly-annotated training data, we compare it to models trained on a dataset of human-annotated queries and different synthetically generated datasets. All datasets are described in Table 2.

3.1 Manual and Synthetic Baselines

We asked two expert human annotators to annotate a set of 7000 randomly selected queries from our logs. They also had to discard all nonsense queries from the original set (e.g., incomplete queries, unrelated queries, incomprehensible ASR transcriptions). We kept only the non-discarded queries having complete agreement between annotators.

Eventually, we obtained a set of 5000 manually annotated queries. We used 70% of those for the training of a slot tagging model which serves as our first baseline (MAN). Then, we used 10% as our validation set in all the approaches and baselines. The remaining 20% (i.e., the test set) was used to evaluate the performance of all trained models. We used Sentence Error Rate (SER), i.e. the ratio of queries with at least one slot classification error over all queries, as our evaluation metric. For space reasons we do not report slot-level metrics such F1 score, which showed strong correlation with SER in our experiments.

We generated 4 additional baselines using synthetic query generation to enhance the training set (SYN). Synthetic queries have shown useful for training slot tagging models in low resources scenarios (Goyal et al., 2018), providing the possibility of introducing novel patterns, attributes or entities in the training data. For our experiments, we started from the patterns extracted using the procedure described in Section 2.2.1 over the manual annotated queries from the training set. Every pattern is filled several times using the entities and attributes available in our catalog and in the taxonomy.[4] We generated 4 different synthetic datasets of different sizes, called respectively SYN_S, SYN_M, SYN_L and SYN_{XL} in Table 2.

3.2 Weakly-Supervised Datasets

We compared the baseline annotations against four different weakly-supervised training sets generated using our methodology. We applied the heuristic labeling function (Section 2.1.2) on four random samples from our query logs, respectively containing 100k, 1M, 10M and 100M unlabeled queries. The threshold hyper-parameters τ and ϵ were tuned on the validation set. These datasets are respectively called WS_S, WS_M, WS_L and WS_{XL} in Table 2 and have the same size of the synthetic datasets described in the previous section.

Finally, we generated the WS(F) dataset with human filtering (see Section 2.2) on the same dataset with 100M unlabeled queries used to generate WS_{XL}. Notice that, because of human filtering, the resulting dataset WS(F) has a smaller number of queries than WS_L.

[4]Entities and attributes were sampled proportionally to their intrinsic popularity.

3.3 Model architecture

In order to provide a fair comparison between the several procedures to generate training data, we trained the *same* architecture using the *same* procedure for all datasets.

We used a single layer BiLSTM-CRF network with 100 hidden units and pre-trained word embeddings of size 250 (Reimers and Gurevych, 2017). We used a concatenation of FastText (Mikolov et al., 2018) and Word2Vec (Mikolov et al., 2013) word embeddings trained on an internal corpus of artist biographies. The word embeddings were kept fixed during training to reduce the risk of undesired semantic shifts.

Each instance of the model was trained using ADAM (Kingma and Ba, 2015) with batch size 100 and dropout 0.2 for a maximum of 100k iterations. The initial learning rate was set to 0.001 and damped by factor 0.5 every 7.5k iterations. We used early-stopping to terminate the training whenever the SER on the validation set did not improve for at least 2.5k consecutive iterations. The experimentation was run using TensorFlow (Abadi et al., 2016) and follows closely that of (Reimers and Gurevych, 2017).

4 Results and Discussion

The test set presents a representative sample of the queries that are effectively asked by users using our voice assistant and hence provide valuable insights of how the system will likely perform online. The SER obtained by the BiLSTM-CRF model trained on each of the training datasets is shown in Table 2.

We can immediately notice that the MAN baseline is outperformed by all the SYN, WS and WS(F) variants that we tested. The improvement brought by weak supervision over the baseline grows noticeably with size. When training using small datasets, SYN_S outperforms MAN and WS_S. However, the SER of models trained on SYN data *increases* with training set size. We hypothesize this behavior to be due to an amplification of the bias coming from the relatively small amount of patterns used in data generation. Indeed, in our experimentation setup, we are focusing on a relatively small amount of human-labeled queries. One way of overcoming this issue would be to add more hand-curated queries or patterns –with the corresponding implications in terms of cost and time. Further, research should be also be dedicated to evaluate different ways to generate synthetic data.

Training Dataset	Description	Label	# patterns	# words	# queries	Test SER
MAN	*manually annotated queries*	*MAN*	-	-	*5k*	*31.64*
SYN	*synthetically generated queries*	*SYN$_S$*	*1k*	*630*	*33k*	*25.11*
		SYN$_M$	*1k*	*630*	*180k*	*26.75*
		SYN$_L$	*1k*	*630*	*875k*	*29.65*
		SYN$_{XL}$	*1k*	*630*	*3.5M*	*29.19*
WS	weakly-supervised annotation	WS$_S$	8k	732	33k	29.83
		WS$_M$	38k	2,105	180k	24.93
		WS$_L$	180k	8,785	875k	23.03
		WS$_{XL}$	805k	33,984	3.5M	21.94
WS(F)	weakly-supervised annotation + human filtering	WS(F)	101k	468	2.5M	**13.58**

Table 2: Different configurations of training datasets used in the experiments and respective performances as evaluated by SER on the test set (smaller is better). We report the numbers of distinct patterns and words in the patterns for the synthetic and weakly-supervised datasets, and the total number of training queries for every dataset. Baselines are in *italic*. The best test SER overall is highlighted in **bold**.

In opposition, the weakly-supervised WS datasets are able to feed the network with a set of query patterns and annotations whose variety naturally grows with size. This characteristic turns out to be extremely beneficial, as evidenced by the incremental reduction in SER with dataset size. When considering the largest datasets, the models trained on the WS$_{XL}$ dataset show a reduction in SER of 9.7 w.r.t. the MAN baseline, 7.25 w.r.t. the SYN$_{XL}$ dataset of the same size and of 3.17 w.r.t. the best SYN dataset (SYN$_S$).

We can however notice that the reduction in SER slows down as size grows. One possible explanation can be that the inherent noise in the data. As shown in Table 2, the size of the vocabulary of pattern words drastically increases with the query sample size, implying more possibilities of having wrongly annotated training queries. Once convergence level is reached, manual curation is needed to further improve the peformance of the model. The WS(F) dataset showcases the improvements that can be achieved enhancing the WS$_{XL}$ dataset with a reasonable manual curation effort, removing noise from the vocabulary of pattern words, and therefore, from the training data. This simple but effective curation step results in a reduction of 8.36 in SER w.r.t. the model trained on WS$_{XL}$ dataset and a reduction of 18.06 w.r.t. the MAN baseline.

Moreover, the curation task followed to create WS(F) is much faster than query annotation, not only because the number of words to review is smaller than the number of raw queries, but also because the task itself is easier. We measured that an expert trained annotator is able to manually annotate approximately 100 queries per hour, whereas the pattern vocabulary can be curated following our methodology in less than 4 hours. Therefore, annotating 7000 queries by two annotators took approximately 140 hours, which is orders of magnitude more than the time needed for the vocabulary curation proposed here.

5 Summary and Future Work

This paper presents a methodology to create training data for training slot tagging models inspired by weak supervision. The methodology consists of two steps. First, a simple heuristic query labeling process is applied, that leverages corpus statistics obtained from query logs and comparing them with entity popularity metrics. Second, a pattern extraction and filtering process is applied to the labeled queries, that makes use of human curation.

Our experimental evaluation clearly shows the value of weak supervision for building training datasets to bootstrap slot tagging models. We show that training with large amounts of weakly-supervised data generated from unlabeled voice queries using the proposed methodology outperforms smaller yet reasonable amounts of hand-annotated data. It also outperforms training with large amounts of synthetic data generated from the same hand-annotated data. We showed that the proposed methodology can be combined with a much less time-consuming word vocabulary curation task with very significant reduction in the end model Sentence Error Rate.

Future work should include experimenting with other manual curation tasks, such as manual cleaning of remaining patterns after vocabulary curation, which could help to increase even more model accu-

racy with a task still much faster than manual annotation of queries. In addition, further study should focus on how synthetic generated queries can complement weakly-supervised datasets, which may help not only in terms of accuracy, but also to assess the quality of labeling of less frequently queried entities or attributes. Moreover, other model architectures could be experimented with, such as pre-trained language models. Finally, experiments could evaluate the applicability of our weakly-supervised methodology to other domains with very large catalogs of entities.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pages 265–283.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. BERT for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Anuj Kumar Goyal, Angeliki Metallinou, and Spyros Matsoukas. 2018. Fast and scalable expansion of natural language understanding functionality for intelligent agents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 145–152, New Orleans - Louisiana. Association for Computational Linguistics.

Ido Guy. 2018. The characteristics of voice search: Comparing spoken with typed-in mobile web search queries. *ACM Trans. Inf. Syst.*, 36(3):30:1–30:28.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Pierre Lison, Aliaksandr Hubin, Jeremy Barnes, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. *arXiv preprint arXiv:2004.14723*.

Neil Mallinar, Abhishek Shah, Rajendra Ugrani, Ayush Gupta, Manikandan Gurusankar, Tin Kam Ho, Q Vera Liao, Yunfeng Zhang, Rachel KE Bellamy, Robert Yates, et al. 2019. Bootstrapping conversational agents with weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9528–9533.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011.

Vito Claudio Ostuni. 2019. "Just Play Something Awesome": The personalization powering voice interactions at Pandora. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 523, New York, NY, USA. Association for Computing Machinery.

Jinfeng Rao, Ferhan Ture, and Jimmy Lin. 2018. Multi-task learning with neural networks for voice query understanding on an entertainment platform. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '18, page 636–645, New York, NY, USA. Association for Computing Machinery.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282.

Nils Reimers and Iryna Gurevych. 2017. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *CoRR*, abs/1707.06799.

Mihai Surdeanu, Sonal Gupta, John Bauer, David McClosky, Angel X Chang, Valentin I Spitkovsky, and Christopher D Manning. 2011. Stanford's distantly-supervised slot-filling system.

Sergey Volokhin and Eugene Agichtein. 2018. Towards intent-aware contextual music recommendation: Initial experiments. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1045–1048.