"Are you calling for the vaporizer you ordered?" Combining Search and Prediction to Identify Orders in Contact Centers

Abinaya K
Flipkart Data Science
Bangalore, India
abinaya.k@flipkart.com

Shourya Roy
Flipkart Data Science
Bangalore, India
shourya.roy@flipkart.com

Abstract

With the growing footprint of ecommerce worldwide, the role of contact center is becoming increasingly crucial for customer satisfaction. To effectively handle scale and manage operational cost, automation through chat-bots and voice-bots are getting rapidly adopted. With customers having multiple, often long list of active orders - the first task of a voice-bot is to identify which one they are calling about. Towards solving this problem which we refer to as *order identification*, we propose a two-staged real-time technique by combining search and prediction in a sequential manner. In the first stage, analogous to retrieval-based question-answering, a fuzzy search technique uses customized lexical and phonetic similarity measures on noisy transcripts of calls to retrieve the order of interest. The coverage of fuzzy search is limited by no or limited response from customers to voice prompts. Hence, in the second stage, a predictive solution that predicts the most likely order a customer is calling about based on certain features of orders is introduced. We compare with multiple relevant techniques based on word embeddings as well as ecommerce product search to show that the proposed approach provides the best performance with 64% coverage and 87% accuracy on a large real-life data-set. A system based on the proposed technique is also deployed in production for a fraction of calls landing in the contact center of a large ecommerce provider; providing real evidence of operational benefits as well as increased customer delight.

1 Introduction

With increasing penetration of ecommerce, reliance on and importance of contact centers is increasing. While emails and automated chat-bots are gaining popularity, voice continues to be the overwhelming preferred communication medium leading to millions of phone calls landing at contact centers. Handling such high volume of calls by human agents leads to hiring and maintaining a large employee base. Additionally, managing periodic peaks (owing to sale periods, festive seasons etc.) as well as hiring, training, monitoring make the entire process a demanding operation. To address these challenges as well as piggybacking on recent progress of NLP and Dialog Systems research, voice-bots are gaining popularity. Voice-bot is a common name of automated dialog systems built to conduct task-oriented conversations with callers. They are placed as the first line of response to address customer concerns and only on failure, the calls are transferred to human agents. Goodness of voice-bots, measured by automation rate, is proportional to the fraction of calls it can handle successfully end-to-end.

Customers' contacts in ecommerce domain are broadly about two types viz. for general enquiry about products before making a purchase and post purchase issue resolution; with overwhelming majority of contacts are of the latter type. For post purchase contacts, one of the first information that a voice-bot needs to gather is which product the customer is calling about. The most common practice has been to enumerate all products she has purchased, say in a reverse chronological order, and asking her to respond with her choice by pressing a numeric key. This is limiting in two important ways. Firstly, it limits the scope to a maximum of ten products which is insufficient in a large fraction of cases. Secondly and more importantly, listening to a long announcement of product titles to select one is a time-consuming and tedious customer experience.

In this paper, we introduce the problem of *order identification* and propose a technique to identify or predict the product of interest for which a cus-

Proceedings of the 4th Workshop on e-Commerce and NLP (ECNLP 4), pages 58–69

Fuzzy Search		Predictive Model	
Customer Utterance	**Product titles of active orders with fuzzy search match**	**Product titles of active orders with top-1 match from predictive model**	**Top-k features for the prediction**
`maine order kiya tha to toner 25 ke dikha to uska`	[Aiwa Professional 102 //00 High Quality 40 PCS Socket Set, Hidelink Men Brown Genuine Leather Wallet, CEDO XPRO Edge To Edge Tempered Glass for Realme XT, Realme X2, **Protoner 25 kg PVC weight with 4 rods and Flat bench Home Gym Combo**]	[Sonata 77085PP02 Volt Analog Watch - For Men, Oxhox HBS-730 Wireless compatible with 4G redmi Headset with Mic Bluetooth Headset, **MyTech With Charger M3 Smart Band Fitness Band**]	*number of days since return initiation, Selling Price, is incident created in last 2 days?*
`i can 2000 green color mobile phone`	[Surat Dream Portable Mini Sewing Machine Handheld Handy Stitch Machine Manual Cordless Electric Stitching Machine Electric Sewing Machine, **I Kall K1000 (Green, 64 GB),** I Kall K1000 (Blue, 64 GB)]	[**Whirlpool 7.5 kg 5 Star, Hard Water wash Fully Automatic Top Load Grey**, Asian Running Shoes For Women]	*days since incident creation, days since last chat, rank wrt selling price*
`blue 2 dead phone ke liye`	[Hardys Full Sleeve Solid Men Jacket, Brawo Party Wear Party Wear For Men, **T GOOD Lite SH12 Bluetooth Headset, T GOOD Lite SH12 Bluetooth Headset,** SPINOZA Pink diamond studded attractive butterfly stylish women Analog Watch - For Girls]	[Ncert Chemistry Class 12 (Part 1 And 2) Combo 2 Book (K.C.G), **STROM COLLECTION Men Formal Black Genuine Leather Belt**, OPPO F15 (Blazing Blue, 128 GB)]	*is cancelled?, is incident created in last 2 days?, number of days since return initiation*

Table 1: Examples of top matches from fuzzy search and predictive model. The first column shows transcribed customer utterances and the second column shows all active orders at the time of the call with the top match from fuzzy search emphasized. The examples under Predictive Model shows the most likely order at the time of the call along with top-k features leading to the prediction.

tomer has contacted the contact center[1]. We do it in a natural and efficient manner based on minimal or no explicit additional input from the customer through a novel combination of two complementary approaches viz. **search** and **prediction**. The system is not restricted by the number of products purchased even over a long time period. It has been shown to be highly accurate with 87% accuracy and over 65% coverage in a real-life and noisy environment.

After customer verification, a question was introduced in the voice-bot flow "`Which product you are calling about?`". Her response was recorded and transcribed by an automatic speech recognition (ASR) system to text in real-time. We modeled the **search** problem as a task to retrieve the most matching product considering this response as a query over the search space of all active products represented as a set of product attributes e.g. title, description, brand, color, author etc. While simple in formulation, the task offers a few practical challenges. Customers do not describe their products in a standard manner or as it is described in the product catalog. For example, to describe a *"SAMSUNG Galaxy F41 (Fusion Blue, 128 GB) (6 GB RAM)"* phone, they may say *F41*, *Galaxy F41*, *mobile*, *phone*, *mobile phone*, *cellphone*, *Samsung mobile*, etc. (more examples can be seen in Table1). Secondly, the responses from customers varied widely from being heavily code-mixed to having only fillers (*ummm*s, *aah*s, *what*s etc.) to blank responses. This is complemented owing to the background noise as well as imperfections in ASR systems. Finally, in a not so uncommon scenario, often customers' active orders include multiple instances of the same product, minor variations thereof (e.g. in color), or related products which share many attributes (e.g. *charger* for *"SAMSUNG Galaxy F41 (Fusion Blue, 128 GB) (6 GB RAM)"*) which are indistinguishable from their response to the prompt.

We propose an unsupervised n-gram based *fuzzy search* based on a round of pre-processing followed by custom lexical and phonetic similarity metrics.

[1]We use the terms order and product interchangably to mean different things customers have purchased.

In spite of its simplicity, this solution achieves 32% coverage with an accuracy of 87%, leveraging the relatively small search space. The custom nature of this solution achieves much higher accuracy compared to more sophisticated general purpose product search available on ecommerce mobile apps and websites. This simple technique also does not require additional steps such as named entity recognition (NER) which has been used for product identification related work in literature (Wen et al., 2019). Additionally, NER systems' performance are comparatively poor on ASR transcripts owing to high degree of recognition and lexical noise (e.g. missing capitalization etc) (Yadav et al., 2020).

While fuzzy search works with high accuracy, its coverage is limited owing to various mentioned noise in the data. We observed that about 25% of callers did not answer when asked to identify the product they were calling about. To overcome this challenge, we introduced a complementary solution based on *predictive modeling* which does not require explicit utterances from customers. In simple words, the model creates a ranking of active orders on the basis of likelihood of a customer calling about them. This is based on the intuition that certain characteristics of orders make them more likely to call about e.g. a return, an orders which was supposed to be delivered on the day of calling etc. Based on such features of orders and customer profile, a random forest model gives prediction accuracy of 72%, 88% and 94% at top-1, 2, and 3. For high confidence predictions, the voice-bot's prompt is changed to "`Are you calling for the <PRODUCT-NAME> you ordered?`" For right predictions, not only it reduces the duration of the call, also increases customer delight by the personalized experience. In combination, fuzzy search and predictive model cover 64.70% of all voice-bot calls with an accuracy of 87.18%.

Organization of the paper: The rest of the paper is organized as follows. Section 2 narrates the background of order identification for voice-bot, sections 3 discusses the proposed approach and sections 4 and 5 discuss the datasets used in our study and experiments respectively. Section 6 briefs some of the literature related to our work before concluding in section 7.

2 Background

A typical call flow of the voice-bot would start with greeting followed by identity verification, order identification and confirmation to issue identification and finally issue resolution or transfer to human agent if needed. Figure 1 shows a sample conversation between a bot and a customer with multiple active orders, where the customer is asked to identify the order she called for.

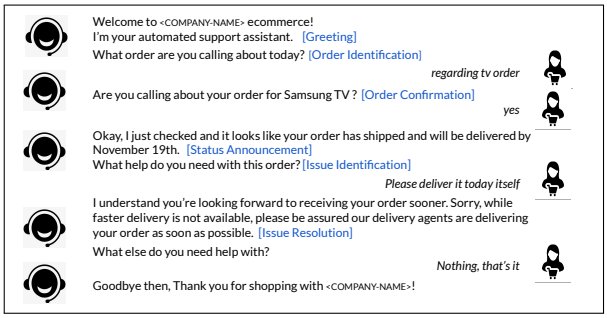

Figure 1: Sample conversation between voice-bot and the customer.

Customers' responses to this question are transcribed to text in real-time using an ASR model. There were some practical difficulties in identifying the corresponding order from the transcribed customer utterances. When asked to identify the order, customers were not sure what they had to talk about, resulting in generic responses like *'hello', 'hai', 'okay'*, etc. in around 23% of the calls. Some customers straightaway mentioned about the issue instead of describing the product - for eg., *'refund order', 'order return karne ke liye call kiya hai', 'mix match pick up cancel'*, etc. We also noticed a prevalence of blank transcripts in around 22% of the calls, mostly from customers who have interacted with the voice-bot in the past. We believe this is due to the change in the call flow of voice-bot from what they have experienced in the past. Another major challenge comes from the ASR errors especially in the context of code-mixed utterances. The transcription noise especially on product tokens (*'mam record'* for *'memory card'*, *'both tropage'* for *'boAt Rockerz'*) made it more difficult to identify the right order. Also by nature of ASR, various lexical signals like capitalization, punctuation are absent in the ASR transcribed text, thereby making the task of order search more challenging.

After the order is identified or predicted, the customer is asked to confirm the chosen order. The customer can confirm positively and continue the conversation with the voice-bot or respond negatively and fallback to human agents. The ideal expectation from order search is to return a single matching order but in cases where multiple similar products exist, the voice-bot may prompt again to help disambiguate.

3 Proposed Approach

We propose to solve the problem of order identification through two steps. In the first phase (fuzzy search), we model the problem as a retrieval-based question-answering task where customer utterance is the query and the set of active orders of the customer is the search space. Towards solving this, we employ a sequence of matching techniques leveraging lexical and phonetic similarities. In the second phase (order prediction), we build a supervised learning model to predict the likelihood of the customer calling regarding different active orders. This later phase does not depend on customer utterances and hence does not get affected by transcription inaccuracies. Both of these approaches are discussed in detail in this section.

3.1 Fuzzy Search

Given the customer utterance and the product attributes of the active orders, fuzzy search proceeds in multiple rounds of textual similarity measures like direct, partial and phonetic match to retrieve the corresponding order, customer called for. These stages are invoked sequentially until a matching order is found. Sequentiality is introduced in fuzzy search in order to maximize the coverage while keeping the false positives low. Various stages involved in fuzzy search is shown in appendix A. Various stages in fuzzy search are detailed below. We use the terms {*token* and *word*} and {*utterance* and *query*} interchangeably.

3.1.1 Pre-processing

We observed prevalence of generic texts like *hello, haan, ok* in the customer utterances, which are of no use in retrieving the order. Hence, such commonly used tokens are to be removed from the query. Also, by nature of ASR, acronyms are transcribed as single letter split words for eg., *a c* for *AC*, *t v* for *TV*, etc. We followed the pre-processing steps as below.

- Removal of generic tokens: The commonly used tokens are identified by taking top 5% of frequently spoken tokens and are manually examined to ensure no product specific terms are removed.
- Handle split words: The split words are handled by joining continuous single letter words.

After these pre-processing steps, some of the customer utterances containing only the generic tokens would become blank, such cases are considered to have no match from the active orders. For non blank processed queries, we use the following similarity measures to identify the matching order(s).

Let q denote the pre-processed customer query composed of query tokens. Let $\{p_i\}_{i=1}^{P}$ denotes list of active orders where p_i denote the product title corresponding to i^{th} order. Product titles are typically concatenation of brand, model name, color, etc- *'Redmi Note 9 Pro (Aurora Blue, 128 GB) (4 GB RAM)'*, *'Lakme Eyeconic Kajal (Royal Blue, 0.35 g)'* are some sample product titles.

3.1.2 Direct Match

The objective of direct match is to handle relatively easier queries, where customer utterance contains product information and is transcribed without any noise. Direct Match looks for exact text matches between query tokens and the tokens of product title. Each product is assigned a score basis the fraction of query tokens that matches with tokens from the corresponding product title. Score for the i^{th} product is obtained as

$$s_i = \frac{1}{|q|} \sum_{x:q} \mathbb{1}_{\{y:y \in p_i, y==x\}!=\phi}$$

where $\mathbb{1}_x$ indicates the indicator function which is 1 if x is true else 0. Product(s) with the maximum score are considered the possible candidate products for a direct match. Direct match between the query and any of the products is said to occur in the following cases.

- Score of 1 would indicate that the product title includes all query tokens. Hence if the maximum score is 1, all product(s) with the score of 1 are returned by direct match.
- If the max score is less than 1, direct match is limited to single candidate retrieval so as to avoid false positive due to similar products in the active orders.

3.1.3 Partial Match

In order to handle partial utterances of product names by the customers and to account for ASR errors in product specific terms of customer utterances, partial match is introduced. For example, partial product utterances like '*fridge*' for '*refrigerator*", '*watch*' for '*smart watch*' and ASR misspelled utterances like '*sandel*' for '*sandal*', would be handled by partial match. Algorithm 1 elucidates various steps in partial matching. It uses partial similarity (Sim) between the n-grams from query and the product titles. We start with individual tokens and then move to bigram, trigram, etc

till 4-gram. Match at a higher n is clearly more promising than a lower n. For eg, a customer could ask for 'JBL wired headset' and the active orders could include '*boAt Wired Headset*' and a '*JBL Wired Headset*'. In such cases, token similarity or bigram similarity might treat both of these headsets as matching orders, however trigram similarity would result in a correct match. i.e., for cases with similar products in the active orders, going for a higher n would help reduce the false positives if customer had specified additional details to narrow the order.

Algorithm 1: n-gram based partial match

Result: R
$R = \phi, Q_0^{'} = q$
for n = {*1,2,3,4*} **do**
  $Q_n = ngrams^{'}(q, n, Q_{n-1}^{'})$
  $Q_n^{'} = \phi$
  for *i=1:P* **do**
    $P_{in} = ngrams(p_i, n)$
    $s_i = \frac{1}{|Q_n|} \sum_{x:Q_n} \mathbb{1}_{\{y:y \in P_{in}, Sim(x,y) > \theta\}! = \phi}$
    $Q_n^{'} = Q_n^{'} \cup \{x \in Q_n : \{y \in P_{in} : Sim(x,y) > \theta\}! = \phi\}$
  $\hat{p} = \{p_i : s_i == max(s_i), max(s_i)! = 0\}$
  if $|\hat{p}| == 1$ *or* $max(s_i) == 1$ **then**
    $R = \hat{p}$

Let $Q_n^{'}$ refer to the n-grams in query string that had a match with any of the product n-grams, $ngrams$ represent a function to return all possible n-grams of input string and $ngrams^{'}$ return the surrounding n-grams of $Q_{n-1}^{'}$. For $n \geq 2$, Q_n would only contain n-grams with one or more product tokens. At a particular n, we obtain a similarity score s_i for each active order, based on the proportion of n-grams in Q_n, that finds a match with n-grams in corresponding product titles and the orders with maximum score are considered candidate orders ($\hat{p}$) for successful match. At any n, matching order(s) is said to have found if n-grams from any order finds a match with all n-grams included in Q_n i.e., $max(s_i) == 1$ or when there is only one candidate product i.e., $|\hat{p}| == 1$.

If none of the products finds a match at higher n, the matched products as of level $n-1$ is considered. A threshold on the similarity measure is imposed to indicate whether two n-grams match.

3.1.4 Phonetic Match

ASR errors on product specific tokens imposes additional challenges in identifying the corresponding order. For example, *'in clinics hot 94'* for *'Infinix Hot 9 Pro'*, *'mam record'* for *'memory card'*, '*double back'* for *duffel bag*, etc. To handle such queries, we consider similarity between phonetic representations of n-grams of product title with that of customer utterance. Algorithmically, phonetic match works similar to fuzzy match (as in algorithm 1), with the important difference the similarity score (Sim) is on phonetic representation of n-grams. With this, strings like *'mam record'*, *'memory card'*, *'double back'*, *duffel bag* are mapped to *'MANRACAD'*, *'MANARACAD'*, *'DABLABAC'* and *'DAFALBAG'* respectively. Clearly, the noisy transcribed tokens are much closer to the original product tokens in the phonetic space.

3.2 Order Prediction

The objective of this step is to build a predictive model for ranking of active orders based on the likelihood of why a customer is calling. We formulate it as a classification problem on active orders and learn a binary classifier to predict the likelihood of customer calling for each order.

3.2.1 Feature Engineering:

The features used in the model are broadly categorized into 4 categories, i.e., order specific, transaction specific, product specific and self serve related features.

- **Order-specific** features includes *order status*, *is delivery due today?*, *is pickup pending?*, *Is Refund issued?* , etc. These features are specific to the time when customer calls.
- **Transaction-specific** features include *price of the product*, *shipping charges*, *order payment type*, etc
- **Product-specific** features include product attributes like *brand*, *vertical*, *is this a large item?* , etc. These features are not dependent on the time of the customer call.
- **Self-serve** features like *number of days since last chat conversation*, *number of days since last incident creation*, etc.

It is important to note that the likelihood of a customer calling for an order is highly related to the features of other active orders of the customer. For example, the chances of customer calling for an order that just got shipped are less when there is another order whose refund is pending for a long time. The formulation by default doesn't consider the relationship among features of other active orders. Hence this is overcome by creating derived features that brings in the relative ordering between features of active orders of a customer. Some of derived features include rank of the order with respect to ordered date (customers are more likely to

call for a recent order than older ones), if refund is pending for any other order, if there are existing complaints for other orders etc.

Preprocessing: Together with these derived features, we have a total of 42 features mix of categorical and numerical. Low cardinality features like order status are one hot encoded, high cardinality features like brand, category, etc are label encoded and the numerical features are standard normalized. The labels available in our dataset is at a call level. Since the classification is at an order level, the label is assigned 1 or 0 depending on whether it's the order the customer called for.

3.2.2 Model Details

In order to learn a binary classifier for order prediction, we experiment with various standard machine learning models like Logistic regression, tree based ensemble model like Random Forest and deep learning models like Deep Neural Network (DNN). As a baseline, we compare with the reverse chronological ranking of orders with respect to date of order. Various hyper parameters involved in these models are tuned using grid search. More details on the range of hyper parameters considered and the chosen best hyper parameter is available in appendix B.

4 Dataset

This section discusses details of the datasets used for order search and order prediction experiments.

Search Dataset: In order to collect data for order search experimentation, customers with multiple active orders were asked to describe the product they are calling for. The transcribed customer utterances along with the product attributes of the orders like product title, brand, etc constitute our dataset. We had a small development dataset of about $2.5K$ calls and a large test set of about $95K$ calls. The development set was used to build and tune the *fuzzy search* technique. The performance on both datasets are reported in section 5.1.

Prediction Dataset: The dataset for predictive modeling is collected from the live customer calls. The dataset contains features of active orders and the customer chosen order, which would serve as ground truth. We came up with a variety of features from order, product related ones to self serve related ones and are collected online or offline depending whether the feature is dependent on the time when customer calls. The features for the active orders and the customer chosen order for $150K$ customer calls constitute our prediction dataset. The experiments and results on this dataset is given in section 5.2.

5 Experiments and Results

The performance of an order search algorithm is evaluated using *Coverage* and *Accuracy*. Coverage refers to the fraction of calls, where proposed technique gave a match. Among the cases where match is found, accuracy refers to the fraction of correct matches. The rest of this section discusses the experiments and results on the development and test sets of the search dataset followed by order prediction experiments and finally the performance combining search and prediction for order identification.

5.1 Order Search Results

We compare our approach fuzzy search with the following two approaches viz. manual baseline and ecommerce search. In manual baseline, customer utterances were tagged for product entities by human agents handling those calls to get a baseline on the coverage. Ecommerce search refers to the general purpose product search used by consumers on ecommerce websites and mobile apps. This latter approach relies on NER for detecting product entities, which we had done through a NER model based on conditional random fields (CRF) (Lafferty et al., 2001).

Figure 2 shows the coverage of fuzzy search vs these two approaches on the **development** set of search dataset. As shown, we had a total of 2515 customer utterances, of which human annotators could spot product entities only in 34% of them demonstrating the difficulty of the task. Fuzzy search and ecommerce search had a coverage of 32.1% and 17.2% respectively. Both fuzzy and ecommerce search have an overlap with the remaining 66% data that manual annotations couldn't cover, showing that product entity malformations due to transcription noise is overcome by these models significantly. The coverage of ecommerce search is affected by poor NER performance on noisy ASR transcripts. At this point, an attentive reader may refer back to Table 1 to see some of the sample matches returned by fuzzy search. Some more qualitative results are shown in Appendix-C to understand the gap between fuzzy and ecommerce search further. Clearly, our customized ap-

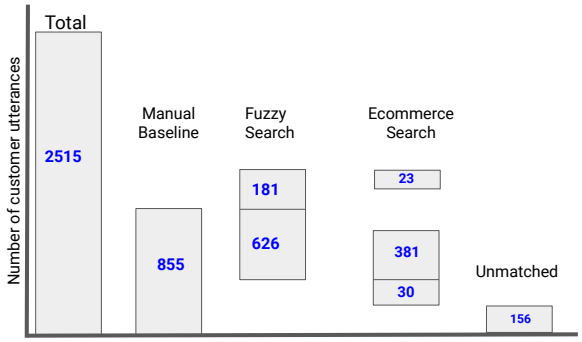

Figure 2: Comparison of coverage of fuzzy search against manual baseline and ecommerce search on development set. Vertical placements are indicative of overlaps between different sets.

proach fuzzy search performs better than ecommerce search.

As a second experiment, we compare fuzzy search against text embedding based approaches - *fasttext based similarity modelling* and *fuzzy search on fasttext*. The former obtains the relevance of a customer utterance to a product by taking cosine similarity between their corresponding sentence embeddings and the most relevant orders(s) with similarity over a threshold is considered. Sentence embeddings are obtained by averaging the word embeddings obtained from a Fasttext model(Bojanowski et al., 2017) trained on customer utterances and product titles. *Fuzzy search on fasttext* combines the benefits of customisations in fuzzy search and the semantics from fasttext, where fuzzy search is done on text embeddings, with textual similarity replaced by cosine similarity over fasttext embeddings of n-grams.

Table 2 shows the coverage and accuracy of single and multiple matches retrieved by various order search approaches on live contact center calls, that constitute the **test** set of search dataset. Fuzzy search is found to perform better with 18.02% single order matches with an accuracy of 86.33%. Similarity modelling on fasttext is found to have lesser coverage and accuracy than fuzzy search. Decrease in accuracy is attributed to calls with multiple similar orders and the retrieval fetches one of them as a match but customer chose a different order during the call. Fuzzy search on fasttext performs on par with fuzzy search on text, showing that semantics captured by word embeddings does not add incremental value. This, we believe, is owing to the lexical nature of product titles and unambiguous context of customer utterances. Fuzzy search despite being unsupervised, experimented on development data, the coverage and accuracy hold good on real life calls as well.

Upon deep diving into the multiple order matches from fuzzy search, we found around 38% of such multiple matches had exact same product (same make and model), 49% of them were same product type - can be different model, color etc (e.g., multiple t-shirts), some of them being similar products (e.g., shampoo, hair conditioner, hair serum of the same brand). Multiple matches, though not acted upon by voice-bot, are still valid matches.

5.2 Order Prediction Results

The performance of order prediction is measured by top-k accuracy(A_k) given by the fraction of calls where the model predicted the ground truth order in top-k predictions. We use Prediction dataset with train/val/test split of 70/10/20 respectively for training order prediction models. Table 3 shows the top-k accuracy of various prediction models. Random Forest, a decision tree based ensemble model is found to perform better at both top-1 and top-2 accuracy of 72.52% and 88.71% respectively and marginally under performing than Deep Neural Network (DNN) at top-3 thereby showing the characteristics of the orders indeed decide the order, customer calls for.

The reader may again refer to Table 1 where the rightmost two columns show some of the sample top-1 predictions and the features that led to such predictions by the Random Forest model. In the first example shown in table 1, among many orders, model predicted *fitness band*, which the customer has already initiated return process and have an existing complaint lodged. Upon looking into the top features that govern the model's predictions, we found self serve features like *chat before call*, *existing complaints before call*, etc. in addition to the *rank wrt ordered date and selling price* to be on top, showing that the customers explore self serve before calling. We show the Shapley Values plot of the feature importance in figure 3. We introduce a threshold on the likelihood of top ranked prediction to improve the accuracy while marginally compromising on coverage. With a threshold of 0.6, top-1 predictions from Random Forest had a coverage and accuracy of 62.5% and 84% respectively.

Both order search and order prediction is also evaluated on an out-of-time data, consisting of $13K$ customer calls. Table 4 shows the coverage and accuracy of order search and order prediction indi-

Approach	Coverage (in %)		Accuracy (in %)	
	Single	Multiple	Single	Multiple
Fuzzy Search	**18.02**	9.62	**86.33**	70.19
Fasttext based Similarity modelling	17.93	4.28	71.03	63.34
Fuzzy Search on fasttext	17.09	**10.47**	85.36	**71.22**

Table 2: Coverage and accuracy of single and multiple matches from order search approaches on the test set

Model	A_1	A_2	A_3
Rev. Chronological	40.09	75.17	87.74
Logistic Regression	71.00	88.00	93.70
Random Forest	**72.52**	**88.71**	94.09
DNN	70.34	88.33	**94.34**

Table 3: Top-k accuracies(%) of order prediction models

Approach	Coverage	Accuracy
Fuzzy Search	20.32	86.83
Order Prediction	58.1	84.46
Search + Prediction	64.7	87.18

Table 4: Performance of Search and Prediction

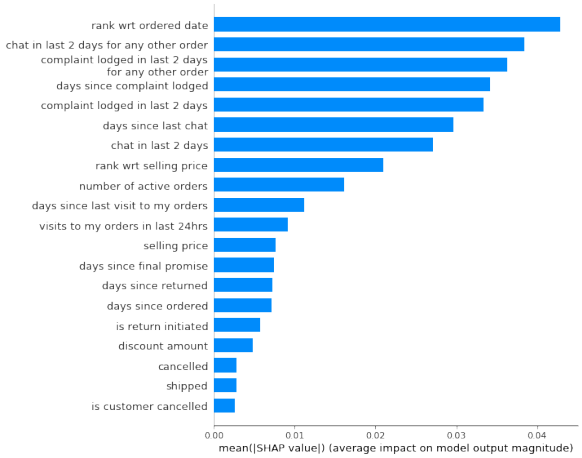

Figure 3: Feature importance of top 20 features for Random Forest model. Features with prefix *'rank'* or suffix *'for any other order'* are derived features introduced to bring relation with other active orders.

vidually and the overall coverage by having both search and prediction in place. Order prediction resulted in an incremental coverage of 44% while maintaining same accuracy.

6 Related work

Order identification has not been much explored in the literature. Most related problem is on NER to identify product entities (Putthividhya and Hu, 2011; Joshi et al., 2015; More, 2016). In the literature, there are many studies focused on NER for product entity extraction ranging from classical techniques (Brody and Elhadad, 2010) to recent deep learning approaches that make use of word embeddings (Majumder et al., 2018; Jiang et al., 2019). While entity extraction from text is well researched in the literature, NER on speech is less studied. Most initial works on speech had a two staged approach - ASR followed by NER (Cohn et al., 2019), recent works directly extract entities from speech (Ghannay et al., 2018; Yadav et al., 2020). While NER helps in ecommerce search on websites and apps, the specific nature of order identification problem and the limited search space of active orders make NER unnecessary.

Another related line of works is on sentence similarity tasks. Owing to the success of word embeddings (Mikolov et al., 2013; Bojanowski et al., 2017), there is a lot of literature on textual similarity related tasks, that make use of word embeddings in a supervised (Yao et al., 2018; Reimers and Gurevych, 2019; Shen et al., 2017) and unsupervised fashion (Arora et al., 2016). (Wieting et al., 2015) showed that simple averaging of word embeddings followed by cosine similarity could provide competitive performance on sentence similarity tasks. We have compared word embeddings based approaches to show that additional semantics does not help in order identification problem.

7 Conclusion

In this paper, we present one of the first studies exploring order identification for ecommerce contact centers. The proposed two-staged fuzzy search and order prediction technique provide 64% coverage at 87% accuracy on a large real-life dataset which are significantly better than manual baseline and relevant comparable techniques. Order prediction though developed for voice-bot, could also be used in other places like chat bot or non bot calls, where we can ask proactively if this is the order customer is looking for help. Finally, going beyond the scientific impact of this work, the proposed solution is also deployed in production for a fraction of calls landing in the contact center of a large ecommerce provider leading to real-life impact.

References

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2016. A simple but tough-to-beat baseline for sentence embeddings.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics*, pages 804–812.

Ido Cohn, Itay Laish, Genady Beryozkin, Gang Li, Izhak Shafran, Idan Szpektor, Tzvika Hartman, Avinatan Hassidim, and Yossi Matias. 2019. Audio de-identification-a new entity recognition task. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 197–204.

Sahar Ghannay, Antoine Caubriere, Yannick Esteve, Antoine Laurent, and Emmanuel Morin. 2018. End-to-end named entity extraction from speech. *arXiv preprint arXiv:1805.12045*.

Yufan Jiang, Chi Hu, Tong Xiao, Chunliang Zhang, and Jingbo Zhu. 2019. Improved differentiable architecture search for language modeling and named entity recognition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3576–3581.

Mahesh Joshi, Ethan Hart, Mirko Vogel, and Jean David Ruvini. 2015. Distributed word representations improve ner for e-commerce. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 160–167.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Bodhisattwa Prasad Majumder, Aditya Subramanian, Abhinandan Krishnan, Shreyansh Gandhi, and Ajinkya More. 2018. Deep recurrent neural networks for product attribute extraction in ecommerce. *arXiv preprint arXiv:1803.11284*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ajinkya More. 2016. Attribute extraction from product titles in ecommerce. *arXiv preprint arXiv:1608.04670*.

Duangmanee Putthividhya and Junling Hu. 2011. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1557–1567.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Yikang Shen, Wenge Rong, Nan Jiang, Baolin Peng, Jie Tang, and Zhang Xiong. 2017. Word embedding based correlation model for question/answer matching. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Musen Wen, Deepak Kumar Vasthimal, Alan Lu, Tian Wang, and Aimin Guo. 2019. Building large-scale deep learning system for entity recognition in e-commerce search. In *Proceedings of the 6th IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, pages 149–154.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*.

Hemant Yadav, Sreyan Ghosh, Yi Yu, and Rajiv Ratn Shah. 2020. End-to-end named entity recognition from english speech. *arXiv preprint arXiv:2005.11184*.

Haipeng Yao, Huiwen Liu, and Peiying Zhang. 2018. A novel sentence similarity model with word embedding based on convolutional neural network. *Concurrency and Computation: Practice and Experience*, 30(23):e4415.

A Flow of Fuzzy Search

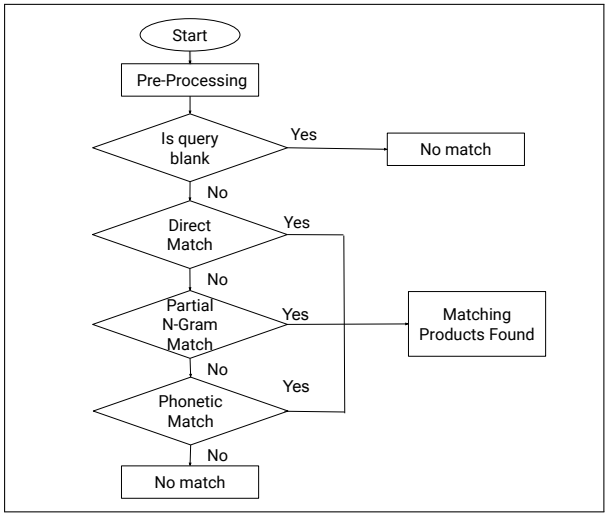

Figure 4: Flow Chart showing the stages involved in fuzzy search

B Hyper-parameter tuning for Order Prediction

Model	Hyper-parameter	Range of values	Best hyper-parameter
Logistic Regression	penalty	l1, l2, elasticnet,none	l2
	tol	[1e-2,1e-6]	1e-5
	C	[1e2, 1e-2]	1
Random forest	n_estimators	[10,1000]	50
	criterion	gini,entropy	gini
	max_depth	10, 20, 50, 100	20
	min_samples_leaf	2,10,50,100	10
	bootstrap	False, True	True
DNN	number of hidden layers	2,3	2
	number of neurons	50,100,200	100
	lr	[1e-2,1e-4]	1e-3
	activation	relu, sigmoid, leaky relu, tanh	leaky relu

Table 5: Range of values for various hyper-parameters and the chosen hyper-parameter with best top-1 accuracy on validation set for various order prediction models

C Sample predictions from fuzzy search & ecommerce search

Customer Utterance	Product titles of active orders	Comments
mi tv ke baare mein	[STAMEN 153 cm (5 ft) Polyester Window Curtain (Pack Of 4), Sauran 26-55 inch Heavy TV Wall Mount for all types of Fixed TV Mount Fixed TV Mount, **Mi 4A 100 cm (40) Full HD LED Smart Android TV**, Leemara Virus Protection, Anti Pollution, Face Mask, Reusable-Washable Outdoor Protection Cotton Safety Mask]	✓Fuzzy Search ✓Ecommerce Search
cover ke baare mein **mobile cover** ke baare mein	[**Aspir Back Cover for Vivo V15, Mobi Elite Back Cover for Vivo V15**, RUNEECH Back Camera Lens Glass Protector for VIVO V 20, Shoes Kingdom Shoes Kingdom _LB791 Mocassins Casual Loafers For Men (Brown) Loafers For Men, **Aspir Back Cover for Vivo V20**, CatBull In-ear Bluetooth Headset]	✓Fuzzy Search ✗Ecommerce Search
datacable ke liye	[Easy Way Fashion Doll with Dresses Makeup and Doll Accessories, **Vrilliance Traders Type C Compatible Fast Data Cable Charging Cable for Type C Android Devices (1.2 M,Black) 1.2 m USB Type C Cable**]	✓Fuzzy Search ✗Ecommerce Search
in clinics hot 94	[JOKIN A1 MULTI FUNCTIONAL SMARTWATCH Smartwatch, **Infinix Hot 9 Pro (Violet, 64 GB)**, Vivo Z1Pro (Sonic Blue, 64 GB), Vivo Z1Pro (Sonic Blue, 64 GB), Vivo Z1Pro (Sonic Blue, 64 GB), Tech Unboxing Led Rechargeable Fan With Torch 120 mm 3 Blade Exhaust Fan]	✓Fuzzy Search ✗Ecommerce Search
chappal ke liye paanch sau saat sattar pe ka product tha mera	[Highlander Full Sleeve Washed Men Jacket, **Oricum Slides**, BOLAX Black Slouchy woolen Long Beanie Cap for Winter skull head Unisex Cap, **Oricum Slides**, BOLAX Black Slouchy woolen Long Beanie Cap for Winter skull head Unisex Cap]	✗Fuzzy Search ✓Ecommerce Search

Table 6: Examples of predictions from fuzzy search and ecommerce search. First column shows the customer utterances along with the NER predictions emphasized. Second column shows all active orders at the time of call, with matching orders emphasized. Last column shows the correctness of order search approaches.