Gaining Insights into Unrecognized User Utterances in Task-Oriented Dialog Systems

Anonymous ACL submission

Abstract

The rapidly growing market demand for dialogue agents capable of goal-oriented behavior has caused many tech-industry leaders to invest considerable efforts into task-oriented dialog systems. The performance and success of these systems is highly dependent on the accuracy of their intent identification – the process of deducing the goal or meaning of the user's request and mapping it to one of the known intents for further processing. Gaining insights into unrecognized utterances – user requests the systems fails to attribute to a known intent – is therefore a key process in continuous improvement of goal-oriented dialog systems.

We present an end-to-end pipeline for processing unrecognized user utterances, including a specifically-tailored clustering algorithm, a novel approach to cluster representative extraction, and cluster naming. We evaluated the proposed clustering algorithm and compared its performance to out-of-the-box SOTA solutions, demonstrating its benefits in the analysis of unrecognized user requests.

1 Introduction

The development of task-oriented dialog systems has gained much attention in both the academic and industrial communities over the past decade. Task-oriented (also referred to as goal-oriented) dialog systems help customers accomplish a task in one or multiple domains (Chen et al., 2017), compared with open-domain dialog systems aimed at maximizing user engagement (Huang et al., 2020). A typical pipeline system architecture is divided into several components, including a natural language understanding (NLU) module. This module is responsible for classifying the first user request into potential *intents*, performing a decisive step that is required to drive the subsequent conversation with the virtual assistant in the right direction.

Goal-oriented dialog systems often fail to recognize the intent of natural language requests due to system errors, incomplete service coverage, or insufficient training (Grudin and Jacques, 2019; Kvale et al., 2019). In practice, these cases are normally identified using intent classifier uncertainty. Here, user utterances that are predicted to have a level of confidence below a certain threshold to any of the predefined intents, are identified and reported as unrecognized or *unhandled*. Figure 1 presents the NLU module from a typical task-oriented dialog system: the user utterance is either transformed into an intent with an appropriate flow of subsequent actions, or labelled as unrecognized and stored in the *unhandled pool*.

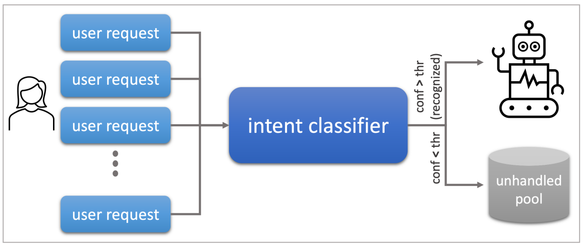

Figure 1: Natural language understanding (NLU) module. Based to the intent classifier's confidence level, first user utterances are 'recognized' and associated with an execution flow, or stored in an unhandled pool.

Unhandled utterances often carry over various aspects of potential importance, including novel examples of existing intents, novel topics that may introduce a new intent, or seasonal topical peaks that should be monitored but not necessarily modeled within the system. In large deployments, the amount of unhandled utterances can reach tens of thousands each day. Despite their evident importance for continuous bot improvement, tools for gaining effective insights into unhandled utterances have not been developed sufficiently, leaving a vast body of knowledge, as well as a range of potential actionable items, unexploited.

Gaining insights into the topical distribution of user utterances can be achieved using unsupervised text analysis tools, such as clustering or topic modeling. Indeed, identifying clusters of semantically similar utterances can help surface topics of interest to a conversation analyst. We show that traditional clustering algorithms result in sub-optimal performance due to the unique traits of unhandled utterances in dialog systems: an unknown number of expected clusters and a very long tail of outliers. Consequently, we propose and evaluate a simple radius-based variant of the k-means clustering algorithm (Lloyd, 1982), that does not require a fixed number of clusters and tolerates outliers gracefully. We demonstrate that it outperforms its out-of-the-box counterparts on a range of datasets.

We further propose an end-to-end process for surfacing topical clusters in unhandled user requests, including utterance cleanup, a designated clustering procedure and its extensive evaluation, a novel approach to cluster representatives extraction, and cluster naming. We demonstrate the benefits of the suggested clustering approach on multiple publicly available, as well as proprietary datasets for real-world task-oriented chatbots. The rest of the paper is structured as follows. We survey related work in Section 2 and detail our clustering procedure and its evaluation in Section 3. Cluster representatives selection is presented in Section 4, and the process used to assign clusters with names is described in Section 5. Finally, we conclude in Section 6.

2 Related Work

In the context of the pipeline approach to building goal-oriented dialog systems, our work is related to the task of intent detection, performed by the NLU component. Intent detection is normally formulated as a standalone classification task (Xu and Sarikaya, 2013; Guo et al., 2014; Chen et al., 2019), which is loosely interlaced with the successive tasks in the pipeline. Out-of-domain utterance detection, which is the task of accurately discriminating between requests that are within- and outside the scope of a system, has gained much attention recently (Schuster et al., 2019; Larson et al., 2019; Gangal et al., 2020; Cavalin et al., 2020). Contrary to these works, we assume a set of utterances already labeled by a system's NLU module as unrecognized; these are user requests that the system failed to attribute to an existing intent. We demonstrate an end-to-end approach for extracting potentially actionable insights from these utterances, by making them easily accessible to a conversation analyst.

Clustering is one of the most useful techniques for extracting insights from data in an unsupervised manner. In the context of text, clustering typically refers to the task of grouping together units (e.g., sentences, paragraphs, documents) carrying similar semantics, such that units in the same cluster are more semantically similar to each other than those in different clusters. The unique nature of our setting imposes two constraints on the clustering algorithm: (1) unknown number of partitions, and (2) tolerating *outliers* that lie isolated in low-density regions. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996) and its hierarchical version (HDBSCAN) (McInnes et al., 2017) are two common choices that satisfy these requirements. We evaluate our clustering approach against (H)DBSCAN and show its benefits across multiple datasets.

Another popular clustering algorithm that determines the number of partitions is MeanShift (Cheng, 1995), a non-parametric method for locating the maxima of a density function. Outlier detection can further be achieved with MeanShift by considering only clusters that exceed a predefined minimal size, where the rest are assigned to the outlier pool. MeanShift yielded inferior performance in all our experiments; we, therefore, exclude its results from this work.

3 Clustering of Unrecognized Requests

Consider a virtual assistant aimed to attend to public questions about Covid-19. The rapidly evolving situation with the pandemic means that novel requests are likely to be introduced to the bot on a daily basis. For example, changes in international travel regulations would entail requests related to PCR test availability, and the decision to offer booster shots for seniors might cause a spike in questions about vaccine appointments for elderly citizens. Monitoring and promptly detecting these topics is fundamental for continuous bot improvement. We next describe the pipeline we apply, including utterance cleanup, clustering, cluster representative extraction, and cluster naming.

3.1 Cleaning and Filtering Utterances

Clustering unrecognized utterances aims at gaining topical insights into client needs that are currently poorly covered by the automatic reply system. In some cases, these utterances include easily identifiable, yet practically useless clusters, such as greetings (‘hello, how are you?’), acknowledgements (‘thank you’), or other statements of little practical importance (‘would you please check that for me?’). Generally treated as dialog *fluff*, these statements and their semantic equivalents can be filtered out from the subsequent processing.

We address this issue by manually collecting a sample set of fluff utterances: a set of domain-independent ones and a (small) set of domain-specific ones, where both are treated as anchors for data cleanup. Specifically, given a predefined anchor set of fluff utterances F, and a set of utterances subject for clustering U, we encode utterances in both F and U into their semantic representations using the SentenceTransformer (ST) encoder (Reimers and Gurevych, 2019)[1], and filter out each utterance $u{\in}U$ that exceeds a minimal cosine similarity threshold to any fluff utterance $f{\in}F$. We set the similarity threshold to 0.7 using qualitative evaluation over the $[0.5, 0.8]$ range. Requests such as ‘hi, how are you doing today’, and ‘thanks for your help’ would be filtered out prior to the clustering procedure since they closely resemble utterances from the anchor fluff set.

3.2 Clustering Utterances

Here we describe the main clustering procedure followed by an optional single merging step.

3.2.1 Main Clustering Procedure

Clustering requirements Multiple traits make up an effective clustering procedure in our scenario. First, the number of clusters is unknown, and has to be discovered by the clustering algorithm. Second, the nature of data typically implies several large and coherent clusters, where users repeatedly introduce very similar requests, and a very long tail of unique utterances that do not have similar counterparts in the dataset. While the latter are of somewhat limited importance, they can amount to a significant ratio of the input data. There is an evident trade-off between the size of the generated clusters, their density or sparsity, and the amount of outliers: smaller and denser clusters entail larger amounts of outliers. The decision regarding the precise outcome granularity may vary according to domain and bot maturity. Growing deployments, with a high volume of unrecognized requests could benefit from surfacing large and coarse topics that are subject to automation. That said, mature deployments are likely to focus on fine-grained coherent clusters of utterances, introducing enhancements into the existing solution. Our third requirement is, therefore, a configurable density of the outcome clusters, which can be set up prior to the clustering procedure. Figure 2 illustrates a typical outcome of the clustering process; identified clusters are depicted in color, while the outliers, which are the majority of instances in this case, appear in grey.

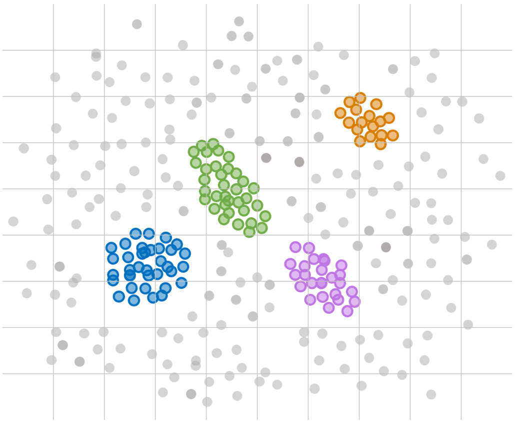

Figure 2: Example outcome of the clustering process. Identified clusters (four in this case) are depicted in color, while outliers appear in grey.

Existing clustering solutions can be roughly categorized across two major dimensions in terms of functional requirements: those requiring a fixed number of output clusters (1.a) and those that do not (1.b); those forcing cluster assignment on the entire dataset (2.a) and those tolerating outliers (2.b). Our clustering solution should accommodate (1.b) and (2.b): the number of clusters is determined by the clustering procedure, allowing for outliers. DBSCAN (Ester et al., 1996) and its descendant variants constitute a popular family of clustering solutions that satisfies these requirements; we, therefore, evaluate our algorithm against implementations of DBSCAN and its hierarchical version HDBSCAN (McInnes et al., 2017).

Data representation Given a set of m unhandled utterances $U{=}(u_1,\ u_2,\ ...,\ u_m)$, we compute their vector representations $E{=}(e_1,\ e_2,\ ...,\ e_m)$ using a sentence encoder. A distance matrix D of size $m{\times}m$ is then computed, where $D[i,j]{=}1.0{-}cos(e_i, e_j)$. The matrix D is further used as an input to the core clustering algorithm.

Radius-based clustering (RBC) We introduce a variant of the popular k-means clustering algorithm.

[1] Using the Universal Sentence Encoder (USE) (Cer et al., 2018) yielded similar results, see Section 3.3.2 for details.

This variant complies with our clustering requirements by (1) imposing a strict cluster assignment criterion and (2) eventually omitting points that do not constitute clusters exceeding a predefined size. Specifically, we iterate over randomly-ordered vectors in E, where each utterance vector can be assigned to an existing cluster if certain conditions are satisfied; otherwise, it initiates a new cluster. To join an existing cluster, the utterance is required to surpass a predefined similarity threshold min_sim for the cluster's centroid[2], implying its placement within a certain *radius* from the centroid. If multiple clusters satisfy the similarity requirement, the utterance is assigned to the cluster with the highest proximity i.e., the cluster with the highest semantic similarity to its centroid. Additional iterations over the utterances are further performed, re-assigning them to different clusters if needed, until convergence or until a pre-defined number of iterations is exhausted. The amount of clusters generated by the final partition is controlled by the predefined min_size value: elements that constitute clusters of small size (in particular, those with a single member) are considered outliers. Algorithm 1 presents the Radius-based Clustering (RBC) pseudo-code.

Algorithm 1: Radius-based Clustering

```
input: E (e1, e2, ... en) /* elements */
input: D (n×n) /* dist matrix */
input: min_sim /* min similarity */
input: min_size /* min cluster size */
$C \leftarrow \emptyset$
while convergence criteria are not met do
    for each element $e_i \in E$ do
        if the highest similarity of $e_i$ to any existing cluster exceeds min_sim then
            assign $e_i$ to its most similar cluster $c$
            re-calculate the centroid of $c$
        else
            create a new cluster $c'$ and assign $e_i$ to it
            set the centroid of $c'$ to be $e_i$
            add $c'$ to $C$
/*clusters with fewer elements than the predefined $min\_size$ are considered outliers */
return: each $c \in C$ of size exceeding $min\_size$
```

3.2.2 Merging Clusters

Cluster merging has been extensively used as a means to determine the optimal clustering outcome in the scenario where the 'true' number of partitions is unknown (Krishnapuram, 1994; Kaymak and Setnes, 2002; Xiong et al., 2004). These start with a large number of clusters and iteratively merge compatible partitions until the optimization criteria is satisfied. Beginning with a fine-grained partitioning, we perform a single step of cluster merging, combining similar clusters into larger groups. A similar outcome could potentially be obtained by relaxing the *min_sim* similarity threshold and thereby, generating more heterogeneous flat clusters in the first place. However, a single step of cluster merging yielded results that outperform flat clustering on a range of datasets (see Table 3 and Section 3.3.2 for details).

Classical agglomerative hierarchical clustering (AHC) algorithms merge pairs of lower-level clusters by minimizing the agglomerative criterion: a similarity requirement that has to be satisfied for a pair of clusters to be merged. Similar to AHC, we seek to merge clusters exhibiting high mutual similarity. In contrast to AHC, our approach is not pair-wise, rather it constitutes a subsequent invocation of Algorithm 1 that takes inter-cluster (and not inter-utterance) distance matrix D_c as its input. We next describe two approaches for building this distance matrix towards a single merging step.

Semantic Merging Formally, given a set of clusters C of size $k{=}|C|$, identified by Algorithm 1, we compute the set of cluster centroid vectors (cn_1, cn_2, ..., cn_k); these vectors are assumed to reliably represent the semantics of their corresponding clusters. A distance matrix D_c is then computed by calculating the pairwise semantic distance between all pairs of centroids in the set C. D_c is further used as an input to subsequent invocation of the RBC algorithm, where the *min_sim* parameter can possibly differ from the previous invocation.

Keyword-based Merging User requests to a goal-oriented dialog system are likely to be characterized by the extensive use of a domain-specific lexicon. For example, in the domain of banking, we are likely to encounter terms related to 'accounts', 'transactions', and 'balance', while in the context of a Covid-19 Q&A bot, the lexicon is likely to contain extensive use of words related to 'vaccine', 'boosters', 'appointments', and so on. Although impressive at capturing meaning, semantic representations do not necessarily capture the domain-specific notion of similar requests. For example,

[2] Following the k-means notation, we compute a cluster's centroid as the arithmetic mean of its member vectors.

cluster name: **difference covid flu** (28)	cluster name: **covid pregnancy** (17)
is covid the same as the flu? (4)	covid 19 and pregnancy (10)
how is covid different from the flu? (3)	covid risks for a pregnant woman (4)
what is the difference between covid 19 and flu?	what is the risk of covid for pregnant women?
what's the difference between covid and flu	is covid-19 dangerous when pregnant?
is the covid the same as cold?	7 months pregnant and tested positive for covid, any risks?
covid vs flu vs sars	covid 19 during pregnancy

Table 1: Example clusters of user requests generated by the RBC algorithm when applied on the Covid-19 dataset. Only a partial list of cluster members is presented in the table; the number in parenthesis denotes a cluster size.

the two utterances 'covid 19 and pregnancy' and '7 months pregnant and tested positive for covid, any risks?' do not exhibit exceptional semantic similarity, while practically they should be clustered together. The intuition that stems from the fact that both sentences contain 'pregnant'/'pregnancy', and 'covid' – words typical of the underlying domain. We therefore suggest the additional, keyword-based merging approach, as detailed below.

A common way to extract lexical characteristics of a corpus is using a *log-odds ratio with informative Dirichlet prior* (Monroe et al., 2008) – a method that discovers markers with excessive frequency in one dataset compared to another. We used the collection of unhandled utterances as our target corpus and a random sample of 100K sentences from a Wikipedia dump[3] as our background neutral dataset. Setting the strict log-odds score of -5, markers identified for the dataset of Covid-19 requests included {'quarantine', 'measures', 'emergency', 'pregnant', 'sick', 'leave', 'risk'}.

Given a set of markers, we now define cluster similarity as follows: we denote the set of domain-specific markers discovered by the log-odds ratio procedure by M and the set of top-k most frequent words[4] in two clusters c_1 and c_2, by W_1 and W_2, respectively. The similarity of c_1 and c_2 is then defined to be proportional to the number of markers from M that can be found in both W_1 and W_2: $sim(c_1, c_2) \propto |M \cap W_1 \cap W_2|$, where $|M|$ amounts to the maximal possible similarity. Pairwise cluster distances are further computed by normalizing the similarity values to the $[0, 1]$ range, and subtracting them from 1. A distance matrix D_c is constructed by calculating pairwise distance on the set of clusters in C, and is further used as an input to subsequent invocation of the RBC algorithm, with an adjusted *min_sim* threshold.

Following this definition and assuming a sample set of domain specific markers {'covid', 'risk', 'quarantine', 'pregnant', 'appointment', 'test', 'positive'}, the two utterances 'covid 19 and pregnancy' and '7 months pregnant and tested positive for covid, any risks?' will exhibit considerable keyword-based similarity (intersection size=2), despite only moderate semantic proximity.

Example Clustering Result Table 1 presents two example clusters generated from user request to the Covid-19 bot. We applied the main RBC clustering procedure and a subsequent keyword-based merge step. As can be observed, semantically related utterances are grouped together, where the number beside an utterance reflects its frequency in the cluster. As an example, 'is covid the same as the flu?' was asked four times by different users.

3.3 Evaluation of Clustering

We performed a comparative evaluation of the proposed clustering algorithm and HDBSCAN[5], using common clustering evaluation metrics. The nature of topical distribution of unrecognized utterances is probably most closely resembled by *intent classification* datasets, where semantically similar training examples are grouped into classes, based on their underlying intent. We used these classes to simulate cluster partitioning for the purpose of evaluation. We make use of three publicly available intent classification datasets (Liu et al. (2019), Larson et al. (2019) and Tepper et al. (2020)), as well as three datasets from real task-oriented chatbots in varying domains. Table 2 presents details for the datasets used in our evaluation.

3.3.1 Evaluation Approach

The main approaches to clustering evaluation include extrinsic methods, which assume a ground truth, and intrinsic methods, which work in the

[3] We used the Wikipedia 2006 dump available at `https://nlp.lsi.upc.edu/wikicorpus/`.

[4] k=10 by qualitative evaluation over the $[3, 15]$ range.

[5] DBSCAN resulted in outcome systematically inferior to HDBSCAN; hence, it was excluded from further experiments.

dataset	intents	examples	mean	STD
Liu et al. (2019)	46	20849	453.23	896.34
Larson et al. (2019)	150	22500	150.00	0.00
Tepper et al. (2020)	57	844	14.80	14.16
dataset1	157	5954	37.92	26.74
dataset2	135	2387	17.68	25.28
dataset3	112	1821	16.25	11.42

Table 2: Datasets details: the number of intents, total training examples, mean and STD of the num of examples. We excluded out-of-scope examples from the Larson et al. (2019) dataset for the sake of evaluation.

absence of ground truth. Extrinsic techniques compare the clustering outcome to a human-generated *gold standard* partitioning. Intrinsic techniques assess the resulting clusters by measuring characteristics such as cohesion, separation, distortion, and likelihood (Pfitzner et al., 2009). We employ two popular extrinsic and intrinsic evaluation metrics: adjusted random index (ARI, (Hubert and Arabie, 1985)) and Silhouette Score (Rousseeuw, 1987). We vary the parameters of the RBC algorithm: merge type with none vs. semantic vs. keyword-based, see Section 3.2.2); the encoder used for distance matrix construction using ST vs. USE; min similarity threshold used as a cluster “radius” (see Algorithm 1 for details). Both ARI and Silhouette yield values in the [-1, 1] range, where -1, 0 and 1 mean incorrect, arbitrary, and perfect assignment, respectively.

The unique nature of our clustering requirements introduces a challenge to standard extrinsic evaluation techniques. Specifically, the min cluster size attribute controls the amount of outliers, by considering only clusters that exceed the minimal number of members (see Figure 2). As such, a high *min_size* value will yield a large amount of left-out utterances, while a *min_size*=1 will partition the entire data, including single-member clusters. Aiming to mimic the ground truth partition (i.e, the intent classification datasets), we set the *min_size* attribute according to the minimal class size in the dataset, subject to evaluation. For example, this attribute was set to 150 for the Larson et al. (2019) dataset, but to 2 for dataset2.

Both evaluation techniques assume partitioning of the input space. Therefore, for our evaluation, we exclude the set outliers generated by our clustering algorithm altogether: only the subset of instances constructing the outcome clusters (e.g., instances depicted in color in Figure 2) was used to compute both ARI and Silhouette. For completeness, we also report the ratio of a dataset utterances covered by the generated partition (‘% clustered’ in Table 3), where the higher, the better.

3.3.2 Evaluation Results

Table 3 presents the results of our evaluation. Clearly, the RBC algorithm outperforms HDBSCAN across the board for both ARI and Silhuette scores, with the exception of dataset3, where the second best ARI score (0.37) is obtained by RBC along with over 80% of clustered utterances (compared to only 49.79% by HDBSCAN). HDBSCAN also outperforms RBC in terms of the ratio of clustered utterances for Liu et al. (2019) and dataset1. However, these results are achieved by a nearly arbitrary partition of the input data, as mirrored by the extremely low ARI and Silhuette scores. We conclude that RBC outperforms its out-of-the-box counterpart on virtually all datasets in this work.

The ratio of clustered examples (% clustered) exhibits considerable variance among the datasets; this result is indicative of the varying levels of semantic coherence of the underlying intent classes, which are typically constructed manually by a bot designer. As such, over 87% of all training examples were covered by the clustering procedure for dataset3, but only 33.90% for Larson et al. (2019). Although it generated different final outcome, the merging step does not affect the ratio of clustered utterances, which is determined by the first clustering round. For example, 87.18% of the utterances are clustered for all three merge types when using the ST encoder for dataset3.

Various merging strategies, encoders, and similarity thresholds show the benefits for different datasets, with no single parameter configuration outperforming others systematically. This result implies that the decision regarding the precise clustering configuration is dependent on the specific dataset, and should be made per qualitative or quantitative evaluation, where possible.

4 Selecting Cluster Representatives

Contemporary large-scale deployments of virtual assistants must cope with increasingly high volumes of incoming user requests. A typical large task-oriented system can accept over 100K requests (i.e., user utterances) per day, where the amount of conversations that pass the initial step of intent identification can vary between 40% and 80%. Consequently, tens of thousands of requests can be identified as unrecognized on a daily basis. Clustering

	algo	RBC algorithm												HDBSCAN	
	merge type	no merge				semantic merge				keyword merge					
	encoder type	USE		ST		USE		ST		USE		ST		USE	ST
	sim threshold	0.55	0.60	0.55	0.60	0.55	0.60	0.55	0.60	0.55	0.60	0.55	0.60	*	*
Liu	ARI	0.43	0.42	0.32	0.40	0.60	**0.74**	0.44	0.47	0.60	**0.74**	0.43	0.48	0.42	0.03
	Silhouette	0.50	0.47	0.36	0.42	0.59	**0.67**	0.50	0.58	0.59	**0.67**	0.40	0.43	0.39	0.09
	% clustered	14.00	12.12	16.09	12.03	14.00	12.12	16.09	12.03	14.00	12.12	16.09	12.03	12.69	**38.36**
Larson	ARI	0.87	**0.89**	0.86	0.86	0.64	0.68	0.76	0.87	0.66	0.68	0.71	0.75	0.49	0.69
	Silhouette	0.40	0.47	0.47	0.50	0.42	0.48	0.50	**0.54**	0.37	0.38	0.39	0.47	0.39	0.47
	% clustered	26.90	16.29	**33.90**	32.60	26.90	16.29	**33.90**	32.60	26.90	16.29	**33.90**	32.60	24.92	32.98
Tepper	ARI	0.71	0.66	0.65	0.65	0.72	**0.73**	0.52	0.61	0.71	0.66	0.65	0.65	0.69	0.67
	Silhouette	0.46	0.45	0.47	0.49	**0.49**	0.51	0.37	0.47	0.46	0.45	0.47	0.49	0.45	0.46
	% clustered	84.72	79.68	88.18	85.12	84.72	79.68	**88.18**	85.12	84.72	79.68	88.18	85.12	58.31	60.15
dataset1	ARI	0.36	0.32	0.52	0.54	**0.66**	0.63	0.38	0.44	0.40	0.37	0.51	0.53	0.00	0.00
	Silhouette	0.16	0.17	0.16	**0.20**	0.17	0.18	0.11	0.15	0.13	0.12	0.15	0.19	0.00	0.00
	% clustered	38.29	25.18	59.78	46.87	38.29	25.18	59.78	46.87	38.29	25.18	59.78	46.87	83.24	**97.90**
dataset2	ARI	0.45	0.40	0.56	0.42	0.58	0.45	0.46	0.54	**0.61**	0.52	0.56	0.45	0.55	0.59
	Silhouette	0.31	0.37	0.27	**0.39**	0.22	0.35	0.32	0.33	0.34	0.33	0.25	0.33	0.34	0.35
	% clustered	59.17	47.59	**74.28**	63.26	59.17	47.59	**74.28**	63.26	59.17	47.59	**74.28**	63.26	23.46	36.22
dataset3	ARI	0.32	0.28	0.34	0.37	0.29	0.31	0.24	0.30	0.31	0.28	0.31	0.34	0.37	**0.38**
	Silhouette	0.22	0.24	0.28	0.28	0.19	0.22	0.27	0.26	0.21	0.24	0.26	0.26	0.23	**0.32**
	% clustered	77.60	68.19	**87.18**	80.43	77.60	68.19	**87.18**	80.43	77.60	68.19	**87.18**	80.43	37.55	49.79

Table 3: Clustering evaluation results. '*' in HDBSCAN columns denotes similarity threshold (0.55 or 0.60) yielding the highest results (the threshold varies per dataset). The best result in a row is boldfaced.

these utterances would result in large clusters that are often impractical for manual processing. Providing conversation analysts with a limited set of *cluster representatives* can help extract value from the unrecognized data.

4.1 Representative Characteristics

A plausible set of representative cluster utterances would have to satisfy two desirable properties: utterance *centrality* and *diversity*. We define an utterance centrality to be proportional to its frequency in a cluster: requests with higher frequency should be boosted, since they are typical of the way people express their need to the bot. The diversity of the utterance set mirrors the subtle differences in the phrasing and meaning of utterances; these reflect the various ways people can express the same need.

Sampling randomly from a cluster may result in a sub-optimal set of representatives, in terms of both centrality and diversity. Consider the example where no 'covid 19 and pregnancy' requests (Table 1, right) are selected as representatives (low centrality), or both 'what is the difference between covid 19 and flu?' and 'what's the difference between covid and flu' (Table 1, left) are selected (low diversity). Contrary to these examples, the set {'is covid the same as the flu?', 'is the covid the same as cold?', 'covid vs flue vs sars'} contains utterance of high centrality (the first utterance), and comprehensive coverage of the entire cluster semantics.

4.2 Selecting Representatives

Given a set of utterance vectors represented in a k-dimensional Euclidean space, the volume enclosed by the vectors is influenced by two factors – the angle made by the vectors with respect to each other and their length. More orthogonal vectors span higher volume in the semantic space. Similarly, the higher is the length of the vectors, the higher is the volume they encompass. Intuitively, the angle made by the vectors is indicative of how similar the corresponding utterances are. Moreover, if the length of the vectors is equated to the centrality of the corresponding utterances, we reduce the problem of selecting k diverse utterances with high centrality to that of maximizing the volume encompassed by k corresponding vectors.

Selection Approach Given a cluster c of size n, we first project the encodings of the n utterances onto a unit sphere. We further take into consideration the factor of centrality by scaling the vectors' length based on their frequency in a cluster. The volume enclosed by any subset of vector representations is now affected by both angles and the vectors' length, thereby simultaneously satisfying the two objectives for representative set selection: centrality and diversity. Figure 3 illustrates the idea of selecting cluster representatives; we use a 2D space for the sake of interpretability.

Assuming n vectors in a vector space, the square of the k-dimensional volume enclosed by the vec-

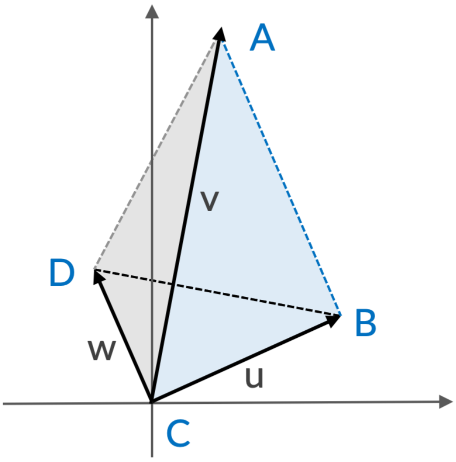

Figure 3: Simplified illustration of cluster representatives selection. While taking into consideration only diversity, the widest angle is ∠BCD, meaning vectors u and w are the most diverse out of the three visualized vectors. Assuming vector length that is proportional to the vectors' centrality in a cluster, the chart shows a larger enclosed area between the vectors u and v, out of all enclosed areas between pairs of vectors; these vectors will be selected as cluster representatives for k=2.

tors is proportional to the Gram-determinant of the vectors. Given n utterances, we select k diverse and central utterances by computing vectors' similarity matrix, and finding a square sub-matrix of size k that has a high determinant; this can be achieved by using a determinant's point process to sample such a sub-matrix (Gong et al., 2014; Celis et al., 2018). We make use of the freely available DPPy Python package[6] for this purpose.

As a concrete example, for the two clusters in Table 1 and k=3, two represenative sets were selected: {'is covid the same as the flu?', 'is the covid the same as cold?', 'covid vs flue vs sars'} and {'covid 19 and pregnancy', 'covid risks for a pregnant woman', '7 months pregnant and tested positive for covid, any risks?'}.

5 Naming Clusters

Assigning cluster with names, or labels, is an essential step towards their consumability. Common approaches to this task resort to simple but reliable techniques based on keyword extraction, such as *tf-idf*; many of these techniques made their way into the first large-scale information retrieval (IR) systems (Ramos et al., 2003; Aizawa, 2003).

We treat all utterances in individual clusters from a set C=$(c_1, c_2, ..., c_k)$ as distinct documents. We first applied lemmatization to these documents using the spacy toolkit[7] (Honnibal and Montani, 2017), excluded stopwords, and further ranked all unigram, bigram, and trigram token sequences by their tf-idf score: term-frequency boosts ngrams typical of a cluster, and inverted-document-frequency down-weights the importance of ngrams, common across clusters.

Favoring long names (e.g., a trigram) over short ones (e.g., a unigram), we defined a tf-idf score threshold for each ngram with more permissive, lower scores for trigrams and higher ones for unigrams. Score thresholds were optimized by qualitative evaluation over the $[0.10, 0.75]$ range, and were set to 0.650, 0.400 and 0.150 for unigrams, bigrams and trigrams, respectively. We further sorted the candidate key-phrases by their length in a primary sort, and by score for a secondary sort. The first ngram to exceed its pre-defined corresponding threshold was selected as the cluster name. Table 1 presents names automatically assigned to the two sample clusters identified in the Covid-19 dataset: 'difference covid flu' and 'covid pregnancy'.

6 Conclusions and Future Work

Analyzing unrecognized user requests is a fundamental step towards improving task-oriented dialog systems. We present an end-to-end pipeline for cleanup, clustering, representatives selection, and cluster naming – procedures that facilitate the effective and efficient exploration of utterances unrecognized by the NLU module. We propose a simple clustering variant of the popular k-means algorithm, and show that outperforms its out-of-the-box counterparts on a range of metrics. We also suggest a novel approach to extracting representative utterances from a cluster while simultaneously optimizing their centrality and diversity.

Our future work includes evaluation of our clustering approach with additional datasets, exploration of additional approaches to representative set selection, and advanced techniques for cluster naming. Leveraging clustering results to automatically identify actionable recommendations for conversation analyst is another venue of significant practical importance, we plan to pursue.

[6] https://github.com/guilgautier/DPPy

[7] https://spacy.io/

References

Akiko Aizawa. 2003. An Information-Theoretic Perspective of tf–idf Measures. *Information Processing & Management*, 39(1):45–65.

Paulo Cavalin, Victor Henrique Alves Ribeiro, Ana Appel, and Claudio Pinhanez. 2020. Improving Out-of-Scope Detection in Intent Classification by Using Embeddings of the Word Graph Space of the Classes. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3952–3961.

Elisa Celis, Vijay Keswani, Damian Straszak, Amit Deshpande, Tarun Kathuria, and Nisheeth Vishnoi. 2018. Fair and diverse dpp-based data summarization. In *International Conference on Machine Learning*, pages 716–725. PMLR.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, et al. 2018. Universal Sentence Encoder. *arXiv preprint arXiv:1803.11175*.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A Survey on Dialogue Systems: Recent Advances and New Frontiers. *Acm Sigkdd Explorations Newsletter*, 19(2):25–35.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. BERT for Joint Intent Classification and Slot Filling. *arXiv preprint arXiv:1902.10909*.

Yizong Cheng. 1995. Mean Shift, Mode Seeking, and Clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. Density-Based Spatial Clustering of Applications with Noise. In *Int. Conf. Knowledge Discovery and Data Mining*, volume 240, page 6.

Varun Gangal, Abhinav Arora, Arash Einolghozati, and Sonal Gupta. 2020. Likelihood Ratios and Generative Classifiers for Unsupervised out-of-domain Detection in Task Oriented Dialog. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7764–7771.

Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. 2014. Diverse sequential subset selection for supervised video summarization. *Advances in neural information processing systems*, 27.

Jonathan Grudin and Richard Jacques. 2019. Chatbots, Humbots, and the Quest for Artificial General Intelligence. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*.

Daniel Guo, Gokhan Tur, Wen-tau Yih, and Geoffrey Zweig. 2014. Joint Semantic Utterance Classification and Slot Filling with Recursive Neural Networks. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 554–559. IEEE.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing. *Sentometrics Research*.

Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. 2020. Challenges in Building Intelligent Open-Domain Dialog Systems. *ACM Transactions on Information Systems (TOIS)*, 38(3):1–32.

Lawrence Hubert and Phipps Arabie. 1985. Comparing Partitions. *Journal of classification*, 2(1).

Uzay Kaymak and Magne Setnes. 2002. Fuzzy Clustering with Volume Prototypes and Adaptive Cluster Merging. *IEEE Transactions on Fuzzy Systems*, 10(6):705–712.

Raghu Krishnapuram. 1994. Generation of Membership Functions via Possibilistic Clustering. In *Proceedings of 1994 IEEE 3rd International Fuzzy Systems Conference*, pages 902–908. IEEE.

Knut Kvale, Olav Alexander Sell, Stig Hodnebrog, and Asbjørn Følstad. 2019. Improving Conversations: Lessons Learnt from Manual Analysis of Chatbot Dialogues. In *International workshop on chatbot research and design*, pages 187–200. Springer.

Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. 2019. An Evaluation dataset for intent classification and out-of-scope prediction. 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 1311–1316.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2019. Benchmarking Natural Language Understanding Services for Building Conversational Agents. In *10th International Workshop on Spoken Dialogue Systems Technology 2019*, volume 714, pages 165–183. Springer.

Stuart Lloyd. 1982. Least Squares Quantization in PCM. *IEEE transactions on information theory*, 28(2):129–137.

Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical Density Based Clustering. *Journal of Open Source Software*, 2(11):205.

Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin'words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict. *Political Analysis*.

Darius Pfitzner, Richard Leibbrandt, and David Powers. 2009. Characterization and Evaluation of Similarity Measures for Pairs of Clusterings. *Knowledge and Information Systems*, 19(3):361–394.

Juan Ramos et al. 2003. Using tf-idf to Determine Word Relevance in Document Queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. 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 3982–3992.

Peter J Rousseeuw. 1987. Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis. *Journal of computational and applied mathematics*, 20:53–65.

Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2019. Cross-Lingual Transfer Learning for Multilingual Task Oriented Dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3795–3805.

Naama Tepper, Esther Goldbraich, Naama Zwerdling, George Kour, Ateret Anaby Tavor, and Boaz Carmeli. 2020. Balancing via Generation for Multi-Class Text Classification Improvement. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1440–1452.

Xuejian Xiong, Kap Luk Chan, and Kian Lee Tan. 2004. Similarity-Driven Cluster Merging Method for Unsupervised Fuzzy Clustering. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 611–618.

Puyang Xu and Ruhi Sarikaya. 2013. Convolutional Neural Network Based Triangular CRF for Joint Intent Detection and Slot Filling. In *2013 ieee workshop on automatic speech recognition and understanding*, pages 78–83. IEEE.