Taylor & Francis
Taylor & Francis Group

Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Automated Creation of an Intent Model for Conversational Agents

Alberto Benayas, Miguel Angel Sicilia & Marçal Mora-Cantallops

To cite this article: Alberto Benayas, Miguel Angel Sicilia & Marçal Mora-Cantallops (2023) Automated Creation of an Intent Model for Conversational Agents, Applied Artificial Intelligence, 37:1, 2164401, DOI: 10.1080/08839514.2022.2164401

To link to this article: https://doi.org/10.1080/08839514.2022.2164401

Published online: 20 Jan 2023.

APPLIED ARTIFICIAL INTELLIGENCE
2023, VOL. 37, NO. 1, e2164401 (16 pages)
https://doi.org/10.1080/08839514.2022.2164401

Taylor & Francis
Taylor & Francis Group

Automated Creation of an Intent Model for Conversational Agents

Alberto Benayas, Miguel Angel Sicilia, and Marçal Mora-Cantallops

Computer Science, University of Alcalá, Alcalá de Henares, Spain

ABSTRACT

Conversational Agents (CA) are increasingly being deployed by organizations to provide round-the-clock support and to increase customer satisfaction. All CA have one thing in common despite the differences in their design: they need to be trained with users' intents and corresponding training sentences. Access to proper data with acceptable coverage of intents and training sentences is a big challenge in CA deployment. Even with the access to the past conversations, the process of discovering intents and training sentences manually is not time and cost-effective. Here, an end to end automated framework that can discover intents and their training sentences in conversation logs to generate labeled data sets for training intent models is presented. The framework proposes different feature engineering techniques and leverages dimensionality reduction methods to assemble the features, then applies a density-based clustering algorithm iteratively to mine even the least common intents. Finally, the clustering results are automatically labeled by the final algorithm.

ARTICLE HISTORY
Received 2 August 2022
Revised 19 November 2022
Accepted 7 December 2022

Introduction

Conversational Agents (CA) are increasingly being employed to provide round-the-clock services to customers (Deloitte 2019; Research, & Markets 2021). Many companies are using commercial solutions to deploy their CA. Despite the differences among such solutions, they have one thing in common: they all need to be trained on large amounts of data. To train a CA effectively, as many intents as possible need to be covered, along with the proper training sentences for those intents.

Many companies have access to the conversation history between human agents and customers. Traditionally, conversational designers go through the corpus to discover intents and training sentences, relying on their experience to design the conversation flow. This is time-consuming, labor-intensive, and lacks consistency as different designers might have different ideas about the intent and flow of conversation.

CONTACT Alberto Benayas, alberto.benayas@edu.uah.es Computer Science, University of Alcalá, Alcalá de Henares, Spain

The intent label sets are usually decided by conversational designers in advance based on their experience. However, it is not often known the exact intents a new unlabeled data has, and the assigned label names maybe, to some extent, be subjective. There have been attempts to help the human labeling process. For instance, Wen et al. (2017) suggested an improvement over Wizard-of-Oz (Kelley 1984) data collection method by incorporating crowd-sourcing to collect domain-specific data.

Finally, data used to train CAs generally belongs to a narrow domain and therefore the differences across intents are in many cases very subtle. In addition, CA data is generally based on short texts between customers and agents that do not include extensive information about its topic or meaning.

Haponchyk et al. (2018) proposed a supervised approach to automatically cluster questions into user intents. Williams et al. (2015) proposed facilitating domain experts' work by allowing them to build an intent model by working on the intent definition, labeling, and evaluation through user interfaces. Shi et al. (2018) leveraged a dynamic hierarchical clustering method for intent and slot labeling and showed the effectiveness of their model in reducing the cost of labeling. Chatterjee and Sengupta (2020) proposed an extension to a density-based clustering algorithm to mine the intents in a highly skewed corpus of conversation history. Furthermore, they also argue that their framework can detect even the least common intents which are usually left out due to the nature of common clustering algorithms.

In this work, a framework for a data-driven approach to train conversational agents is proposed, with the objective of improving on the most common approach of training them on conversational designers' intuition. The main contribution of the work is as follows:

- A framework to extract intents and their associated sentences from raw conversation history is proposed. It works in three different steps: a) representing utterances using an assembly of feature sets, b) dimensionality reduction and c) automatic clustering of similar utterances.
- A novel feature extraction methodology that can cater to the diversity of topics in conversations and utterances of different lengths.
- An automated dataset labeling methodology that works without human intervention, can be used to build an intent model.

The rest of this paper is structured as follows. In section 2 the related work is reviewed. In section 3, the methodology that has been followed is detailed. Section 4 describes the design of the experiments that are carried out and presented in section 5, finally, section 6 discusses the results, implications, and limitations of the current work, while section 7 concludes.

Related Work

Identifying intents and the way they are sentenced by the customers is crucial in training reliable task-oriented conversational agents. Intents are labels that are given to a group of customers' inquiries that summarize what they want. Traditionally, conversational designers use their expertise and subject knowledge to come up with a list of intents and different sentences that they think might be uttered by the customers to convey those intents. However, a more desirable approach would be to discover the intents and their corresponding sentences from real conversations between customers and agents.

Collecting good quality data and labeling it properly is an expensive and costly task. Previous works aimed to reduce the labeling cost and effort using various techniques. In Mallinar et al. (2018) a labeling framework is proposed using weak supervision functions. Other works use techniques such as clustering (Chatterjee and Sengupta 2020; Shi et al. 2018), semi-supervised learning (Thomas 2009), transfer learning (Goyal, Metallinou, and Matsoukas 2018) and active learning (Settles 2009). Clustering in particular is one of the most popular methods of identifying patterns in data. Different algorithms are available for clustering purposes, with some of the most popular and widely used being K-Means (Lloyd 1982), DBSCAN (Ester et al. 1996) and HDBSCAN (McInnes, Healy, and Astels 2017). Each of these algorithms has its own set of pros and cons. For example, K-means is very fast and assigns every point to a cluster but the number of clusters must be known beforehand. Moreover, the clusters have to be of spherical shape. On the other hand, density-based algorithms like DBSCAN or HDBSCAN are appropriate when there is no prior knowledge about the number of clusters. The problem with these algorithms is that they are not able to assign a cluster to all points and, thus, some of them are left out of any cluster (and labeled as noise) in the end.

Shi et al. (2018) provided several feature engineering methods, but they assumed that no feature set should overweight the others as they all should have the same predictive power. Chatterjee and Sengupta (2020) utilized a modified version of DBSCAN (ITER-DBSCAN) to discover intents in low-density areas of the space. This work is

different from that of Chatterjee and Sengupta (2020) as we do not use prior knowledge about the number of clusters in the dataset for fine-tuning the clustering algorithm but we rather set the hyper-parameters on the training data and then let the algorithm find the intents on the test dataset which is what is done in real-world situations. This work is also different from that of Shi et al. (2018) as there is no reason to assume that all feature sets should weigh equally. This work proves that it is not true in most cases.

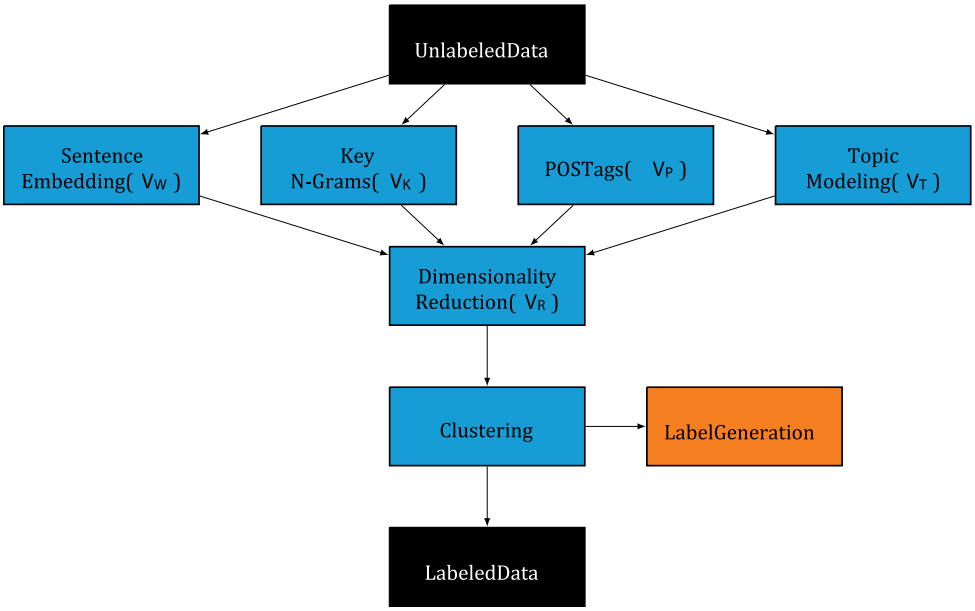

Figure 1. Process steps. In the first step, several features sets V_* are calculated and concatenated. In the second step, the different feature sets are passed through a dimensionality reduction process, in which the number of feature sets V_* are then concatenated to form a single feature set V_R. In the last step, the resulting feature set V_R is then passed to a density-based clustering process. The resulting clusters are then labeled using the TF-IDF algorithm. The goal of this step is to produce a reduced but reliable labeled dataset.

Methodology

Each conversation between agent and customer is considered as a sequence of queries and response pairs where the Q is the customer's and R is the agent's utterances. Our aim is to label each Q with its corresponding label which is the intent of the customer. The number of intents C is to be determined in the labeling process. In detail, the framework can be split into three steps. The full pipeline is described in Figure 1.

Feature Engineering

Utterances in a dialog are normally short texts, which in many cases do not contain a strong signal. For this reason, using a single feature engineering method might not be enough for all cases. In order to capture a rich signal, a set of feature engineering methods are used to extract semantic, lexical, syntactic and the topics in the text, which generate features sets V_* for utterances Q. A concatenation of different feature sets forms a richer representation of the inputs.

Semantics

To capture the semantics, sentence embeddings are calculated using transformers based models (Vaswani et al. 2017). Extracting embeddings from models such as BERT (Devlin et al. 2019), which is a deep bidirectional transformers model, can be performed by pooling the last hidden state, either maxing or averaging or by taking the CLS token's vector embeddings, which is a token that contains a sentence-level embedded representation. Also, a fine-tuned BERT model is trained using a Siamese BERT network, as described in (Reimers and Gurevych 2019). It is tuned on positive pairs by matching sentences belonging to the same class, and on negative examples by matching sentences from different classes. In situations in which there is no prior knowledge of the classes, which is in most cases, the positive examples are generated with the support of a probabilistic retrieval algorithm such as the BM25 OKAPI algorithm to match sentences with high similarity, as in (Thakur et al. 2021). The length of the vectors obtained using this method varies depending on the model, but typically they are 768, which is BERT's original design. This vector is called V_W.

Lexical

In many cases, especially in very short sentences, the intent is decided by the presence of certain lexical units. So this consideration is included by introducing the frequent key n-gram feature V_K. An n-gram is a contiguous sequence of n items from a given sample of text. $V_K = \{x_1,\cdots,x_c\}$ represents the information of n-grams in the utterance. After removing stop words, the top K n-grams (for $n = 1,2,3,4$) are chosen and the occurrence frequency of each n-gram as a discrete vector V_K is counted. If domain experts are available, they can also define or include a specific set of n-grams to be counted. The length of the resulting vectors is, thus, k_* 4.

Syntactical

It is assumed that the arrangement of the *part of speech* (POS) tags may affect the sentence syntactical structure. A POS tag is a special label assigned to each token in a text to indicate the part of speech and often also other grammatical categories. Therefore, a bag-of-POS as the POS tag feature V_P is used. Given n_p types of POS tags, $V_P = \{p_1,\cdots,p_{np}\}$ is a discrete vector in which each dimension p_i represents a POS tag POS_i.

Topics

To capture existing topics, each sentence is mapped to a vector of possible topics within the corpus. Firstly, a *term frequency-inverse document frequency* (TF-IDF) matrix of the whole dataset is generated. Using an *Latent Dirichlet Allocation* (LDA) topic model, for each utterance, a vector is built in which each dimension represents a topic and the value that represents the probability

of the sentence belonging to that topic. The length of the generated vector depends on the requested number of topics passed to the LDA model. This is denoted as vector V_T.

Dimensionality Reduction

The generated vectors could be too long, becoming unpractical, and, therefore, would hinder the subsequent clustering process. A dimensionality reduction process is recommended in most cases.

The dimensionality reduction methodologies included in this framework are linear such as *Principal Component Analysis* (PCA) (Jolliffe 1986), not linear like *Uniform Manifold Approximation and Projection* (UMAP) (McInnes, Healy, and Melville 2020) or *t-distributed stochastic neighbor embedding* (tSNE) (van der Maaten and Hinton 2008), and deep autoencoder Salakhutdinov and Hinton (2009).

The dimensionality reduction can either be performed on each feature set individually or after concatenating all of them. The implications of doing it at different moments are that some feature sets would be overweight or underweight in the final process (if performed after the concatenation) or they would all have equal weights (if performed after the concatenation).

Clustering

Algorithm

Given a corpus of conversation history, the number of intents is unknown, thus density-based clustering algorithms are adopted, as they are the most appropriate for this task. The algorithm receives a set of points and groups the points together if they are closely packed, meaning that they are close neighbors within a high-density area. In this model, there is no need to know the number of clusters beforehand and the clusters do not need to have a spherical shape, with a radius equal to the distance between the centroid and the furthest data point as opposed to some algorithms such as a K-means. The current work uses a modified version of ITER-DBSCAN as proposed by Chatterjee and Sengupta (2020).

Normalization

Data points might have a high dimensionality and different scales, which complicates tuning the algorithm. For this reason, points are L2-normalized to push them onto the surface of a hyper-sphere in which the maximum distance between two points is 2. After normalization, the distances are bounded in the range [0,2] making the tuning process faster.

Execution

In each iteration of the clustering loop the minimum allowed distance for a density group to be considered a cluster is extended, thus conditions are relaxed. Since the distances between points are bounded in the range [0,2], there is a high level of control over the granularity level of the clusters. When the loop finishes, a silhouette analysis is performed to remove noise: points with negative values are removed from their clusters, and clusters with negative silhouette mean are canceled completely. This means that all their points remain unlabeled. This ensures that points are not forced into clusters where they do not belong in. Finally, as clustering is an iterative process, some clusters might overlap. To address this, a merging process is carried out to merge clusters based on similarity. A greedy algorithm checks whether cluster pair distances are below a given threshold to merge them.

Label Generation

The final step of the process is to generate labels for the extracted clusters. Texts within each cluster share features and words that can be used to infer labels for them. For every cluster, all the texts within are joined to form a unique paragraph. The result is a paragraph per cluster, which is used to generate a TF-IDF matrix. Output rows are normalized using the L2 norm and then the most relevant terms are selected by filtering those above a given threshold. The resulting set of words is used as labels for each cluster.

Experiments

The proposed framework suggests a feature extraction method and a clustering method. The clustering method depends on the output of the feature extraction process. At the same time, feature extraction needs a learning process to generate metrics to understand its performance.

Feature extraction is evaluated in a standalone approach by training a classifier such as random forest on each feature set combination. This way the predictive power of each feature set can be compared against the others. By using Shapley additive explanations (Lundberg and Lee 2017), which is a game-theoretic approach to explain the output of any machine learning model, the contribution of each feature set to the final score can be calculated.

To measure the performance of the proposed clustering algorithm, it is compared against other well-known clustering algorithms, such as DBSCAN, OPTICS (Ankerst et al. 1999) and HDBSCAN. A concatenation of all the feature sets proposed (V_W, V_K, W_P, W_T) is used as the data to be clustered in this test. Some variants in terms of dimensionality reduction will also be applied, to look for that setup that performs better with each clustering algorithm.

Finally, the entire framework is evaluated by looking at the best hyperparameter combination and feature set selection for each dataset.

Metrics

To measure the performance of the feature extraction method, due to the unbalanced nature of the data sets the most appropriate classification metric is F1-Score, which is the harmonic mean of the precision and recall.

Evaluating the performance of a clustering algorithm is not as trivial as counting the number of errors of a supervised classification algorithm. Doing so requires the existence of a ground truth, which is rarely available in real-life problems or requires manual assignment by human annotators.

In this work three publicly available labeled data sets are used. This allows leveraging their labels to measure how well a clustering process performs based on two principles: completeness (each class is contained in fewer clusters) and homogeneity (each cluster contains fewer classes).

V-Measure (Rosenberg and Hirschberg 2007) balances completeness and homogeneity but is not adjusted to randomness. A random clustering process in which there are as many clusters as samples in the dataset will obtain a score above zero, depending on the number of labels in the dataset. To overcome this problem, Adjusted Mutual Information (AMI) (Vinh, Epps, and Bailey 2010) is a mutual information-based metric that normalizes against chance.

Parameter Settings

Our framework relies both on the feature engineering method and on the clustering method to perform well, as they are dependent on each other. Therefore, hyperparameters on both sides must be fine-tuned together. A hyperparameter optimization process is performed to search for the best hyperparameter setup. There are several hyperparameter optimization techniques available (Alibrahim and Ludwig 2021), such as Bayesian Optimization (Mockus and Mockus 1991) or Genetic Algorithms (Di Francescomarino et al. 2018). For practical reasons, in this work Bayesian Optimization is the chosen approach. The decision of whether to select a certain feature set or not is also considered a hyperparameter in this optimization process.

Data

To evaluate the effectiveness of our model in different contexts, experiments are conducted on three publicly available data sets, namely the ATIS dataset (Hemphill, Godfrey, and Doddington 1990), Banking (Casanueva et al. 2020) and SNIPS (Coucke et al. 2018).

ATIS is a non-balanced dataset consisting of 4454 observations and 16 classes, having a single class dominating the rest, with over 74% of occurrences, and 2 classes with less than 7 samples, which are difficult to cluster. The Banking dataset includes 10,003 examples and 77 classes. Classes are not balanced but there is not a class that dominates the rest. SNIPS is a balanced dataset that contains 13,084 and 7 classes.

Results

The no-free lunch theorem is validated in our experiments. An optimal configuration that works in all cases does not exist, but rather the selection of feature sets and the hyperparameters change from problem to problem.

Feature Engineering

A feature set is considered of high quality when it represents accurately the relations within the data. The texts or sentences representing similar ideas or belonging to the same class have vectors that are close to each other, and at the same time, are far away from those belonging to different ideas or classes.

Feature sets have been tested in a standalone approach and also as part of the entire framework. An example of the results of the standalone testing can be seen in Figure 2. The contribution of each feature set to the F1-Score

when using all of them is calculated using Shapley additive explanations and it is shown in Table 1. Semantics feature set V_W has the most predictive power

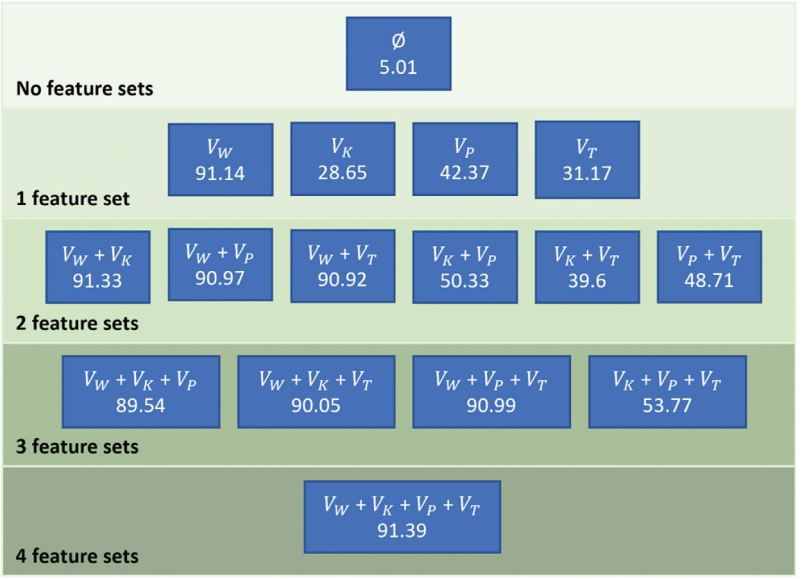

Figure 2. F1-score calculated using random forest on ATIS dataset for all feature sets combinations.

Table 1. Contribution of each feature set to the F1-score in absolute values. V_W outperforms all the other feature sets.

	F1-Score			Contribution			
	Baseline	All Features	Gain	*VW*	*VK*	V_P	V_T
SNIPS	14.28	99.25	84.97	**31.39**	20.27	22.07	21.88
ATIS	5.01	91.39	86.30	**56.18**	13.96	7.62	8.60
Banking	0.04	96.63	96.59	**52.18**	12.54	14.85	17.00

Table 2. F1-score comparison on the different ways of calculating V_W. fine tuned models outperforms pretrained models. Max pooling and CLS token yield better result than mean pooling.

	Pretrained			Fine Tuned		
	CLS	Max	Mean	CLS	Max	Mean
SNIPS	96.30	95.11	96.38	99.24	**99.27**	93.82
ATIS	37.40	35.80	40.45	**91.14**	88.43	89.68
Banking	73.86	70.62	74.84	96.64	**96.69**	79.13

among all the feature sets for all evaluated data sets. In fact, any feature set combination which excludes feature set V_W yields worse results. Based on these results, feature set V_W should be considered as the backbone of any intent mining process, and the rest of the feature sets should work as a complement to it. Taking a closer look at the different ways of producing feature set V_W, the fined tuned model using a siamese training approach performs better than a vanilla pre-trained BERT model, as expected. Regarding the pooling approach, using the CLS token or max-pooling yields similar results and outperforms mean pooling. However, this is not true when using the pre-trained model, in this case using a mean pooling produces better results, as can be seen in Table 2. Mean pooling is safer than max-pooling because it smooths out the output but misses specific information, so it works better on models not fit to the data. A fine-tuning process can specialize the model on the given data, so in this case, the usage of max pooling can collect more meaningful features.

Clustering

The clustering process is in charge of identifying relevant groups of data points and labeling them accordingly. This task becomes easier if such groups of data points are compact and have clear boundaries. Nevertheless, the iterative nature of the algorithm makes it possible to identify such groups.

Clustering algorithm work by calculating distances. When confronting the proposed iterative clustering method with other existing algorithms, this generally outperforms the rest. This is in fact expected due to the iterative nature of the algorithm, but the trade-off is more computational time. Depending on the complexity of the data set, the performance difference

Table 3. Clustering comparison. Iterative clustering method with other existing algorithms generally outperforms the rest. Depending on the complexity of the dataset, the performance difference varies. Simpler datasets such as SNIPS do not show any significant difference. Therefore, depending on the problem complexity, it might make sense to use one or another.

	DBSCAN		OPTICS		HDBSCAN		Iterative (Ours)	
	AMI	V-Msr	AMI	V-Msr	AMI	V-Msr	AMI	V-Msr
SNIPS	0.9721	0.9721	0.9746	0.9746	0.9591	0.9592	**0.9752**	**0.9753**
ATIS	0.9000	0.9015	0.8893	0.8906	0.8196	0.8219	**0.9200**	**0.9210**
Banking	0.9155	0.9216	0.9266	0.9319	0.8998	0.9072	**0.9285**	**0.9334**

varies. Simpler data sets such as SNIPS do not show any difference. Therefore, depending on the problem complexity, it might make sense to use one or another. The results of the clustering comparison for each dataset can be observed in Table 3.

The results of the entire framework hyperparameter optimizations can be seen in Table 4. It is to be noted that the hyperparameter optimization process aimed to maximize AMI score and no other metrics such as the number of clusters found. The SNIPS dataset is not considered a challenging problem, based on the balanced nature of the dataset and the reduced number of classes. The best setup is to use V_W (with max-pooling), V_P, and V_T. Dimensionality reduction is applied using UMAP and a number of dimensions of 120. The clustering algorithm runs for 8 iterations, with a merge threshold of 0.0399. All 7 classes were successfully discovered, and the number of labeled samples was 100%. AMI score is 0.9782, outperforming all the previous standalone tests. Regarding ATIS, the existence of a class that heavily dominates the data set makes it more challenging. In this case, two setups have been selected, one maximizing AMI and the other optimizing the number of clusters found. The feature sets selected varies from one approach to the other, but what does not change is the use of UMAP for dimensionality reduction. Finally, the Banking data set represents a problem in which there is a large number of classes, it is unbalanced and some of them have a small number of samples. In this case, V_W (with CLS pooling), V_P, and V_K are the selected feature sets. Dimensionality reduction using UMAP is also the best option. AMI score resulted in 0.9287 and 68 classes were found, covering 100% of the samples. All

Table 4. Framework results. *Features* columns indicate the feature sets that were selected. For V_W it indicates the pooling approach. *Dim. Reduction* and *clustering* columns indicate the methods and hyperparameters used in the dimensionality reduction and clustering steps, respectively. *Results* column summarizes AMI and V-measure score, number of clusters found and percentage of points assigned a label.

	Features				Clustering			Results		
	VW	*VK*	V_P	V_T	Iters	Thres.	AMI	V-Msr	Clusters	Label %
SNIPS	Max	-	OK	OK	8	0.039	0.9782	0.9783	7/7	100%
ATIS$_{AMI}$	Max	-	OK	OK	6	0.077	0.9210	0.9220	9/16	99.62%
ATIS*nclusters*	CLS	OK	OK	OK	8	0.022	0.8111	0.8145	16/16	97.4%
Banking	CLS	OK	OK	-	9	0.010	0.9287	0.9335	68/77	100%

Table 5. Generated labels for SNIPS dataset.

Original Class	Generated Label
AddToPlaylist	add playlist
PlayMusic	music play
GetWeather	forecast weather
RateBook	points rate
BookRestaurant	book restaurant table
SearchCreativeWork	called tv
SearchScreeningEvent	playing schedule theatres

those results remark that there is not a fixed configuration or hyperparameter setup that works best in all cases, but instead varies from problem to problem.

Generated Labels

The generated labels for the SNIPS dataset are presented in Table 5. It can be observed that the automatically generated labels are a great approximation to the original class labels. Some of them can be considered an exact match, like *add playlist* or *music play* that match *AddToPlaylist* and *PlayMusic* respectively. Examples such as *forecast weather*, *points rate* and *book restaurant table* contain at least one of the keywords existing in the original class name. For class *SearchScreeningEvent*, the proposed label does not contain any of the words in the class name, but it could be argued that the words *playing schedule theaters* are related to the actual intent, which is, in this case, to look for the schedule of a certain play. On Table 6 the generated labels for the top 7 intents in ATIS dataset are presented. The intents are either an exact match, such as *flight* for *atis_flight*, or have a straightforward interpretation like in *does code fare mean* for *atis abbreviation*, which refers to a common question posed by airlines customers.

Final Discussion

In this work, an automated end-to-end framework has been proposed to find intents from unlabeled conversational data and also to generate an initially labeled dataset that will work as a starting point for building an intent classifier. Results show that the generated labels for the clusters found are a good approximation to the actual labels.

Table 6. Generated labels for ATIS dataset.

Original Class	Generated Label
atis flight	flight
atis airfare	fare round trip
atis ground service	ground transport
atis airline	airline flight
atis abbreviation	does code fare mean
atis aircraft	aircraft type use
atis flight time	flight schedule time

In practice, this framework could speed up the development of projects such as conversational agents or chatbots, by reducing the time required to analyze and extract value from large volumes of unlabeled data, which is a common scenario in large enterprises. This framework could also eliminate, or at least reduce by a significant factor, the need for human labelers, which represent an important component when calculating the economic impact of a project.

The main limitations of this framework can be found in the common computational and memory limitations of density-based clustering algorithms, in which requirements grow exponentially with the number of samples to be clustered. Therefore, when working with enormous volumes of data, it will require working with a sample of the data instead of the entire dataset. To address this limitation, this framework could be extended in future work by adding a label propagation layer. This way, the generated labels can be extended to the rest of the target dataset at a small cost.

Another limitation of this study lies in the interpretation of the results. It is up to the user to check if the proposed clusters and labels are fit for the intended purpose. The way data is clustered is based on semantic, lexical, and syntactic similarity, which might not follow the same subjective criteria the user might be looking for. For instance, the user might want data to be clustered based on which would be the best downstream task agent or processor to be sent to, and not based on the semantic, lexical, and syntactical similarity.

Conclusion

The proposed framework can tackle the intent mining problem end to end. It sets a sequence of steps to be performed that extracts linguistic features from texts in different ways and finds clusters within the data representing the intents. The framework also produces labels automatically and without the need for human intervention that can be used as the intent labels.

There is not a single configuration that works best in all cases, even though there are some patterns that perform better in most cases. Features generated by transformer models have the most predictive power, being the other feature sets a good complement for it. However, one has to fine-tune a transformer model to get the best performance out of it. Choosing the right dimensionality reduction method is important but UMAP showed to work better than the rest, at least for the data set tested in this work. Finally, using an iterative version of the density-based clustering algorithm works best at capturing clusters of different sizes and shapes.

Future research lines would include an extension to the current work by adding a label propagation layer that assigns the generated labels to a larger dataset. Another improvement could be the extension of the cardinality of the

least represented clusters found by generating synthetic data that would still fall within the same cluster to facilitate the training of the intent classifier. Finally, this framework deals with mining intents but could also be extended to identify the relevant entities within the data.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Alibrahim, H., and S. A. Ludwig. 2021. Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization. In *2021 IEEE congress on evolutionary computation (cec)*, 1551–59. doi: 10.1109/CEC45853.2021.9504761

Ankerst, M., M. M. Breunig, H. -P. Kriegel, and J. Sander. 1999. Optics: Ordering points to identify the clustering structure. In *Proceedings of the 1999 acm sigmod international conference on management of data*, 49–60. New York, NY, USA: Association for Computing Machinery. Retrieved from doi: 10.1145/304182.304187

Casanueva, I., T. Temcinas, D. Gerz, M. Henderson, and I. Vulic. 2020, marth). Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd workshop on nlp for convai - acl 2020*, Retrieved from doi: 10.48550/arXiv.2003.04807

Chatterjee, A., and S. Sengupta (2020, December). Intent mining from past conversations for conversational agent. In *Proceedings of the 28th international conference on computational linguistics*, 4140–52. Barcelona, Spain (Online): International Committee on Computational Linguistics. Retrieved from doi: 10.18653/v1/2020.coling-main.366

Coucke, A., A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril, et al. 2018. Snips voice platform: An embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, *abs/1805.10190*. Retrieved from http://arxiv.org/abs/1805.10190

Deloitte. 2019. *What is conversational ai?* Retrieved from https://www2.deloitte.com/content/dam/Deloitte/au/Documents/strategy/au-deloitte-conversational-ai.pdf

Devlin, J., M. -W. Chang, K. Lee, and K. Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*, ed. J. Burstein, C. Doran, and T. Solorio, 4171–4186. Minneapolis, MN, USA: Association for Computational Linguistics.

Di Francescomarino, C., M. Dumas, M. Federici, C. Ghidini, F. M. Maggi, W. Rizzi, and L. Simonetto. 2018. Genetic algorithms for hyperparameter optimization in predictive business process monitoring. *Information Systems* 74:67–83. Retrieved from (Information Systems Engineering: selected papers from CAiSE 2016) doi:https://doi.org/10.1016/j.is.2018.01.003.

Ester, M., H. -P. Kriegel, J. Sander, and X. Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the second international conference on knowledge discovery and data mining*, 226–31. Portland, OR: AAAI Press.

Goyal, A. K., A. Metallinou, and S. Matsoukas. (2018, June). 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*, vol *3 (industry papers)* 145–52. New Orleans - Louisiana: Association for Computational Linguistics. Retrieved from doi: 10.18653/v1/N18-3018

Haponchyk, I., A. Uva, S. Yu, O. Uryupina, and A. Moschitti. 2018, October-November. Supervised clustering of questions into intents for dialog system applications. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, 2310–21. Brussels, Belgium: Association for Computational Linguistics. Retrieved from doi: 10.18653/v1/D18-1254

Hemphill, C. T., J. J. Godfrey, and G. R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Speech and natural language: Proceedings of a workshop held at hidden valley, Pennsylvania*, june 24-27, 1990. Retrieved from https://www.aclweb.org/anthology/H90-1021

Jolliffe, I. T. 1986. Principal components in regression analysis. In *Principal component analysis*, ed. I. T. Jolliffe, 129–155. New York, NY: Springer.

Kelley, J. F. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems* 2 (1):26–41. doi:10.1145/357417.357420.

Lloyd, S. P. 1982. Least squares quantization in pcm. *IEEE Transactions on Information Theory* 28 (2):129–36. doi:10.1109/TIT.1982.1056489.

Lundberg, S., and S. Lee 2017. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874. Retrieved from http://arxiv.org/abs/1705.07874

Mallinar, N., A. Shah, R. Ugrani, A. Gupta, M. Gurusankar, T. K. Ho ..., and B. McGregor.2018. Bootstrapping conversational agents with weak supervision. arXivRetrieved from. 10.48550/ARXIV.1812.06176.

McInnes, L., J. Healy, and S. Astels. 2017, march. Hdbscan: Hierarchical density based clustering. *Journal of Open Source Software* 2(11):205. doi: 10.21105/joss.00205.

McInnes, L., J. Healy, and J. Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv.

Mockus, J. B., and L. J. Mockus. 1991, July. Bayesian approach to global optimization and application to multiobjective and constrained problems. *Journal of Optimization Theory and Applications* 70(1):157–72. doi: 10.1007/BF00940509.

Reimers, N., and I. Gurevych. 2019, November. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing*, Association for Computational Linguistics. Retrieved from https://arxiv.org/abs/1908.10084

Research, & Markets. 2021. *Global conversational ai market (2021-2026)*. Retrieved from https://www.researchandmarkets.com/reports/5359446/global-conversational-ai-market-2021-2026-by

Rosenberg, A., and J. Hirschberg. 2007, June. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 410–20. Prague, Czech Republic: Association for Computational Linguistics. Retrieved from https://www.aclweb.org/anthology/D07-1043

Salakhutdinov, R., and G. Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning* 50 (7):969–78. doi:10.1016/j.ijar.2008.11.006.

Settles, B. 2009. *Active learning literature survey*. 1648, University of Wisconsin-Madison Department of Computer Sciences, Madison, WI.

Shi, C., Q. Chen, L. Sha, S. Li, X. Sun, H. Wang, and L. Zhang. 2018, October-November. Auto-dialabel: Labeling dialogue data with unsupervised learning. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, 684–89. Brussels, Belgium: Association for Computational Linguistics. Retrieved from doi: 10.18653/v1/D18-1072

Thakur, N., N. Reimers, J. Daxenberger, and I. Gurevych. 2021. Augmented sbert: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. arXiv.

Thomas, P. 2009, January. Semi-supervised learning by Olivier Chapelle, Bernhard sch¨olkopf, and alexander zien (review). *IEEE Transactions on Neural Networks* 20(3):542. doi: 10.1109/TNN.2009.2015974.

van der Maaten, L., and G. Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research* 9 (86):2579–605. Retrieved from. http://jmlr.org/papers/v9/vandermaaten08a.html.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *NIPS 2017*, Vol. 30. Long Beach, CA, USA: Curran Associates, Inc.

Vinh, N. X., J. Epps, and J. Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research* 11 (95):2837–54. Retrieved from. http://jmlr.org/papers/v11/vinh10a.html.

Wen, T. -H., D. Vandyke, N. Mrkˇsi´c, M. Gaˇsi´c, L. M. Rojas-Barahona, P. -H. Su, Utlas, S. and S. Young. 2017, April. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: Volume 1, long papers*, 438–49. Valencia, Spain: Association for Computational Linguistics. Retrieved from https://www.aclweb.org/anthology/E17-1042

Williams, J., N. Niraula, P. Dasigi, A. Lakshmiratan, C. Suarez, M. Reddy, and G. Zweig. 2015 January. Rapidly scaling dialog systems with interactive learning. *Rapidly Scaling Dialog Systems with Interactive Learning* 1–13. doi:10.1007/978-3-319-19291-81.