Intent Clustering from Conversations for Task-Oriented Dialogue

Tian Jin
NYU Shanghai
tj1059@nyu.edu

Tang Sheng
NYU Shanghai
ts3906@nyu.edu

Penghao Weng
NYU Shanghai
pw1298@nyu.edu

Abstract

A reliable intent clustering algorithm for task-oriented dialogues contributes to precise detection of speakers' intents, therefore allowing more comprehensive applications in different domains. Because of the high cost of time and expense and the numerous emerging domains in need of such application, it is challenging to make and generalize progress in this area to various tasks and fields. In this paper, given an insurance-related customer support dataset, we propose a GAT-based Sentence Transformer Decoding Model to encode dialogue sentences, and introduce a cross-validation fixing mechanism for the purpose of correcting those wrongly-clustered utterances after applying the K-Means algorithm. The proposed model has been verified effective in improving clustering accuracy through experiments, and our next focus is to evaluate the performance of applying cross-validation to the enhanced clustering results.

I Introduction

Task-oriented dialogue modeling has drawn great attention for its widespread use in fields applying conversational systems (primarily virtual assistants), especially in business-related ones with great demand for customer service interactions between virtual agents and customers. The capability of precisely identifying the intent of each interaction (mostly customers' requests) therefore has for long become the centerpiece and has been widely researched in recent years. Considering the great cost to obtain well-annotated data for training, previous work has gained traction looking at slots or dialogue states given conversational logs (Chatterjee and Sengupta, 2020), (Hudecek et al., 2021). However, as novel services and domains continue to emerge, a lack of shared benchmarks hinders the generalization of progress in this area to various tasks and fields.

In this paper, we specifically look at an insurance-related customer support dataset, creating a set of intent labels based on conversations and assigning one to each of the utterances with semantic intent. We first construct a GAT-based Sentence Transformer Decoding Model by implementing techniques including Encoder-Decoder architecture, Graph Attention Network, BERT Generation Decoder Model framework, in order to encoding the utterances with as much information as possible. Then, for those wrongly-clustered utterances, we introduce the cross-validation fixing mechanism for the purpose of enhance the model performance.

II Related Work

Intent induction has been intensively studied in the past several years. While virtual assistants are introduced in a vast number of fields in the real world, obtaining data of high quality with accurately-assigned labels for model training requires a great cost of both time and money. Therefore, one direction of progress in this field points to dealing with low data sources. (Goyal et al., 2018) explores transfer learning to reuse available chat logs instead of introducing new well-developed datasets, yet it fundamentally assumes that the utterances are all pre-labeled beforehand and looks at the intersection of deep learning and transfer learning from a supervised learning perspective. Similarly, (Chatterjee and Sengupta, 2020) proposes a novel method to create high-quality training data given the carefully reviewed and labeled clusters, regardless of its unsupervised method to mine utterance intents in the first round.

On the other hand, semi-supervised and unsupervised methods are also well investigated for better classify the utterances by their respective intent. (Yang et al., 2014) compute the similarity between dialogue sentences based on vector-space representation and seed the clusters through semi-supervised approach on manually annotated data, while (Lv et al., 2021), specifically in the field of task-oriented dialogues, has even proposed Dialogue Task Clustering Network recently in an effort to enhance encoding from an unsupervised learning aspect.

III Self-Supervised Approach

III.1 Baseline

The baseline of the intent clustering from the conversation for task-oriented dialogue basically contains two parts: 1) encoding the utterances with intents from given dialogues using a Sentence Transformer model to a vector representation and then, 2) clustering the vector representations by built-in clustering algorithms from the scikit-learn library.

To facilitate the performance of the intent clustering baseline architecture, we need to figure out 1) how to get better vector representation for each utterance for clustering and, 2) how to classify the utterances correctly by different algorithms.

III.2 Encoder-Decoder

For better vector representations, we introduce an Encoder-Decoder architecture to tune the performance of the original sentence transformers. We obtain the idea from the Frame of Dialogue Task Clustering Network (DTCN) designed by (Lv et al., 2021) in paper *Task-Oriented Clustering for Dialogues*.

In DTCN, (Lv et al., 2021) attempt to promote the vector representation of each utterance by an Encoder-Decoder architecture. To gain more information through the contexts of each utterance, the authors use Graph Attention Networks (GAT) (Velickovic et al., 2018) to get an enhanced version of the sentence transformers' output for better utterance clustering results which later serve for the dialogue clustering task.

From the inspiration of DTCN, we design a GAT-based Sentence Transformer Decoding Model to fine-tune the representation vectors returned from the original sentence transformers. Detailed implementation will be discussed in section IV.

IV GAT-based Sentence Transformer Decoding Model

The GAT-based Sentence Transformer Decoding Model mainly contains two components: 1) a sentence transformer model with a Graph Attention Network as the Encoder component and 2) a framework of the BERT Generation Decoder Model (Devlin et al., 2019) as the Decoder component.

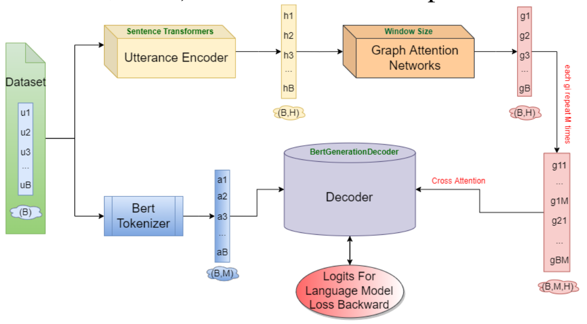

Figure 1: GAT-Based Sentence Transformer Decoding

-

To illustrate the details of the model, we take one batch from the training process as an example. This model design is shown using a batch example, which is demonstrated in Figure 1 above.

IV.1 Sentence Transformer Raw Hidden States

In one particular batch with batch size B, there is an utterance set for each utterance $u = \{u_t\}_{t=1}^{B}$. Getting the original vector representation of u_i through a sentence transformer model we obtained the output hidden states h with size H where $h = \{h_t\}_{t=1}^{B}$.

IV.2 GAT-Enhanced Hidden States

After getting the raw hidden states h from the sentence transformer model, we enhance the hidden state through a Graph Attention Network. In order to implement GAT, we build up an adjacency graph $G = (V, E)$ for the utterances in all dialogues.

Each dialogue $d_i \in \{d_t\}_{t=1}^{K}$ which contains n_i utterances represents as $d_i = \{u_t\}_{t=1}^{n_i}$, and each utterance u_i serves as an node in the node set:

$$V = \{u_i\}_{i=1}^{N}$$

where $N = \sum_{i=1}^{K} n_i$.

We set up the Edge set E from the adjacency relationship from each utterance. To specify, we define a Window Size W and obtain the edges for each utterance from one specific dialogue in the following equation:

$$e_{ij} = \begin{cases} 1, |j - i| \leq W \\ 0, Otherwise \end{cases}$$

We feed the adjacency graph G along with the sentence transformer output hidden state of the batch discussed in section IV.1 where $h = \{h_t\}_{t=1}^{B}$ into the Graph Attention Network to obtain the enhanced hidden states from the contextual information:

$$g = GAT(G, h)$$

We now have the enhanced hidden state g with size H for this batch where $g = \{g_t\}_{t=1}^{B}$.

IV.3 BERT Generation Decoder

In the Decoder component, we basically utilize the framework of the BERT Generation Decoder Model from Hugging face, where the hidden size and the intermediate size of the model config file need to be matched both the hidden size and the intermediate size of the state dictionary with the first layer module (a BERT model) of the chosen sentence transformer. We initiate the decoder embedding using the state dictionary.

Then we tokenize the utterances in the batch mentioned in section IV.1 by setting the maximum sequence length to M in order to save the GPU memory for training. Now we have the input ids as the decoder input with size $(B \times M)$. In order to make cross attention with the corresponding GAT output g during the decoding process, we replicate each $g_i \in g$ by M times. We obtain the encoder input g_{rep} with size $(B \times M \times H)$ so that we can feed it into the decoder as the encoder input ids.

Now we can use this language model to make the next-word prediction task where the labels are the one token left-shift of the decoder input ids. After getting the probability distribution of the next word and the ground truth label, we apply cross-entropy loss and make the loss backward to better learn the contextual representation.

V Semi-Supervised Approach

V.1 Idea

Pure self-supervised learning is limited in downstream tasks in that it lacks some significant information including the ground-truth labels of each utterance. As a result, the performance of the self-supervised approach is not satisfying enough if it is placed in competition with supervised learning. However, having experts label every utterance in one dataset is extremely expensive. A semi-supervised strategy can be taken to improve the performance of the model on top of the pure self-supervised one without an expensive and time-consuming labeling process.

V.2 Architecture

This model architecture is designed on top of the GAT-based encoder-decoder model (see figure1). One multi-class classifier and one additional training object are introduced. A hyper-parameter, identifying probability α, controls the proportion of intended utterances that will be identified with ground-truth intents. Other utterances' intents will be labeled as unknown and ignored by the classifier. In other words, $(1-\alpha) * len(intendedutterances)$ intended utterances and all the utterances without intents will be labeled as -100 (to be ignored); $alpha * len(intendedutterances)$ intended utterances will be labeled with their actual intents. We feed the utterances into the GAT-based Encoder and get the GAT-enhanced hidden states of each utterance g_i. As before, we feed g_i to the decoder and get $loss_i^{decoder}$ with a cross-entropy loss function. Additionally, we feed g_i into a classifier head to get $loss_i^{classifier}$.

$$logits_i = W_{classifier} g_i$$

Then we take a cross-entropy approach to get $loss_i^{classifier}$ according to the ground-truth labels and the identifying probability α.

$$loss_i^{classifier} = \begin{cases} 0, y_i = -100 \\ CrossEntropy(logits_i, y_i), Otherwise \end{cases}$$

Combining the two losses we gained with a hyper-parameter λ to be the coefficient, we get the object that we want to optimize.

$$L_i = loss_i^{classifier} + \lambda * loss_i^{decoder}$$

VI Cross-Validation Fixing

VI.1 Idea

In each approach that we have discussed above, we essentially get a hidden state g_i for each intended utterance u_i. Then, we use K-Means algorithm (Hartigan and Wong, 1979) to cluster all the hidden states in a linear approach. However, the feature of linearity can lead to some problematic clustering results as some intended utterances with different intent labels may appear indeed very close to each other in the virtual clustering space. For situations like this, we introduced a Cross-Validation (Wang, 2010) Fixing mechanism to fix some wrongly-clustered intended utterances and assist them to return to the cluster they ought to belong to.

VI.2 Implementation

We essentially take the traditional k-fold cross-validation approach. After the K-Means clustering process, we will get the predicted label c_i (i.e. clustering label) for each intended utterance u_i. We reference c_i as the sudo-label we will later use as the ground-truth label for the classification task. We set $k = 10$. The intended utterances will be randomly split into 10 folds. Each time we define 9 folds as the training set and 1 fold as the testing set. We use the training set to train a BERT classifier with their sudo-labels as the ground-truth object.

$$L_i = CrossEntropy(Classifier(u_i), c_i)$$

Then we use the trained BERT classifier to predict labels for the utterances in the testing set. We will get the $logits_i$ for each tested utterance u_j. Then,

$$c_i = argmax(Softmax(logits_i))$$

Therefore, the sudo-labels for the tested utterances will be changed into a new one if the classifier is not confident enough with the current one. After one round for the k-fold cross-validation is done, we will be able to update all the sudo-labels for the intended utterances and take the updated sudo-labels as the new clustering label.

VII Detailed Experiments

VII.1 Query-Response Concatenation

On top of the raw data set provided by DSTC11, we extracted more information for every intended utterance. Here, let's define an intended utterance as q_i for clarity. For each q_i, we search through the dialogue it belongs to. We get the vector representation of each sentence aside from the intended utterance itself in this dialogue $x_j, j \neq i$. Then we get the similarity score between q_i and each x_j.

$$\cos(\mathbf{t}, \mathbf{e}) = \frac{\mathbf{te}}{\|\mathbf{t}\|\|\mathbf{e}\|} = \frac{\sum_{i=1}^{n} \mathbf{t}_i \mathbf{e}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{t}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{e}_i)^2}}$$

,

$$Score(i, j) = 100 * (cos(q_i, x_j) + 1)/2$$

, After getting all the similarity scores, we choose the x_j that has the best similarity score with q_i.

$$m = argmax_j Score(i, j)$$

We define $r_i = x_m$ and concatenate q_i and r_i to get a information-enhanced hidden states of the intended utterance u_i: $h_i = q_i \oplus r_i$.

VII.2 Model Concatenation

In order to gain more information from different outputs of different sentence transformers, we approach making a concatenation of the vectors. We try a two-pair concatenation among six different sentence transformers using the query-response pairs data set and most of the combinations show improvements. Detailed experiment results are shown in Table 1 and Table 2.

Model Name (Label)
all-MiniLM-L12-v2 (1)
multi-qa-MiniLM-L6-cos-v1 (2)
paraphrase-MiniLM-L6-v2 (3)
all-MiniLM-L6-v2 (4)
multi-qa-mpnet-base-dot-v1 (5)
paraphrase-multilingual-MiniLM-L12-v2 (6)

Table 1: Model Labels

Label	Single F1 Score	Best Concat F1 Score
1	60.8	65.3 (with 2)
2	60.5	65.3 (with 1)
3	55.6	62.7 (with 5)
4	58.4	63.4 (with 4)
5	55.1	62.7 (with 3)
6	56.8	62.4 (with 5)

Table 2: Concatenation Trials

VIII Results

Overall, we observe a great improvement in applying our proposed approach on the test set. Among all six sentence transformer models, all-MiniLM-L12-v2 and multi-qa-MiniLM-L6-cos-v1 have a greater performance compared to other four, reaching F1 Scores of over 60. When experimenting with query-response concatenation and two-pair model concatenation, we achieve great improvements in every experiment, in which F1 Score was boosted from 55.1 to 62.7 when concatenating multi-qa-mpnet-base-dot-v1 and paraphrase-MiniLM-L6-v2. The concatenation of all-MiniLM-L12-v2 and multi-qa-MiniLM-L6-cos-v1 achieves the best performance as it hits the F1 Score of 65.3. Although our proposed approaches are still to be experimented, the primary results strongly verify our assumptions and indicate the potentials for future application.

IX Conclusion

This paper proposes a GAT-based Sentence Transformer Decoding Model for dialogue utterances encoding, and introduces a cross-validation fixing mechanism in the attempt to help correct those wrongly-clustered utterances after applying the K-Means algorithm. Due to the time limit, we haven't apply the proposed fixing mechanism to our proposed model. As our experiments at the current stage already demonstrate the capability of the GAT-based Sentence Transformer Decoding Model in improving accuracy of clustering dialogue sentences by their semantic intents, and our next step is to evaluate the effectiveness of cross-validation fixing mechanism based on the optimal concatenated model (in Section VIII).

References

Ajay Chatterjee and Shubhashis Sengupta. 2020. Intent mining from past conversations for conversational agent. *ArXiv*, abs/2005.11014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Anuj Goyal, Angeliki Metallinou, and Spyridon Matsoukas. 2018. Fast and scalable expansion of natural language understanding functionality for intelligent agents.

J. A. Hartigan and M. A. Wong. 1979. Algorithm AS 136: A K-Means clustering algorithm. *Applied Statistics*, 28(1):100–108.

Vojtech Hudecek, Ondrej Dusek, and Zhou Yu. 2021. Discovering dialogue slots with weak supervision. In *ACL*.

Chenxu Lv, Hengtong Lu, Shuyu Lei, Huixing Jiang, Wei Wu, Caixia Yuan, and Xiaojie Wang. 2021. Task-oriented clustering for dialogues. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4338–4347, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio', and Yoshua Bengio. 2018. Graph attention networks. *ArXiv*, abs/1710.10903.

Junhui Wang. 2010. Consistent selection of the number of clusters via crossvalidation. *Biometrika*, 97:893–904.

Xiaohao Yang, Jia Liu, Zhenfeng Chen, and Weilan Wu. 2014. Semi-supervised learning of dialogue acts using sentence similarity based on word embeddings. *2014 International Conference on Audio, Language and Image Processing*, pages 882–886.