# JOINT TRAINING OF CLASSIFICATION MODEL AND SIMILARITY MODEL FOR LOW-RESOURCE TEXT CLASSIFICATION IN CHATBOT

*Jingwen Huang*

{hanmei613}@gmail.com

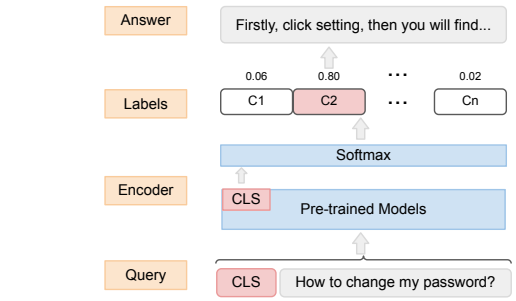
## ABSTRACT

Building conversational chatbot system has become a popular solution to customer services under various business scenarios. A conversational chatbot needs to detect user's intent given a few words, which essentially equals to short-text classification problem in the field of Natural Language Processing. Moreover, each time for a new service, the task-specific chatbot system often needs to perform well in few-shot setups due to lack of domain-specific data, which is quite a challenge for single-model system such as text classification model system or sentence-pair semantic similarity model system under such a low-resource condition. Therefore, in this paper, we propose SFC, a 2-stage joint system with multi-task training technique for both text classification task and sentence-pair semantic similarity task to overcome this challenge. Furthermore, we additionally propose an improved version of SFC to allow text classification model and sentence-pair model be combined into a joint model organized in hierarchical structure. We also conduct extensive experiments on 4 public and 1 private datasets in few-shot setup (i.e., with only 5 to 20 training data per class). The experimental results show that our system can steadily outperform several competitive single-model baselines by 2 percent in average accuracy.

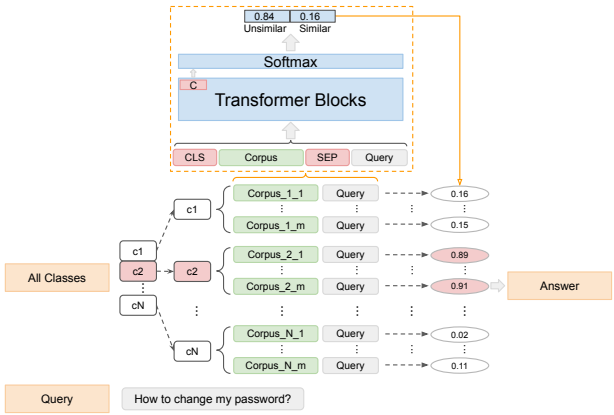
## 1. INTRODUCTION

Task-specific conversational chatbot[1] are designed to share the working pressure of human customer service agents who are responsible for solving customers' questions or queries about certain products or services. Regardless of whether the conversation is single-turn or multi-turn, the essential technical solution behind the chatbot is intent classification model which can identify the correct intent behind user's input text and find the corresponding answer.

Although task-specific conversational chatbot already has a wide application in various business and industries, it's still a quite challenging task due to its natural properties of low-resource. First, customers' utterance within a conversation is usually quite short and completed in a few words. Short text[2] are generally more ambiguous in comparison with long texts such as paragraphs or documents since they don't contain enough contextual information, which poses a great challenge[3] for classification task[4, 5, 6]. Second, at the initial stage of building a chatbot for specific task or service, it's often extremely hard to collect sufficient data samples for each class. The reason is that it's usually too expensive and time consuming to extract different ways of natural language expression for each intent class from history conversation log or even manually compose the corpus from nothing. Therefore, the key to building a task-specific chatbot with high performance becomes solving the challenge of short-text classification[7] problem under few-shot setting[8].




(a) classification model




(b) similarity model

**Fig. 1**. Two popular structures of building task-specific conversational chatbot

One of the most popular approach among existing work was text classification model based on neural network. Since the length of the text is quite short, many previous work[1] choose neural network such as convolutional neural networks

(CNNs)[9, 10, 11] or long short term memory networks (LSTMs)[12, 13] to accomplish the task of extracting semantic feature from limited amount of words. A common model structure is adding a softmax classifier to the top of the neural network. Another common approach is text classification neural network based on label-word joint embedding, such as LEAM[14]. This approach involves label embedding when constructing the text-sequence representation, which poses requirements on label information. In our cases, since we have large amount of class labels without being well-defined, this approach can do only a few help to the low-resource chatbot scenario. Afterwards, pre-trained language models on large corpus like BERT[15] and RoBERTa[16] has been proven more powerful in solving many NLP tasks including short-text classification[17]. Especially for few-shot scenarios[8], pre-trained model based on transformers[18], shown in Fig. 1(a), tends to do more help to reducing the negative effects brought by scarcity of training data.

Another popular approach among previous work was based on sentence-pair model. The motivation of this approach was started from the idea of Information Retrival (IR) based chatbot[19, 20]. Having a Q-A (Question-Answer) pairs dataset and user query Q, the IR based conversational system will look up in the Q-A dataset for the pair (Q', A') that best matches query Q through semantic analysis and returns A' as the answer to Q[21]. In this way, sentence-pair model pre-trained on large corpus of semantic similarity identification task can be applied for being used as a tool to identify the class with highest semantic similarity to customer's query. A common model structure of pre-trained[15] sentence-pair model is based on multiple cross-attention mechanism[22], shown in Fig. 1(b). Due to the fact that many experiment results have shown that RoBERTa[16] is an improved version of BERT, and has achieved amazing results on both text classification and sentence-pairs semantic similarity tasks, we choose RoBERTa as an important baseline approach in this paper.

Despite the success of these 2 approaches, they still have some limitations in task-specific chatbot scenario. As for the text classification model approach, it's quite hard to use data augmentation method to solve the data insufficiency challenge since the it's usually unfeasible to get large amount of domain-specific data when facing a new task. With respect to the sentence-pair model approach, though we can obtain large amount of data in semantically duplicate sentence pair identification domain for transfer learning[23], it's still quite hard to perform well on task-specific dataset because the training objective of sentence-pair model is semantic similarity, which is slightly different from the target of classification task. That is to say, there always exists some intent classes which cannot be distinguished by each other merely depending on semantic similarity. For example, we have 2 user query saying, A: What should I do if I want to change my password? B: What is the modification rule if I want to change my password? In this case, sentence A is semantically similar to B, since they both express the desire for changing password. However, it's still reasonable to classify them into 2 intent classes, since A is asking for the method for changing password, while B is asking for the rule to follow when creating a new password.

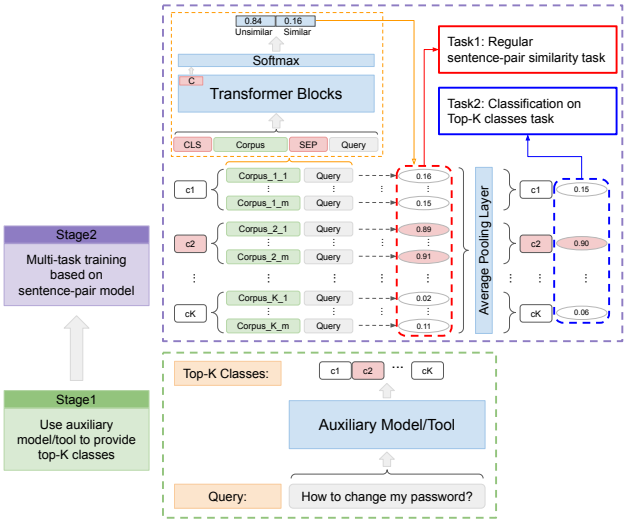



**Fig. 2. Network Structure of 2-stage SFC:** a joint system of multi-task for both text classification task and sentence-pair semantic similarity task.

The above limitation motivates us to propose a joint system of both text classification task and sentence-pair semantic similarity task, named SFC here, shown in Fig. 2. Our goal is to utilize the advantage of the feasibility of transfer learning based on semantic similarity, while in the mean time, add the classification task's target into the training process of sentence-pair model for multi-task learning[24, 25]. To obtain such a joint system, we first start with preparation work, which is pre-training a sentence-pair model on external corpus for sentence pair duplication identification, and then find an auxiliary model or tool which can help us sample out top-K most related intent classes. This auxiliary model or tool can either be searching engine such as elasticsearch[26] or a text classification model trained on task-specific chatbot data. Afterwards, we further fine-tune the sentence-pair model with 2 training tasks: 1) the regular sentence pair similarity tasks. Here we use the text classification model as an auxiliary model in this paper to sample sentence pairs for training based on negative sampling strategy[27]. The reason is that a task-specific chatbot often has hundreds of intent classes, which forms too many sentence pairs for training since any two sentences from two different classes can form a sentence-pair of negative sample. Besides, according to Bamler[27], training on negative samples from most confusing wrong label can help model converge faster and obtain better performance. 2) the classification task on top-K (here K is a hyperparameter) candidate classes provided by

our auxiliary model or tool. Here we use the average pooling of the representation given by sentence-pair model for sentence pairs formed by the user input query and the corpus sentences in each class as the feature. We stack the task-specific layers on the top of the shared-parameter sentence-pair model structure, which is a classic parameter sharing mechanism[25, 28, 29] for multi-task[23]. In this way, specific knowledge contained in these 2 tasks can be fully explored and deeply interact with each other to obtain better overall performance.

However, since the auxiliary tool in joint system mentioned above works in a separate stage from sentence-pair model, we find the the quality of sampled candidate classes might limit the overall performance of SFC, since the candidate classes for each user query is always fixed during the multi-task training process. This observation motivates us to further improve SFC into a joint model of sentence-pair model and text classification model in a hierarchical architecture by putting different tasks at different network layers[30, 31] and then training them together, shown in Fig 3. Here, the text classification model involved in joint training process replace the role of auxiliary tools. In this way, the sentence pairs selected from top-K classes becomes dynamic, which means the text classification model will also be further optimized to provide better top-K classes pooling result.

We summarize our contributions as follow:

1) We propose a novel joint system named SFC with multi-task training technique designed for task-specific conversational chatbot with low-resource, in which we make full utilization of the advantages brought by both the text classification task and sentence-pair semantic similarity task.

2) To further improve the system performance of SFC, we also propose an innovative joint model structure, in which the text classification model and sentence-pair model are fused into one single model for joint training.

3) Experiment results on 4 public and 1 private short-text classification datasets with respect to intent recognition tasks under conversational chatbot scenario all demonstrated that our proposed SFC joint system with multi-task, as well as the improved version of SFC, can both achieve remarkable improvement comparing to some powerful single-task and single-model baselines, especially under few-shot settings.

## 2. METHODOLOGY

In this section, we will describe a system of similarity model fused with classification model, named as SFC, in detail. We come up with 2 different approaches for implementation of SFC system, in which hierarchical SFC is a further improvement of 2-stage SFC for better performance.

### 2.1. 2-stage SFC with multi-task learning

The detailed network structure of 2-stage SFC is shown in Fig. 2. It is composed with 2 separate stages in charge of different jobs. In stage 1, we can use any auxiliary model or tool to provide top-K most related class labels for sampling sentence pairs of both positive samples and negative samples for stage 2. Suppose we have a training data set $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$ of $N$ data points, in which $x_i$ is the user input query and $y_i$ is a single class label from a class label set of $C$ classes in total. If we want to generate sentence pair data set $\mathcal{D}' = \{[(x, x')_j, l_j]\}_{j=1}^{M}$, where $(x, x')_j$ is sentence pair grouped by 2 user input queries and $l_j \in \{0, 1\}$ is similarity class label, see Equ. 1,

$$l_j = \begin{cases} 0 & y_x \neq y_{x'} \\ 1 & y_x = y_{x'} \end{cases} \tag{1}$$

directly from original data set $\mathcal{D}$ without any sampling strategy, then $\mathcal{D}'$ will have $M = N^2$ data points of sentence pairs, which might be too time-consuming for the training process of stage 2 since we often have hundreds of class labels under task-specific chatbot scenario. Although we may only have a few data points for a single class label, $M$ is still quite a big number.

Therefore, we adopt the idea of adversarial negative sampling strategy raised by [27]. The key idea of this sampling strategy is to train with negative samples that are hard to be distinguished from positive samples. Now suppose we have a user input query $x_i$ and top-K most related class labels set $\mathcal{C}' = \{c'_1, \ldots, c'_K\}$ provided by auxiliary model or tool in stage 1, where $K$ is a hyperparameter set by us that controls candidate class number, we apply the adversarial negative sampling strategy into our sentence pair sampling process by the following 2 steps:

**1. positive sentence pair sampling**: During the training process, we have to first make sure that the ground truth class label $y_i$ for $x_i$ is within the candidate class label set $\mathcal{C}'$. If $y_i \notin \mathcal{C}'$, we will manually add $y_i$ into $\mathcal{C}'$ by replacing $c'_K$ with $y_i$, since $c'_K$ is the least promising candidate label according to the auxiliary model in stage 1. Afterwards, we will randomly sample $P$ sentences $\{x'_0, x'_1, \ldots, x'_P\}$ with the same class label as $y_i$ from data set $\mathcal{D}$ to form the set of sentence pairs with positive label $\mathcal{P}'_i = \{[(x_i, x'_0), 1], [(x_i, x'_1), 1], \ldots, [(x_i, x'_P), 1]\}$, where $P$ is also a hyperparameter set by us that controls the number of sentences we should sample from each class.

**2. negative sentence pair sampling**: As for negative sentence pairs, we will also randomly sample $P$ sentences for each class in the negative candidate class label set $\mathcal{C}'\backslash\{y_i\}$. The class labels in $\mathcal{C}'\backslash\{y_i\}$ are the set of most confusing class labels comparing to the ground truth $y_i$, so we assume that the sentence pairs with negative label grouped by user input query $x_i$ and sentences with class labels in $\mathcal{C}'\backslash\{y_i\}$ are strong adversarial negative

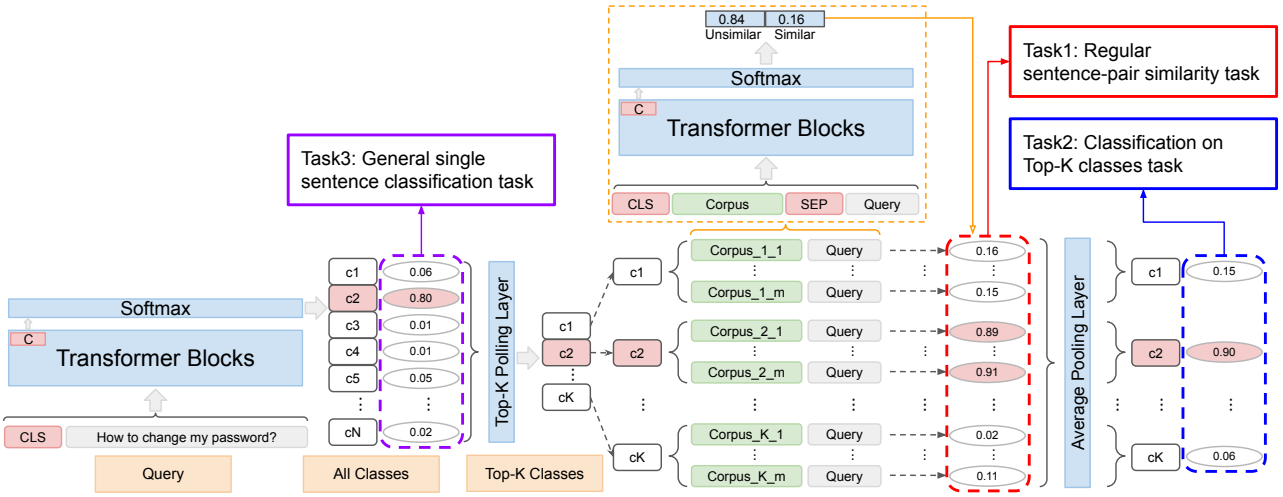



**Fig. 3**. **Network Structure of hierarchical SFC:** a joint model of sentence-pair model and text classification model organized in hierarchical structure by completing different tasks at different network layers.

samples for sentence-pair similarity model that can help enhance the training speed and performance. According to the same method as positive sentence pair sampling, we can obtain the set of sentence pairs with negative label as $\mathcal{N'}_i = \{[(x_i, x'_0), 0], [(x_i, x'_1), 0], ..., [(x_i, x'_{P\cdot(K-1)}), 0]\}$

In this way, we can generate the sentence pair data set $\mathcal{D}'$ for stage 2 based on the top-K class label set $\mathcal{C}'$ provided by stage 1 as $\mathcal{D}' = \{\mathcal{P'}_i \cup \mathcal{N'}_i\}_{i=1}^{N}$, in which we will have $M = N \cdot K \cdot P$ data points of sentence pairs in total.

After explaining the data sampling principle for sentence pairs and how the 2 stages in SFC system connect with each other, let's go deeper into model structure within these 2 stages respectively to see how it works.

### *2.1.1. stage 1: classification model for providing top-K candidate class labels*

As for stage 1, our SFC system can adapt to any auxiliary model or even searching tool based on Term Frequency-Inverse Document Frequency (TF-IDF) retrieval. In this paper, however, we choose to fine-tune RoBERTa on our task-specific data set $\mathcal{D}$ by adding a softmax classifier on top of it, the same as the structure shown in Fig. 1(a), for better overall performance. According to our experimental result, RoBERTa can guarantee that approximately 95% ground truth label be included in top-5 candidate class labels with highest probability after being fine-tuned on task-specific data set with experimental result measured by F-score at around 0.75. So we use RoBERTa as encoder of classification model in our setting at stage 1 to ensure low limitation for the performance of stage 2.

For single-sentence text classification tasks, given a data point $x_i$ with class label $y_i$ from data set $\mathcal{D}$, RoBERTa takes the final hidden state $h_i$ of the first token [CLS] as the representation for the whole sequence of $x_i$. Let's suppose we have a linear layer as $\Phi_i^C = W^C h_i + b^C$, where $W^C$ and $b^C$ are trainable parameters, the probability score for $x_i$ can be calculated as $p_i^C = softmax(W^C h_i + b^C) = softmax(\Phi_i^C)$. Then, the loss for stage 1 classification model is shown in Equ. 2, in which the first term $\Phi_{i,y_i}^C$ encourages high probability for prediction of the correct label $y_i$.

$$\begin{aligned}\mathcal{L}^C &= \frac{1}{N}\sum_{i=1}^{N} -log(\frac{exp(\Phi_{i,y_i}^C)}{\sum_{j=1}^{C} exp(\Phi_{i,j}^C)}) \\ &= \frac{1}{N}\sum_{i=1}^{N} -\Phi_{i,y_i}^C + log(\sum_{j=1}^{C} exp(\Phi_{i,j}^C))\end{aligned} \tag{2}$$

### *2.1.2. stage 2: sentence-pair model for multi-task learning*

As for stage 2, we also choose RoBERTa as the encoder for sentence pairs, as shown in Fig. 2. Before sending sentence pairs to Roberta, we pack sentence pairs into a single sequence and separate them with a special token [SEP]. Roberta will also add a learned embedding to every token indicating whether it belongs to the first sentence or the second one. Then, we utilize the cross-attention mechanism inside RoBERTa, and then extract semantic relationship between 2 sentences from the output softmax layer, which is placed on top of the last hidden representation of the [CLS] token.

Before we start to fine-tune our sentence-pair model on task-specific dataset, we first fine-tune RoBERTa on Quora dataset[32], which contains 404,290 potential duplicate question pairs, for transfer learning. Then we continue to do multi-task training on sentence-pair dataset $\mathcal{D}'$ sampled from the top-K candidate class labels provided by stage 1.

The 2 tasks for stage 2 training process are listed below.

**• Task 1: Regular sentence-pair similarity task**

For sentence pair semantic similarity task, given a data point $(x, x')_i$ with similarity label $l_i$ from data set $\mathcal{D}' = \{[(x, x')_i, l_i]\}_{i=1}^{M}$, Roberta takes the final hidden state $h_i$ of the first token [CLS] as the representation for the sequence of packed sentence pair $(x, x')_i$. Let's suppose we have a linear layer as $\Phi_i^S = W^S h_i + b^S$, where $W^S$ and $b^S$ are trainable parameters, and the probability score for $(x, x')_i$ can be calculated as $p_i^S = softmax(\Phi_i^S)$.

Due to the fact that sentence pairs within the same class is much less than sentence pairs from different classes, the data set $\mathcal{D}'$ is imbalanced. Therefore, we will accommodate a weight variable $w^S = [K-1, 1]$ to the loss to eliminate the bias brought by imbalanced data, shown in Equ. 3.

$$\begin{aligned}\mathcal{L}^S &= \frac{1}{M}\sum_{i=1}^{M} -w_{l_i}^S \cdot log(\frac{exp(\Phi_{i,l_i}^S)}{\sum_{j=0}^{1} exp(\Phi_{i,j}^S)}) \\ &= \frac{1}{M}\sum_{i=1}^{M} -w_{l_i}^S \cdot \Phi_{i,l_i}^S + w_{l_i}^S \cdot log(\sum_{j=0}^{1} exp(\Phi_{i,j}^S))\end{aligned} \tag{3}$$

**• Task 2: Classification task on top-K classes based on sentence-pair model**

As for top-K classification task, it shares the same sentence-pair similarity model structure with task 1. The only difference is that we add a task-specific average pooling layer, shown in Fig. 2 to implement the classification task based on sentence-pair model.

We already know that the size of $\Phi^S$ in task 1 is $M \times 2 = (N \cdot K \cdot P) \times 2$, where $N$ is the total number of data points in original training data $\mathcal{D}$, $K$ is a hyperparameter that controls the number of candidate class labels provided by stage 1, and $P$ is also a hyperparameter that controls the number of sentences we should randomly sample from each candidate class labels. Now, for the average pooling layer, we first reshape $\Phi^S$ into the size of $N \times K \times P \times 2$, and then split it into $\Phi^{K,pos}$ and $\Phi^{K,neg}$, and both of them will have the size of $N \times K \times P$. In this way, $\Phi^{K,pos}$ can represent the level of similarity for each sentence pair, and in the mean time, $\Phi^{K,neg}$ can represent the level of dissimilarity for each sentence pair. Then, we can do average pooling for each candidate class among the top-K classes as shown in Equ. 4 and Equ. 5.

$$\xi_{i,j}^{K,pos} = \sum_{p=1}^{P} \Phi_{i,j,p}^{K,pos} \tag{4}$$

$$\xi_{i,j}^{K,neg} = \sum_{p=1}^{P} \Phi_{i,j,p}^{K,neg} \tag{5}$$

With the average pooling result $\xi^{K,pos}$ and $\xi^{K,neg}$, we also generate a top-K class label $l_i^K \in [1, \ldots, K]$ for each $\xi_i^{K,pos}$ and $\xi_i^{K,neg}$. The loss for top-K classification task is shown in Equ. 6. In Equ. 7 and Equ. 8, the first term $\xi_{i,l_i^K}^{K,pos}$ and $\xi_{i,l_i^K}^{K,neg}$ encourage high level of similarity and low level of dissimilarity for prediction of the correct label $l_i^K$ within the top-K candidate classes.

$$\mathcal{L}^K = \frac{1}{N}\sum_{i=1}^{N}(\mathcal{L}_i^{K,pos} - \mathcal{L}_i^{K,neg}) \tag{6}$$

where,

$$\begin{aligned}\mathcal{L}_i^{K,pos} &= -log(\frac{exp(\xi_{i,l_i^K}^{K,pos})}{\sum_{j=1}^{K} exp(\xi_{i,j}^{K,pos})}) \\ &= -\xi_{i,l_i^K}^{K,pos} + log(\sum_{j=1}^{K} exp(\xi_{i,j}^{K,pos}))\end{aligned} \tag{7}$$

$$\begin{aligned}\mathcal{L}_i^{K,neg} &= -log(\frac{exp(\xi_{i,l_i^K}^{K,neg})}{\sum_{j=1}^{K} exp(\xi_{i,j}^{K,neg})}) \\ &= -\xi_{i,l_i^K}^{K,neg} + log(\sum_{j=1}^{K} exp(\xi_{i,j}^{K,neg}))\end{aligned} \tag{8}$$

Therefore, the overall loss function for multi-task learning in stage 2 is shown in Equ. 9. The training objective of stage 2 is to minimize the weighted sum of task-specific losses. Here $\alpha_S$ and $\alpha_K$ are weights of task 1 and task 2 respectively, which are $\alpha_S = 0.125$ and $\alpha_K = 0.875$ in our setting.

$$\mathcal{L} = \alpha_S \mathcal{L}^S + \alpha_K \mathcal{L}^K \tag{9}$$

### 2.2. hierarchical SFC with joint model structure

Despite the outstanding experimental result of 2-stage SFC on various tasks, we still find a limitation brought by the 2-stage system structure. In 2-stage SFC, Stage 1 and stage 2 are separate, so they do not interact with each other during training. In this case, the performance of stage 1 can limit the potential of stage 2, meanwhile, stage 2 also cannot give training feedback back to stage 1 for fine-tuning.

Therefore, to further improve the overall performance of SFC, we proposed a hierarchical joint model structure, shown in Fig. 3. In hierarchical SFC, classification model is being placed in lower layer level to dynamically provide top-K candidate class labels for the sentence-pair similarity model placed in higher layer level through a top-K pooling layer. In this way, classification model and similarity model can be merged into one single joint-model for multi-task training

| Dataset | Domain | #Class | #Samples | #Samples/Class | Experimental Settings |
|---|---|---|---|---|---|
| ITG | FAQ chatbot | 228 | 3938 | 17 | 3-fold |
| Amazon-670k | Product review | 250 | 1942 | 8 | class sampling |
| HWU64 | Intent detection | 64 | [320, 640, 960, 1280] | [5, 10, 15, 20] | data sampling |
| CLINC150 | Intent detection | 150 | [750, 1500, 2250] | [5, 10, 15] | data sampling |
| BANKING77 | Intent detection | 77 | [385, 770, 1155, 1540] | [5, 10, 15, 20] | data sampling |

**Table 1**. Key statistics for experimental datasets and few shot settings

with sentence pairs sampled from varying candidate class labels, thus avoiding the limitation brought by 2 separate stage structure.

There are 3 tasks in total during the training process of hierarchical SFC, shown in Fig. 3, and the overall loss is also the weighted sum of all the task-specific losses, shown in Equ. 10. Here, $\alpha_S$, $\alpha_K$ and $\alpha_C$ are the weights for task 1: sentence-pair similarity, task 2: top-K classification and task 3: general single sentence classification respectively, which are $\alpha_S = 0.25$, $\alpha_K = 0.25$ and $\alpha_C = 0.5$ in our setting.

$$\mathcal{L} = \alpha_S \mathcal{L}^S + \alpha_K \mathcal{L}^K + \alpha_C \mathcal{L}^C \tag{10}$$

Finally, when doing inference, for each user input query $x$, we will choose the class label that contains the sentence $x'$ with highest similarity score $\Phi$, after being packed into a sentence pair $(x, x')$ with query $x$, as our final result for prediction.

## 3. EXPERIMENTS

In this section, we conduct extensive experiments over 5 task-specific datasets to evaluate the performance of our proposed SFC system and the effectiveness of the methodology of multi-task learning and hierachical joint-model structure under low-data settings.

### 3.1. Datasets

The datasets used in our experiments aims to help us study low-resource text classification problem under task-specific conversational chatbot scenario. Therefore, we keep all the dataset under low-resource chatbot corpus setting, which is a large number of class labels (50-250) with a few examples(5-20) in short text for each class, by sampling smaller subsets from the full scale when the raw datasets are oversized, see Table 1.

According to the task and domain, we categorize our experimental datasets into the following 3 categories:

**• Full-scale real-world task-specific faq chatbot dataset**

**ITG** is a real-world FAQ chatbot dataset composed of question and answer pairs about online English teaching. It contains 3,938 sample questions for 228 class labels, where each class label corresponds to an unique answer for a bunch of similar sample questions.

We divide the whole dataset into 3 folds randomly with approximately the same amount of data within each fold, and then choose 2 folds as the training set, the remaing 1 fold as the test set for our experiments. We can obtain 3 different combinations among the 3 folds, and we will use the average experimental results of these 3 combinations as the final evaluation result for this dataset.

**• Subset of customer review dataset for extreme classification**

**Amazon-670K** is a customer product review dataset for text classification task from the extreme classification repository with full data containing 670,091 class labels, 285,176 training samples and 150875 test samples[33]. As the data set contains multiple class labels per training sample, we keep only the first class label for each data point. Besides, we also randomly sampled 250 class labels from all the classes that contain only 5-15 data points. In this way, we simulate the low-resource chatbot corpus setting with the subset of 1,942 training samples and 716 test samples for 250 class labels.

**• Subset of recently-released intent detection datasets**.

From our point of view, task-specific chatbot usually starts with small amount of data at the initial stage. As more and more customers start to communicate with the chatbot, more data can be extracted from conversation log. Base on these assumptions, we do experiments on the following 3 intent detection dataset under sampling method similar to the approach in [34] but in more sophisticated few-shot settings, while keeping the same test sets for each experiment to see the change of the improvement scale with decreasing data amount.

**HWU64** is a intent detection dataset designed for home robot scenario[35]. It aims at the specific task of capturing the intent for different user requests to home robot and finding the corresponding answer. The raw dataset contains 25,716 data points for 64 class labels through crowdsourcing, which raises much more data than common real-world circumstances. Therefore, we experiment with the setups where only 5, 10, 15 and 20 training data points are available for each class label by random sampling.

**CLINC150** is a dataset designed for task-oriented systems with 23,700 queries that are short and unstructured for 150 intents, in the same style made by real users through crowdsourcing[36]. Here, we sample 5, 10 and 15 training

| Models | CLINC150 | | | BANKING77 | | | | HWU64 | | | | ITG | Amazon-670k |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 5 | 10 | 15 | 5 | 10 | 15 | 20 | 5 | 10 | 15 | 20 | Full | Full |
| TextCNN (classification) | 0.5318 | 0.6963 | 0.7609 | 0.4408 | 0.6436 | 0.7366 | 0.7918 | 0.3112 | 0.4007 | 0.4823 | 0.5272 | 0.6624 | 0.4401 |
| LEAM (classification) | - | - | - | - | - | - | - | 0.4545 | 0.5554 | 0.5936 | 0.6599 | 0.7086 | 0.6091 |
| BERT-large (classification) | 0.8080 | 0.8904 | 0.9265 | 0.578 | 0.8004 | 0.8518 | 0.8827 | 0.4711 | 0.5963 | 0.6342 | 0.7010 | 0.7485 | 0.6658 |
| ALBERT-xxlarge (classification) | 0.6500 | 0.9008 | 0.9296 | 0.3608 | 0.6345 | 0.8090 | 0.8530 | 0.3549 | 0.3876 | 0.5808 | 0.6320 | 0.7253 | 0.6893 |
| RoBERTa-base (classification) | 0.8732 | 0.9254 | 0.9363 | 0.7305 | 0.8654 | 0.8808 | 0.9080 | 0.5831 | 0.6790 | 0.7064 | 0.7100 | 0.7734 | 0.6708 |
| RoBERTa-large (classification) | 0.8974 | 0.9372 | 0.9508 | 0.7690 | 0.8728 | 0.8966 | 0.9099 | 0.6044 | 0.7002 | 0.7129 | 0.7436 | 0.7990 | 0.7156 |
| RoBERTa-large (similarity) | 0.8266 | 0.8861 | 0.9023 | 0.757 | 0.8476 | 0.8614 | 0.8743 | 0.5425 | 0.6164 | 0.6503 | 0.6729 | 0.7418 | 0.6362 |
| 2-stage SFC (task1 only) | 0.8979 | 0.9457 | 0.9517 | 0.7975 | 0.8818 | 0.8962 | 0.9109 | 0.6477 | 0.7055 | 0.7200 | 0.7232 | 0.7972 | 0.7189 |
| 2-stage SFC (task2 only) | 0.9162 | 0.9424 | 0.9530 | 0.7997 | 0.8823 | 0.8945 | 0.9123 | 0.6498 | 0.6980 | 0.7202 | 0.7358 | 0.8020 | 0.7311 |
| 2-stage SFC (task1 + task2) | 0.9167 | 0.9456 | 0.9571 | 0.7954 | 0.8854 | 0.8931 | 0.9192 | 0.6525 | 0.7001 | 0.7168 | 0.7476 | **0.8124** | 0.7364 |
| Hierarchical SFC | **0.9231** | **0.9560** | **0.9644** | **0.8270** | **0.9069** | **0.9103** | **0.9209** | **0.6697** | **0.7211** | **0.7254** | **0.7497** | 0.8114 | **0.7445** |

**Table 2**. F-scores on five task-specific dataset for text classification in chatbot under low resource. For ITG, we keep the full dataset. For Amazon-670k, we randomly sampled 250 classes with training sample numbers within 5-15 samples per class. For CLINC150, BANKING77, HWU64, we set up various few-shot settings (5/10/15/20 samples per class) while keeping the test set to be fixed. The highest scores for each data setting are marked in bold.

data points for each intent class label by random sampling for our experimental setup.

**BANKING77** is a intent detection dataset for bank customer services. The raw dataset contains 13,084 data points for 77 class labels[34]. Here, we sample 5, 10, 15, 20 training data points for each class label by random sampling for our experimental setup.

### 3.2. Baselines

We choose 3 groups of representative baselines:

**Group 1: single-task classification model.** This group of baseline models are based on the network structure for classification model shown in 1(a). As for non-pretrained neural network, we choose TextCNN[9] as a representative baseline. As for the pretrained model, we choose RoBERTa, BERT and ALBERT, all of which are among the most powerful pretrained models that achieve outstanding results in many NLP tasks, in our experimental setting. Since we apply RoBERTa in our SFC system, we naturally adopt single-task classification model based on RoBERTa-large and RoBERTa-base as our baseline models. Besides, we also adopt the recently released baseline pre-trained model ALBERT-xxlarge[37] for comparison. Since we use different few-shot settings from [34] for those intent detection datasets, we also use BERT-large tuned on our few-shot settings as an essential reference for experimental results.

**Group 2: single-task similarity model.** This group of baseline models are based on network structure for similarity model shown in 1(b). We choose RoBERTa-large for the pretrained model part as an powerful baseline. But during the evaluation step, we use elastic search to do a rough filter for candidate sentence pairs to guarantee a reasonable inference time for each user query.

**Group 3: SFC system with ablation study.** This group of baseline is based on the network structure of 2-stage SFC shown in 2. We conduct ablation study by introducing single task SFC system with only Task 1 or Task 2 to investigate the influence of each component within the total loss.

### 3.3. Implementation Details

For our SFC system, we use RoBERTa-large as the pretrained model for both classification task and sentence-pair similarity task. We further fine-tune our SFC system on task-specific datasets with max sequence length of 128, learning rate of 1.5e-5 for RoBERTa-large pretrained model and learning rate of 5e-4 for task-specific layer with polynomial decay and Adam optimizer. Also, we set the max number of the epoch to 100 and save the best model on validation set based on test result in F-score. We also report the F-score as the main evaluation measure for all experimental results.

For the experiments in Table 2, we empirically set the hyperparameter to be $K = 5$ and $P = 10$ for all SFC variants. In Table 3, we conduct extensive experiments to see the performance of SFC system under various hyperparameter set-

tings.

### 3.4. Result Analysis

Table 2 presents the results for each experimental data and model set-up in detail. The performance of each model is measured in F-score on the fixed test set for each data set. The overall performance of SFC system is better than all the competitive single-model baseline models on five experimental datasets. This observation gives us a general idea that classification model and similarity model can actually provide each other with complementary information, and combining them together into a joint system can effectively enhance the overall capability based on richer extracted semantic information. However the relative performance among all the variants of SFC also seems to depend on some specific charateristic of each data set. Therefore, we did the following more detailed analysis from several different aspect.

**Comparison among SFC variants.** First, we turn to compare among the four SFC variants shown in Table 2 to analyze the improvement brought by multi-task and hierarchical joint-model structure. We conduct ablation experiments to study the benefits brought by multi-task training by comparing multi-task 2-stage SFC with single-task 2-stage SFC. We can see from Table 2 that 2-stage SFC with only task 1 (sentence-pair similarity task) degrades by 0.65 percent on average and 2-stage SFC with only task 2 (top-K classification task) degrades by 0.3 percent on average. The degradation indicates that training with these 2 tasks together in SFC can effectively help the sentence-pair model extract more knowledge from the limited amount of data. Besides, task 2 degrades 0.35 percent less than task 1. This observation suggests that task 2 may play a relatively more important role in multi-task learning, which leads us to put larger weight $\alpha_K$ to the loss $\mathcal{L}^K$ of task 2 when calculating the total loss $\mathcal{L}$. Also, we can see that the hierachical SFC steadily outperforms 2-stage SFC on 5 diverse datasets by 2 percents on average. This observation supports the idea that the hierachical joint model structure can alleviate the limitation brought by the lack of interaction between similarity model and classification model in 2-stage SFC. Therefore, for the following analysis, we will focus on the comparison between hierachical SFC and the other baseline models.

**Analysis under few-shot scenarios.** Next, we analyze the overall improvement trend of SFC under various few-shot settings. Our main focus is to study chatbot building in common situation during real-world application where only a few sample sentences are available for each class. The experimental results for the 3 diverse intent detection datasets under few-shot settings indicate that our proposed SFC can achieve +2.47 percents average improvement over one of the most powerful baseline models, which is RoBERTa-large classification model, on all these few-shot settings. Moreover, the improvement in F-score becomes more and more prominent with the decrease of data sample number per class. Our hierarchical SFC can achieve +4.97 percents average improvements over RoBERTa-large classification model in the most extreme few-shot setting with only 5 training samples per class. This observation indicates that SFC is especially suitable for being applied during the initial stage of building task-specific chatbot system, where the amount of data sample for each class label is extremely scarce.

**Fusion of classification and similarity model.** Then, we begin to compare the general performance of SFC with single-task classification and similarity baselines. As for the classification model, hierarchical SFC outperforms TextCNN by +37.99 percents, ALBERT-xxlarge by +17.87 percents[1], BERT-large by +9.81 percents, RoBERTa-base by +4.52 percents, RoBERTa-large by +2.47 percents on average F-score over 5 datasets. In this way, we can see that RoBERTa is the best baseline pre-trained model for the low-resource text classification task we are studying. As for the similarity model, we only choose the strongest pretrained model RoBERTa-large as our baseline, and hierarchical SFC achieves the improvement of +7.81 percents on average. Moreover, hierarchical SFC's result on 2 datasets without data sampling for few-shot settings, which are ITG and Amazon-670k, are also 1.24 and 2.89 percents higher than Roberta-large classification model, and are 6.96 and 10.77 percents higher than Roberta-large similarity model. These improvements illustrate that putting classification model and similarity model at different network layers to cooperate with each other is a successful design of hierarchical network structure for joint model, which enables these two models assist each other for better overall results.

**Settings of hyperparameters.** Finally, we analyze the effect brought by different settings of hyperparameters, see Table 3. We conduct experiments on 5 different settings of hyperparameters, which are the number of candidate class number $K$ and the number of sentence pairs $P$ for each class, on ITG and Amazon-670K datasets to find the most suitable setting of these 2 hyperparameters. We control the total number of sentence pairs sampled for each user query into about the same level of amount(50-60). Through analyzing the experimenal results shown in Table 3, we find that the best setting of hyperparameters might be slightly different for datasets of different data size. The class label number of ITG and Amazon-670k are similar in our experimental setting, which is around 250. However the total amount of data points of ITG is approximately 2 times larger than Amazon-670k. Therefore, the setting of $K = 5$ and $P = 10$ is good enough for ITG since most of the classes contains over 10 data points. However, for Amazon-670k, the amount of data sample for each class is scarce, thus, we may need more candidate classes

[1] We found that the performance of ALBERT.xxlarge is extremely unstable for classification task. Here, we did multiple run with different settings of parameters such as learning rate and batch size for each dataset, and record the best result we can obtain in Table 2.

| Dataset | Model | $K=3$ $P=20$ | $K=5$ $P=10$ | $K=10$ $P=5$ | $K=15$ $P=4$ | $K=20$ $P=3$ |
|---|---|---|---|---|---|---|
| ITG | 2-stage SFC | 0.8034 | 0.8124 | 0.8008 | 0.7967 | 0.7934 |
| | Hierarchical SFC | 0.7882 | 0.8114 | 0.8010 | 0.7972 | 0.7826 |
| Amazon-670k | 2-stage SFC | 0.7278 | 0.7364 | 0.7366 | 0.7328 | 0.7204 |
| | Hierarchical SFC | 0.7334 | 0.7445 | 0.7516 | - | - |

**Table 3**. We show the performance of SFC from different settings of hyperparameters for $K$, the candidate class number, and $P$, the sample number for each class.

to sample enough effective sentence pairs for sentence-pair model layer of SFC. In this way, the setting of $K = 10$ and $P = 5$ becomes a good choice.

In conclusion, to the best of our knowledge, these results place SFC as the best performing model on 5 diverse experimental datasets for text classification task under low-resource task-specific conversational chatbot scenario.

## 4. CONCLUSION

In this work, we successfully proposed SFC, a joint system of classification model and sentence-pair similarity model with multi-task learning, which can effectively alleviate the limitation of classification model and similarity model for low-resource text classification tasks. Experimental results illustrate that our proposed system can achieve great improvement on 5 diverse datasets for task-specific chatbot system over several competitive single-task baseline models. Furthermore, we especially analyze the excellent performance of SFC when facing the challenge brought by few-shot scenarios, which is quite common in real-world application of chatbot system.

## 5. REFERENCES


[1] Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young, "A network-based end-to-end trainable task-oriented dialogue system," *arXiv preprint arXiv:1604.04562*, 2016.

[2] Ge Song, Yunming Ye, Xiaolin Du, Xiaohui Huang, and Shifu Bie, "Short text classification: A survey," *Journal of multimedia*, vol. 9, no. 5, pp. 635, 2014.

[3] Jindong Chen, Yizhou Hu, Jingping Liu, Yanghua Xiao, and Haiyun Jiang, "Deep short text classification with knowledge powered attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 6252–6259.

[4] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi, "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *Proceedings of the 17th international conference on World Wide Web*, 2008, pp. 91–100.

[5] Rui Yan, Xian-bin Cao, and Kai Li, "Dynamic assembly classification algorithm for short text," *Acta Electronica Sinica*, vol. 37, no. 5, pp. 1019–1024, 2009.

[6] Wen Hua, Zhongyuan Wang, Haixun Wang, Kai Zheng, and Xiaofang Zhou, "Short text understanding through lexical-semantic analysis," in *2015 IEEE 31st International Conference on Data Engineering*. IEEE, 2015, pp. 495–506.

[7] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas, "Short text classification in twitter to improve information filtering," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 2010, pp. 841–842.

[8] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou, "Diverse few-shot text classification with multiple metrics," *arXiv preprint arXiv:1805.07513*, 2018.

[9] Yoon Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[10] Xiang Zhang, Junbo Zhao, and Yann LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[11] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun, "Very deep convolutional networks for text classification," *arXiv preprint arXiv:1606.01781*, 2016.

[12] Amr Mousa and Björn Schuller, "Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, pp. 1023–1032.

[13] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang, "Recurrent neural network for text classification with multi-task learning," *arXiv preprint arXiv:1605.05101*, 2016.

[14] Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin, "Joint embedding of words and labels for text classification," *arXiv preprint arXiv:1805.04174*, 2018.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[17] Harish Tayyar Madabushi, Elena Kochkina, and Michael Castelle, "Cost-sensitive bert for generalisable sentence classification with imbalanced data," *arXiv preprint arXiv:2003.11563*, 2020.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[19] Sina Jafarpour, Christopher JC Burges, and Alan Ritter, "Filter, rank, and transfer the knowledge: Learning to chat," *Advances in Ranking*, vol. 10, pp. 2329–9290, 2010.

[20] Anton Leuski and David Traum, "Npceditor: Creating virtual human dialogue using information retrieval techniques," *Ai Magazine*, vol. 32, no. 2, pp. 42–56, 2011.

[21] Maali Mnasri, "Recent advances in conversational nlp: Towards the standardization of chatbot building," *arXiv preprint arXiv:1903.09025*, 2019.

[22] Oren Barkan, Noam Razin, Itzik Malkiel, Ori Katz, Avi Caciularu, and Noam Koenigstein, "Scalable attentive sentence pair modeling via distilled sentence embedding.," in *AAAI*, 2020, pp. 3235–3242.

[23] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang, "How to fine-tune bert for text classification?," in *China National Conference on Chinese Computational Linguistics*. Springer, 2019, pp. 194–206.

[24] Rich Caruana, "Multitask learning: A knowledge-based source of inductive bias icml," *Google Scholar Google Scholar Digital Library Digital Library*, 1993.

[25] Ronan Collobert and Jason Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167.

[26] Manda Sai Divya and Shiv Kumar Goyal, "Elasticsearch: An advanced and quick search technique to handle voluminous data," *Compusoft*, vol. 2, no. 6, pp. 171, 2013.

[27] Robert Bamler and Stephan Mandt, "Extreme classification via adversarial softmax approximation," *arXiv preprint arXiv:2002.06298*, 2020.

[28] Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," *arXiv preprint arXiv:1804.00079*, 2018.

[29] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao, "Multi-task deep neural networks for natural language understanding," *arXiv preprint arXiv:1901.11504*, 2019.

[30] Anders Søgaard and Yoav Goldberg, "Deep multi-task learning with low level tasks supervised at lower layers," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 231–235.

[31] Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher, "A joint many-task model: Growing a neural network for multiple nlp tasks," *arXiv preprint arXiv:1611.01587*, 2016.

[32] Shankar Iyer, Nikhil Dandekar, and Kornél Csernai, "First quora dataset release: Question pairs," *data. quora. com*, 2017.

[33] Kush Bhatia, Kunal Dahiya, Himanshu Jain, Yashoteja Prabhu, and Manik Varma, "The extreme classification repository: multi-label datasets & code," *URL http://manikvarma. org/downloads/XC/XMLRepository. html*, 2016.

[34] Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić, "Efficient intent detection with dual sentence encoders," *arXiv preprint arXiv:2003.04807*, 2020.

[35] Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser, "Benchmarking natural language understanding services for building conversational agents," *arXiv preprint arXiv:1903.05566*, 2019.

[36] 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., "An evaluation dataset for intent classification and out-of-scope prediction," *arXiv preprint arXiv:1909.02027*, 2019.

[37] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.