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

Name of author

Address - Line 1
Address - Line 2
Address - Line 3

ABSTRACT

Task-specific chatbot systems have gained many important applications, such as Siri, Google home, and customer service systems. In these applications, one of the core problems is the intention detection from a user's input. Then, the intention with its associated information is mapped into a predefined dialog logic graph, and the chatbot returns corresponding answer. This is the main difference from a free chatbot system. The intention detection, assuming only one intention in a user's input here, can be modeled as a short text classification. In practice, the number of intentions can be hundreds of thousands, as opposed to a few number in the traditional text classification problem; enough training data is not trivial to be filtered out and labeled from tons of raw user logs with miscellaneous information, and generally only 10 to 20 samples per intention are available in the early stage. Then, how to train an efficient model in such a practical and important application is critical. In this work, we propose a novel model, called similarity model fused with classification Model (SFC), which combines a classification model and a similarity model, and also borrows the power of multi-task training as well. We conduct extensive experiments on 4 public and 1 private datasets. The experimental results show that our new model outperforms very strong baselines, BERT, RoBERTa and ALBERT based pretrained models by 2 percentages in average accuracy.

1. INTRODUCTION

Task-specific conversational chatbot (Wen et al. 2016) has been applied into many practical products. For example, it can share the working burden of human customer service agents responsible for customers' questions about certain products or services. Another example is the speech-based control in devices like intelligent speakers, such as Siri, Google home, Baidu Xiaodu, and smart home equipments. Regardless of whether these conversations belong to single-turn or multi-turn, an essential technique underlying the chatbot is to identify the real intention behind a user's question or reply. Then the chatbot is caple of conducting expected actions, text reply or accomplish some operations.

It is a challenging task due to some inherent properties. First, a customer's utterance within a conversation is usually quite short. Since it does not contain enough information, short text (Song et al. 2014) are thought to be more ambiguous compared to long texts such as paragraphs or documents, poseing a great challenge (Chen et al. 2019) for classification task (Phan, Nguyen, and Horiguchi 2008; Yan, Cao, and Li 2009; Hua et al. 2015). Second, in the initial stage of building a chatbot , it is often rather hard to collect sufficient labeled data to train a good model. Because developers have to examine tons of real system logs for typical user's utterances, and then label them with expected intentions. Therefore, the key to building a task-specific chatbot with high performance becomes solving the challenge of short-text classification (Sriram et al. 2010) problem under few-shot setting (Yu et al. 2018).

Generally there are two kinds of approaches to address this problem, classification model that directly maps an input into a most likely label, and similarity model that searches a label whose associated data is most similar to an input.

The first one, *text classification*, includes classic long text classification and short text classification in this scenario. Besides traditional machine learning models like SVM, boosting tree, many work (Wen et al. 2016) choose neural network such as convolutional neural networks (CNNs) (Kim 2014; Zhang, Zhao, and LeCun 2015; Conneau et al. 2016) and long short term memory networks (LSTMs) (Mousa and Schuller 2017; Liu, Qiu, and Huang 2016) to accomplish the task of extracting semantic feature from limited amount of words. Their common model structure is adding a softmax classifier as the final layer and mapping to labels. Another two interesting lines of work, label-word joint models, and joint NER and classification are discussed in the related work section. Afterwards, pre-trained language models on large corpus like BERT (Devlin et al. 2018) and RoBERTa (Liu et al. 2019b) has been proven more powerful in solving many NLP tasks including short-text classification (Madabushi, Kochkina, and Castelle 2020). Especially for few-shot scenarios (Yu et al. 2018), pre-trained model based on transformers (Vaswani et al. 2017), shown in Fig. **??**, tends to do more help to reducing the negative effects brought by scarcity of training data.

The second one, *similarity model* is also called sentence-pair model. Its motivation originates from the idea of Information Retrival (IR) based chatbot (Jafarpour, Burges, and Ritter 2010; Leuski and Traum 2011). 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 (Mnasri 2019). 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 (Devlin et al. 2018) sentence-pair model is based on multiple cross-attention mechanism (Barkan et al. 2020), shown in Fig. **??**. Due to the fact that many experiment results have shown that RoBERTa (Liu et al. 2019b) 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 two kinds of approaches, they still have some limitations in task-specific chatbot scenario. As for the *text classification model*, it is quite hard to use data augmentation method to solve the data insufficiency challenge since the it is usually unfeasible to get large amount of domain-specific data when facing a new task. With respect to the *similarity model*, though we can obtain a large amount of data from semantically duplicate sentence pair identification domain for transfer learning (Sun et al. 2019), it is still quite hard to perform well on a new task-specific dataset, since their objective losses are totally different.

The above limitation motivates us to propose a joint system that is capble of taking advantagges from both classification model and similarity model. We calle it SFC, short for similairty model fused with classification Model, shown in Fig. **??**. To better train SFC, we further bring in multi-task learning (Caruana 1993; Collobert and Weston 2008). Our basic idea cames from two stages. In the first stage, we use an auxiliary model to select top-K most possible labels for a input. This model can be an elastic search (Divya and Goyal 2013) or a text classification model trained on current task-specific chatbot data. In the second stage, we build a classification model whose main modules are similarity models. Then, this structure derives two goals to train towards, the classification loss for our application, and the similarity loss for the main modules. When the two stages are independently optimized, we call *two-stage SFC*. We further find, the the quality of output from the first stage might limit the final performance of system, since the candidate labels for each user query is always fixed in the whole multi-task training process. This observation motivates us to further improve the two-stage SFC into a joint training, called *joint SFC*. In this way, the sentence pairs associated with the top-K classes also become dynamic, and the text classification model in the first stage will also be further optimized to provide better top-K result by the final training loss.

We summarize our contributions as follow:

1. We propose a novel joint system to makes full utilization of the advantages from both text classification model and similarity model.
2. To better train such a system, we bring in multi-task training to obtain reasonable performance.
3. Experiment results from 4 public and 1 private short-text classification datasets, show that our proposed SFC joint system can achieve significant and consistent improvements over strong baselines, especially in the low resource settings.

2. RELATED WORK

Another common approach is text classification neural network based on label-word joint embedding, such as LEAM (Wang et al. 2018), MTLE (Zhang et al. 2017) and EXAM (Du et al. 2019). This approach involves label embedding when constructing the text-sequence representation. Take LEAM as an example, the text classification model is implemented by jointly embedding the word and label in the same latent space (Wang et al. 2018). Afterwards, the text representation are constructed based on text-label compatibility through attention mechanism. In this way, additional sources of semantic information from class labels can be fully leveraged, which is quite helpful especially under few-shot scenario.

As for *label-word joint embedding approach*, it poses noticeable requirements on label information. However, in our cases, since we have large amount of class labels without being well-defined for question answering chatbot, the only thing we can do is to choose one of the user utterance as the class label to represent the key information of the whole class. In this way, the label will become not only prolix but also biased, which will bring some negative effect to label embedding.

Add joint NER and classification problem here.

3. METHODOLOGY

In this section, we will describe our proposed system of similarity model fused with classification model, named as SFC, in detail. We propose two approaches to implement the SFC idea, two-stageg SFC, and its more compact version called joint-SFC.

3.1. Two-stage SFC

The network structure of two-stage SFC is composed with 2 separate stages in charge of different jobs. In stage 1, we can use any auxiliary text classification model or tool to provide top-K most related class labels. These selected class labels will later be used to sample sentence pairs of both positive samples and negative samples for stage 2. Then, in stage 2, we will use sentence pair similarity model to identify the class label with strongest semantic similarity to the user input sentence. Suppose we have a training dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^{N}$ 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 dataset $\mathcal{D}' = \{[(x, x')_j, l_j]\}_{j=1}^{M}$, 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 dataset $\mathcal{D}$ 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.

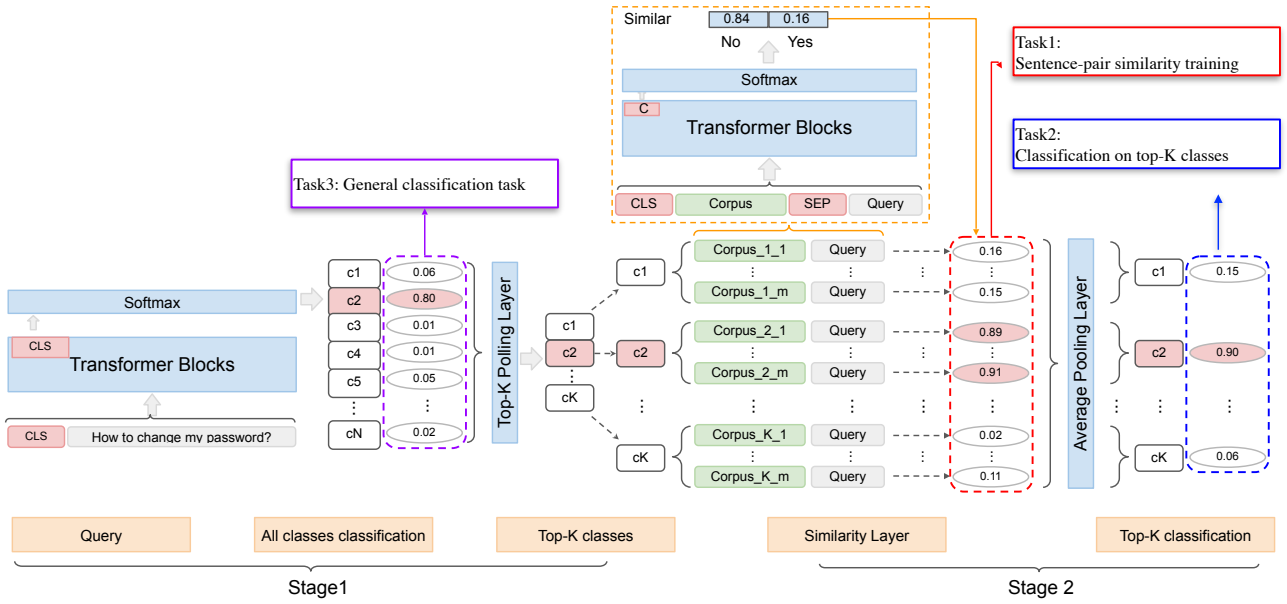

Fig. 1. **Network Structure of SFC:** two-stage SFC and joint-SFC are sharing the same network from stage 1 and stage 2, with the only difference whether two stages being jointly trained.

Therefore, we adopt the idea of adversarial negative sampling strategy raised by (Bamler and Mandt 2020). 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:

The first step is *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 dataset $\mathcal{D}$ 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.

The second step is *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 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 dataset $\mathcal{D}'$ 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 two stages respectively to see how it works.

Stage 1: classification model for providing top-K candidate class labels As for stage 1, our two-stage 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 pretrained model such as RoBERTa on our task-specific dataset $\mathcal{D}$ by adding a softmax classifier on top of it for better overall performance. According to our experimental result, RoBERTa can guarantee that approximately 95% ground truth labels be included in top-5 candidate class labels with highest probability if it has been fine-tuned on task-specific dataset and can obtain a well performance measured by F-score over 0.75 on this dataset. So we use RoBERTa as the 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 dataset $\mathcal{D}$, 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 Φ^C_{i,y_i} encourages high probability for prediction of the correct label y_i.

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

Stage 2: sentence-pair model for multi-task learning As for stage 2, we also choose RoBERTa as the encoder for the sentence pair similarity model. 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 (Iyer, Dandekar, and Csernai 2017), 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.

3.2. Multitask based training

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 $\Phi^S_i = W^S h_i + b^S$, where W^S and b^S are trainable parameters, and the probability score for $(x, x')_i$ can be calculated as $p^S_i = softmax(\Phi^S_i)$.

Due to the fact that sentence pairs within the same class is much less than sentence pairs from different classes, the dataset $\mathcal{D}'$ 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.

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

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. 1 to implement the classification task based on sentence-pair model.

We already know that the size of Φ^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 Φ^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.

$$\xi^{K,pos}_{i,j} = \sum_{p=1}^{P} \Phi^{K,pos}_{i,j,p} \tag{4}$$

$$\xi^{K,neg}_{i,j} = \sum_{p=1}^{P} \Phi^{K,neg}_{i,j,p} \tag{5}$$

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

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

where,

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

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

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 α_S and α_K are weights of task 1 and task 2 respectively. which are $\alpha_S = 0.125$ and $\alpha_K = 0.875$ in our setting.

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

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	intention detection	64	[320, 640, 960, 1280, 1920, 3200]	[5, 10, 15, 20, 30, 50]	data sampling
CLINC150	intention detection	150	[750, 1500, 2250, 3000, 4500, 7500]	[5, 10, 15, 20, 30, 50]	data sampling
BANKING77	intention detection	77	[385, 770, 1155, 1540, 2310, 3850]	[5, 10, 15, 20, 30, 50]	data sampling

Table 1. Statistics for all datasets and few shot settings[1].

3.3. Joint SFC with joint model structure

In two-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 joint model structure, shown in Fig. 1. In joint 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 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 joint SFC, shown in Fig. 1, and the overall loss is also the weighted sum of all the task-specific losses, shown in Equ. 10. Here, α_S, α_K and α_C are the weights for task 1: sentence-pair similarity, task 2: top-K classification and task 3: general single sentence classification respectively.

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

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 Φ, after being packed into a sentence pair (x, x') with query x, as our final result for prediction.

4. 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 joint-model structure under low-data settings.

4.1. 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 did experiments on datasets 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. Besides, we also did extensive experiments with larger scale datasets with 30-50 examples per class to test the stability of our proposed SFC system under normal-size data settings, also see Table 1.

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

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.

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 (Bhatia et al. 2016). As the dataset 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.

Recently-released intention 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 intention detection dataset under sampling method similar to the approach in (Casanueva et al. 2020) 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 intention detection dataset designed for home robot scenario (Liu et al. 2019a). It aims at the specific task of capturing the intention 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, 20, 30 and 50 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

Models	CLINC150						BANKING77						HWU64						ITG	Amazon-670k
	5	10	15	20	30	50	5	10	15	20	30	50	5	10	15	20	30	50	Full	Full
TextCNN (classification)	0.5318	0.6963	0.7609	0.8142	0.8526	0.8867	0.4408	0.6436	0.7366	0.7918	0.8228	0.8619	0.3112	0.4007	0.4823	0.5272	0.5782	0.6262	0.6624	0.4401
LEAM (classification)	0.7514	0.8203	0.8612	0.8802	0.9010	0.9180	0.5422	0.7812	0.8129	0.8280	0.8610	0.8727	0.4545	0.5554	0.5936	0.6599	0.6855	0.7046	0.7086	0.6091
BERT-large (classification)	0.8080	0.8904	0.9265	0.9334	0.9497	0.9595	0.5780	0.8004	0.8518	0.8827	0.8858	0.8982	0.4711	0.5963	0.6342	0.7010	0.7117	0.7424	0.7485	0.6658
ALBERT-xxlarge (classification)	0.8497	0.9008	0.9296	0.9297	0.9466	0.9578	0.5549	0.7981	0.8231	0.8530	0.8571	0.9096	0.4879	0.6116	0.6135	0.6996	0.7094	0.7376	0.7253	0.6893
RoBERTa-base (classification)	0.8732	0.9254	0.9363	0.9482	0.9558	0.9637	0.7305	0.8654	0.8808	0.9080	0.9061	0.9293	0.5831	0.6790	0.7064	0.7100	0.7320	0.7472	0.7734	0.6708
RoBERTa-large (classification)	0.8974	0.9372	0.9508	0.9584	0.9621	0.9733	0.7690	0.8728	0.8966	0.9099	0.9227	0.9313	0.6044	0.7002	0.7129	0.7436	0.7493	0.7678	0.7990	0.7156
RoBERTa-large (similarity)	0.8266	0.8861	0.9023	0.9084	0.9090	0.9407	0.757	0.8476	0.8614	0.8743	0.8749	0.8980	0.5425	0.6164	0.6503	0.6729	0.6947	0.7231	0.7418	0.6362
2-stage SFC (task1)	0.8979	0.9457	0.9517	0.9591	0.9610	0.9664	0.7975	0.8818	0.8962	0.9109	0.9187	0.9198	0.6477	0.7055	0.7200	0.7232	0.7484	0.7653	0.7972	0.7189
2-stage SFC (task2)	0.9162	0.9424	0.9530	0.9617	0.9633	0.9690	0.7997	0.8823	0.8945	0.9123	0.9236	0.9317	0.6498	0.6980	0.7202	0.7358	0.7498	0.7657	0.8020	0.7311
2-stage SFC (task1 + task2)	0.9167	0.9456	0.9571	0.9638	0.9658	0.9753	0.8135	0.8854	0.8931	0.9192	0.9257	0.9339	0.6525	0.7092	0.7168	0.7476	0.7519	0.7696	**0.8124**	0.7364
Joint SFC	**0.9231**	**0.9560**	**0.9644**	**0.9669**	**0.9712**	**0.9821**	**0.8270**	**0.9069**	**0.9103**	**0.9209**	**0.9323**	**0.9463**	**0.6697**	**0.7211**	**0.7254**	**0.7497**	**0.7593**	**0.7772**	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 (5/10/15/20/30/50 samples per class) while keeping the test set to be fixed. The highest scores for each data setting are marked in bold.

intents, in the same style made by real users through crowdsourcing (Larson et al. 2019). Here, we sample 5, 10, 15, 20, 30 and 50 training data points for each intention class label by random sampling for our experimental setup.

BANKING77 is a intention detection dataset for bank customer services. The raw dataset contains 13,084 data points for 77 class labels (Casanueva et al. 2020). Here, we sample 5, 10, 15, 20, 30, and 50 training data points for each class label by random sampling for our experimental setup.

4.2. Baselines

We choose 3 groups of typical baselines:

Group 1: single-task classification model. This group of baseline models are based on the network structure for classification model. As for non-pretrained neural network, we choose TextCNN (Kim 2014) 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 (Lan et al. 2019) for comparison. Since we use few-shot settings different from that of (Casanueva et al. 2020) for those intention detection datasets, we also use BERT-large tuned on our few-shot settings as an essential reference baseline for experimental results.

Group 2: single-task similarity model. This group of baseline models are based on network structure for sentence pair similarity model. 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 input query.

Group 3: SFC system with ablation study. This group of baseline is based on the network structure of two-stage SFC. 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.

4.3. 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.

In two-stage SFC, in equ. 9 the weights of two tasks are set as $\alpha_S = 0.125$ and $\alpha_K = 0.875$ in our setting. The reason for giving task 2 relatively more weights is that according to the ablation experiments we did in Section 4, task 2 contributes more to the overall performance. In joint SFC, in equ. 10, the weights of three tasks are set as $\alpha_S = 0.25$, $\alpha_K = 0.25$ and $\alpha_C = 0.5$ in our setting.

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 settings.

4.4. 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 dataset. 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. Furthermore, we also studied the relative performance among all the variants of SFC on different data settings to explore the

Dataset	Model	$K=3$ $P=20$	$K=5$ $P=10$	$K=10$ $P=5$	$K=15$ $P=4$	$K=20$ $P=3$
ITG	two-stage SFC	0.8034	0.8124	0.8008	0.7967	0.7934
	joint SFC	0.7986	0.8114	0.8010	0.7972	0.7918
Amazon-670k	two-stage SFC	0.7278	0.7364	0.7366	0.7328	0.7204
	joint SFC	0.7334	0.7445	0.7516	0.7344	0.7373

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.

benefits brought by multi-task training and the joint-model structure. With all the experimental results shown in Table 2, 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 joint-model structure. We conduct ablation experiments to study the benefits brought by multi-task training by comparing multi-task two-stage SFC with single-task two-stage SFC. We can see from Table 2 that two-stage SFC with only task 1 (sentence-pair similarity task) degrades by 0.80 percent on average and two-stage SFC with only task 2 (top-K classification task) degrades by 0.45 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 α_K to the loss $\mathcal{L}^K$ of task 2 when calculating the total loss $\mathcal{L}$. Also, we can see that the joint SFC steadily outperforms two-stage SFC on 5 diverse datasets by 0.95 percents on average. This observation supports the idea that the joint model structure can alleviate the limitation brought by the lack of interaction between similarity model and classification model in two-stage SFC. Therefore, for the following analysis, we will focus on the comparison between joint 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 intention detection datasets (ClINC150, BANKING77, HWU64) under few-shot settings(5-20 samples per class) indicate that our proposed joint SFC can achieve 2.47 percents average improvement over one of the most powerful baseline models, which is RoBERTa-large classification model. Moreover, the improvement in F-score becomes more and more prominent with the decrease of data sample number per class. Our joint 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 well-performed when 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. Furthermore, our proposed joint SFC can also achieve 2.04 percents average improvement over RoBERTa-large classification model on all the data settings including the ones with large scale (30-50 samples per class). This experimental result indicates that joint SFC can still steadily outperforms single-task pretrained model baseline even if the data size grows bigger. But the improvement is still more prominent when the data is scarce, which makes SFC an excellent model for low-resource chatbot scenario.

Comparison between label embedding and sentence pair embedding. Afterwards, we start to compare the performance of SFC with the text-label joint embedding baseline named as LEAM. As shown in Table 2, joint SFC outperforms LEAM by 11.90 percents on average F-score over 5 datasets. Besides, although the labels within the 3 diverse intention detection datasets (CLINC150, BANKING77, HWU64) are all being well-defined as the semantic summary of each intention, joint SFC still outperforms LEAM by 10.71 percents on average F-score. This observation indicates that it is usually quite hard to use a fixed label with only a few words as the representative of all the sentences within a class since there are always plenty of ways for people to express questions or queries about the same topic during a conversation with chatbot. Therefore, learning the semantic similarity between user query and sample sentences through sentence pair embedding can be more reliable than making decision according to the semantic relationship between user query and class labels through label embedding.

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, joint SFC outperforms TextCNN by 21.52 percents, ALBERT-xxlarge by 7.99 percents[2], BERT-large by 7.73 percents, RoBERTa-base by 3.79 percents, RoBERTa-large by 2.04 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 joint SFC achieves the improvement of 7.09 percents on average. Moreover, joint 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

[2]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.

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 joint 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 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 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. However, in spite of the effect on evaluation performance brought by different settings of hyperparameters, joint SFC still steadily outperforms two-stage SFC by 0.41 percents in overall average F-score on ITG and Amazon-670k datasets.

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

5. 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.

6. REFERENCES

Bamler, R.; and Mandt, S. 2020. Extreme Classification via Adversarial Softmax Approximation. *arXiv preprint arXiv:2002.06298* .

Barkan, O.; Razin, N.; Malkiel, I.; Katz, O.; Caciularu, A.; and Koenigstein, N. 2020. Scalable Attentive Sentence Pair Modeling via Distilled Sentence Embedding. In *AAAI*, 3235–3242.

Bhatia, K.; Dahiya, K.; Jain, H.; Prabhu, Y.; and Varma, M. 2016. The extreme classification repository: multi-label datasets & code. *URL http://manikvarma. org/downloads/XC/XMLRepository. html* .

Caruana, R. 1993. Multitask Learning: A Knowledge-Based Source of Inductive Bias ICML. *Google Scholar Google Scholar Digital Library Digital Library* .

Casanueva, I.; Temčinas, T.; Gerz, D.; Henderson, M.; and Vulić, I. 2020. Efficient Intent Detection with Dual Sentence Encoders. *arXiv preprint arXiv:2003.04807* .

Chen, J.; Hu, Y.; Liu, J.; Xiao, Y.; and Jiang, H. 2019. Deep short text classification with knowledge powered attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6252–6259.

Collobert, R.; and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, 160–167.

Conneau, A.; Schwenk, H.; Barrault, L.; and Lecun, Y. 2016. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781* .

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* .

Divya, M. S.; and Goyal, S. K. 2013. ElasticSearch: An advanced and quick search technique to handle voluminous data. *Compusoft* 2(6): 171.

Du, C.; Chen, Z.; Feng, F.; Zhu, L.; Gan, T.; and Nie, L. 2019. Explicit interaction model towards text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6359–6366.

Hua, W.; Wang, Z.; Wang, H.; Zheng, K.; and Zhou, X. 2015. Short text understanding through lexical-semantic analysis. In *2015 IEEE 31st International Conference on Data Engineering*, 495–506. IEEE.

Iyer, S.; Dandekar, N.; and Csernai, K. 2017. First quora dataset release: Question pairs. *data. quora. com* .

Jafarpour, S.; Burges, C. J.; and Ritter, A. 2010. Filter, rank, and transfer the knowledge: Learning to chat. *Advances in Ranking* 10: 2329–9290.

Kim, Y. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; and Soricut, R. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* .

Larson, S.; Mahendran, A.; Peper, J. J.; Clarke, C.; Lee, A.; Hill, P.; Kummerfeld, J. K.; Leach, K.; Laurenzano, M. A.; Tang, L.; et al. 2019. An evaluation dataset for

intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027* .

Leuski, A.; and Traum, D. 2011. NPCEditor: Creating virtual human dialogue using information retrieval techniques. *Ai Magazine* 32(2): 42–56.

Liu, P.; Qiu, X.; and Huang, X. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101* .

Liu, X.; Eshghi, A.; Swietojanski, P.; and Rieser, V. 2019a. Benchmarking natural language understanding services for building conversational agents. *arXiv preprint arXiv:1903.05566* .

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* .

Madabushi, H. T.; Kochkina, E.; and Castelle, M. 2020. Cost-Sensitive BERT for Generalisable Sentence Classification with Imbalanced Data. *arXiv preprint arXiv:2003.11563* .

Mnasri, M. 2019. Recent advances in conversational NLP: Towards the standardization of Chatbot building. *arXiv preprint arXiv:1903.09025* .

Mousa, A.; and Schuller, B. 2017. 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*, 1023–1032.

Phan, X.-H.; Nguyen, L.-M.; and Horiguchi, S. 2008. 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*, 91–100.

Song, G.; Ye, Y.; Du, X.; Huang, X.; and Bie, S. 2014. Short text classification: A survey. *Journal of multimedia* 9(5): 635.

Sriram, B.; Fuhry, D.; Demir, E.; Ferhatosmanoglu, H.; and Demirbas, M. 2010. 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*, 841–842.

Sun, C.; Qiu, X.; Xu, Y.; and Huang, X. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, 194–206. Springer.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, G.; Li, C.; Wang, W.; Zhang, Y.; Shen, D.; Zhang, X.; Henao, R.; and Carin, L. 2018. Joint embedding of words and labels for text classification. *arXiv preprint arXiv:1805.04174* .

Wen, T.-H.; Vandyke, D.; Mrksic, N.; Gasic, M.; Rojas-Barahona, L. M.; Su, P.-H.; Ultes, S.; and Young, S. 2016. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562* .

Yan, R.; Cao, X.-b.; and Li, K. 2009. Dynamic assembly classification algorithm for short text. *Acta Electronica Sinica* 37(5): 1019–1024.

Yu, M.; Guo, X.; Yi, J.; Chang, S.; Potdar, S.; Cheng, Y.; Tesauro, G.; Wang, H.; and Zhou, B. 2018. Diverse few-shot text classification with multiple metrics. *arXiv preprint arXiv:1805.07513* .

Zhang, H.; Xiao, L.; Chen, W.; Wang, Y.; and Jin, Y. 2017. Multi-task label embedding for text classification. *arXiv preprint arXiv:1710.07210* .

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, 649–657.