A Practical 2-step Approach to Assist Enterprise Question-Answering Live Chat

Ling-Yen Liao
Bloomberg
lliao1@bloomberg.net

Tarec Fares
Bloomberg
tfares1@bloomberg.net

Abstract

Live chat in customer service platforms is critical for serving clients online. For multi-turn question-answering live chat, typical Question Answering systems are single-turn and focus on factoid questions; alternatively, modeling as goal-oriented dialogue limits us to narrower domains. Motivated by these challenges, we develop a new approach based on a framework from a different discipline: Community Question Answering. Specifically, we opt to divide and conquer the task into two sub-tasks: (1) Question-Question Similarity, where we gain more than 9% absolute improvement in F_1 over baseline; and (2) Answer Utterances Extraction, where we achieve a high F_1 score of 87% for this new sub-task. Further, our user engagement metrics reveal how the enterprise support representatives benefit from the 2-step approach we deployed to production.

1 Introduction

With technological advances, more customers are moving online, and so must customer service (Armington, 2019). Live chat plays a critical role in serving customers online, and numerous service organizations provide live chat to help customers today. Because human-to-human interactions are preferred over chatbots (Press, 2019; Shell and Buell, 2019), and enterprise live chat is typically human-to-human, there are tremendous opportunities in assisting live chat to efficiently answer customers' questions.

We are interested in multi-turn question-answering live chat that is common among enterprise customer services. We argue that to model the problem as a Community Question Answering (CQA) problem over other choices like typical Question Answering (QA) systems or goal-oriented dialogue systems has several advantages. QA systems are traditionally single-turn and focus on factoid questions with short answers. Alternatively, goal-oriented dialogue systems, whether modeling with a pipeline or end-to-end methods, there is limited evidence that they work well for the broader domain of enterprise question-answering live chats.

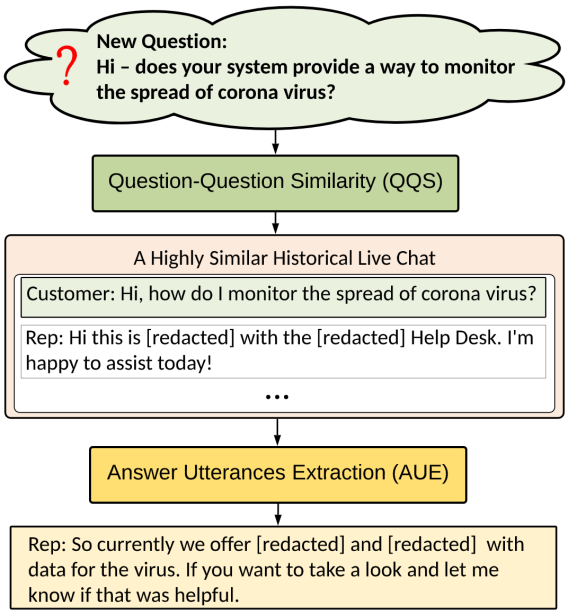

Figure 1: Overview of our 2-step method. A customer question is first matched to a highly similar historical chat (QQS), then the answer is extracted from the matched chat (AUE).

Motivated by these challenges and consider real-world practicality, we propose a new approach to model multi-turn question-answering live chat as a CQA problem, and we focus on answer utterances for evaluation. Our approach is general and the setup is flexible so it can be easily ported to other domains.

The aim of this paper is to assist enterprise support representatives (reps) in answering live chats that are across several knowledge domains. The primary goal is to surface answers for a new question asked by a customer, especially if the rep is not familiar with the question; the secondary goal is

Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, pages 457–468
July 29–31, 2021.

to provide reps a tool to explore questions closely related to the new question hence enhance their domain expertise.

Our key contributions are:

1. We frame the multi-turn question-answering live chat problem as a CQA problem, which is more suitable for real-world use than QA systems and more generalizable than goal-oriented dialogue systems;

2. We present a new sub-task Answer Utterances Extraction (AUE) that focuses on answer utterances and we show that an approach incorporates domain adaptation and dialogue features is effective for this sub-task;

3. Our approach outperforms the corresponding baselines, and the user engagement statistics present how users benefit from the 2-step method we deployed to production with low latency.

2 Related Work

Dialogue systems can be categorized as (1) Question answering (QA) systems, (2) Goal-oriented or task-oriented dialogue systems, and (3) Chatbots or social bots (Gao et al., 2019; Deriu et al., 2020).

QA Systems. Traditional QA systems assume a single-turn setting (Fader et al., 2013). For multi-turn QA systems, one approach is to employ a pipelined architecture like a task-oriented dialogue system (Dhingra et al., 2017); and the pipeline includes either a knowledge base (KB) or a machine reading comprehension (MRC) model (Seo et al., 2017; Gao et al., 2019). Both KB and MRC components are also common in single-turn QA systems.

In KB based QA systems the answer is usually factual and is identified using an entity-centric KB or knowledge graph (KG), after semantic parsing (Iyyer et al., 2017). Also, in those systems a limited number of questions can be answered and they are typically curated (Chen and Yih, 2020).

On the other hand, the typical setup for an open-domain QA system, is to first have a retriever, that uses sparse or dense representations to select relevant passages from an external knowledge source (Karpukhin et al., 2020), then a MRC model, known as an extractive reader, to do span extraction from those passages and mark where the answers are (Rajpurkar et al., 2016; Choi et al., 2018). This is known as a retriever-reader framework (Chen et al., 2017a; Wang et al., 2019; Yang et al., 2019). The reader from the retriever-reader framework can be replaced with a generator to generate answers out of the relevant passages, this system is known as a retriever-generator framework (Lewis et al., 2020; Izacard and Grave, 2021; Weng, 2020). Both frameworks can be trained end-to-end.

One can recommend solving our problem with the above described open-domain QA system; however, such an approach would require a predetermined knowledge source from which answers are extracted or generated. Enterprise customer service departments typically have "help documents" as knowledge sources, but what makes it difficult to use an open-domain QA system approach is that those sources are usually not comprehensive enough.

Finally, all the previously described approaches, even with recent advances that use very large pre-trained language models (Radford et al., 2019; Brown et al., 2020), have limited evidence that shows that they work well for long-answer non-factoid questions that are common among enterprise customer services (Raffel et al., 2020; Chen and Yih, 2020).

Goal-Oriented Dialogue Systems. Conversely, multi-turn question-answering live chats could be viewed as goal-oriented dialogues in which the task is to answer customers' questions. Goal-oriented dialogue systems are typically implemented with a pipelined architecture (Chen et al., 2017b), which consists of different modules for natural language understanding (Goo et al., 2018), dialogue state tracking (Lee and Stent, 2016), dialogue policy (Takanobu et al., 2019), and natural language generation (Wen et al., 2015). End-to-end methods have also emerged to minimize the need of domain-specific feature engineering (Zhao and Eskenazi, 2016; Bordes et al., 2017; Wen et al., 2017; Li et al., 2017; Ham et al., 2020). However, most of these methods are applied on specific domains that have limited intents and detectable slots. Enterprise question-answering live chats can have thousands of different intents and not every question has detectable slots.

Chatbots. Chatbots or social bots have gone beyond chit-chat, can be further categorized as generative methods and retrieval-based methods. These methods are applied to goal-oriented dialogues as well, aiming to directly select or generate a

dialogue response given an input (Gandhe and Traum, 2010; Swanson et al., 2019; Henderson et al., 2019).

Evaluation of Dialogue Systems. For evaluation, goal-oriented dialogue systems can be evaluated to measure task-success and dialogue efficiency (Walker et al., 1997; Takanobu et al., 2020; Deriu et al., 2020). Retrieval-based chatbots often report performance on *Next Utterance Classification*, to test if a next utterance can be correctly selected given the chat context (Lowe et al., 2015; Henderson et al., 2019; Swanson et al., 2019). Conversational QA systems, on the other hand, are evaluated based on the correctness of their answers and the naturalness of the conversations (Reddy et al., 2019; Deriu et al., 2020).

In the following, we describe our CQA approach and how we evaluate it.

3 Approach

The main CQA task is defined in Nakov et al. (2016) as "given (i) a new question and (ii) a large collection of question-comment threads created by a user community, rank the comments that are most useful for answering the new question". Quora and Stack Overflow are examples of CQA websites.

The CQA task has three sub-tasks:

- Question-Comment Similarity (Subtask A): to rank the usefulness of comments below a question in a CQA forum;
- Question-Question Similarity (Subtask B): to find previously asked similar questions;
- Question-External Comment Similarity (Subtask C): to rank comments from other questions for answering a new question.

Subtask C is built upon Subtask A and B.

If we replace *Comment* from the CQA problem with *Utterance* for a live chat, we can view a multi-turn live chat as a question-comment thread. Subtask A then becomes Question-Within Chat Utterance Similarity and Subtask B remains Question-Question Similarity (QQS), where we describe a more robust setup for live chat. We investigate Subtask A and present a new task Answer Utterances Extraction (AUE) that is better suited for question-answering live chat. Figure 1 illustrates our 2-step method of QQS and AUE.

Our approach does not require a KB or a knowledge source with answer passages, that most QA systems require, instead our approach needs only historical chat sessions, which most enterprise customer services have available. Moreover, our approach is flexible, because it is comparing question similarity, and does not rely on specific question intent or slots, and that makes it more generalizable than goal-oriented dialogue systems.

In the next two sections we explain the two subtasks and our approaches in details.

4 Question-Question Similarity

We define the QQS sub-task as: given a new question consisting of m utterances from a customer, obtain n historical chats whose questions are highly similar to the new question. Highly similar questions are defined as having semantic equivalence or high syntactic overlap.

This sub-task is similar to Subtask B from SemEval–2016/2017 Task 3 Community Question Answering related work (Nakov et al., 2016, 2017; Yang et al., 2018) and learning to rank (Joachims, 2002; Surdeanu et al., 2008). The practice of having a machine learning model on top of a search engine is common in the information retrieval (IR) community, it is done also for speed reasons, as it is too slow to calculate the similarity scores between a new question and all historical questions.

To adapt this approach to live chats, the main difference between a CQA question-comment thread and a live chat for this sub-task is that we know which text is the question in a question-comment thread, and the question is typically stand-alone and complete. For a live chat, it's unknown which utterances are the question, a customer question could start with a salutation, and with subsequent utterances together form a complete question.

4.1 Practical Considerations

Table 1: Enterprise live chat characteristics.

Statistic	Value
Initial question is a complete question	58%
Live chats have more than 1 new question asked	<10%
At which turn is the first answer utterance	7
First utterance is a salutation (i.e. "hi, hello")	>10%

Our approach concerns an enterprise customer service live chat system. When a customer creates a live chat request, they enter their question in free-form text and are then routed to a support rep to start their chat. The initial question may be a complete question itself, or it may take a few

more turns/utterances to complete. From **Utterance Annotation** (Section 6.2), we found that in 58% of chats, the initial question is complete; the utterance itself represents a complete question, customers may provide additional information, but the question can be answered without the additional information. Therefore searching historical chats matching on first utterances should cover the bulk of chats, and matching beyond first utterances will increase coverage.

In addition, less than 10% of the chats have more than one question asked; customers may follow up around the topic but rarely ask a completely new question, thus focusing on the first question asked (which could consist of multiple utterances) is reasonable. Finally, on average the 7th utterance is where reps start to give answers (approximately the 3rd customer utterance), hence we want to provide assistance before that. These practical considerations are summarized in Table 1 and drive how we develop the QQS algorithm designed for live chat.

4.2 QQS Algorithm

Our goal is to assist enterprise support reps promptly, therefore the QQS algorithm starts with the first utterance. The same algorithm is utilized again for subsequent utterances until the 3rd customer utterance, with a $query$ consisting of a concatenation of customer utterances up to that point. We use a salutation detector (Section 4.3) to ignore utterances that are not meaningful questions, and then pass the query to a search engine to obtain the top 10 results that are matched using the first utterances of historical chats. The search results are scored against the query with a chosen similarity model (Section 4.4), and search results below a chosen threshold value (Section 7.1) are removed. Finally, the highest scoring search results up to n are returned, $n \in [0, 2]$. Typically n is small otherwise the support reps are overwhelmed.

4.3 Salutation Detector

Salutations and uninformative utterances account for over 10% of the first utterances of our chats, and a rule-based method can detect them accurately. Our salutation detector is implemented using a context-free grammar parser[1] with handcrafted grammar rules to capture uninformative utterances like "hi", "hello", "help desk please", "hi i have a question", etc.

[1] https://github.com/lark-parser/lark

4.4 Similarity Models

To measure the similarity between two initial questions, both unsupervised and supervised methods were considered. For the unsupervised method, we use a word2vec model (Mikolov et al., 2013) trained on live chat initial questions. Similarity is measured using `cosine` of two questions represented as vectors. The model is denoted as `Word2Vec-COS`, and `COS` stands for `cosine`. For the supervised method, the BERT$_{BASE}$ pretrained model (Devlin et al., 2019) is fine-tuned with question-question pairs to classify a pair of texts as *Similar* or *NotSimilar* with a similarity score. The model is denoted as `BERT-QQS`. Additional model details are described in Section 6.1.

5 Answer Utterances Extraction

After the QQS algorithm, n highly similar historical questions and their chats are obtained. For each chat we proceed with the second sub-task, which is defined as: given a chat consisting of m utterances, identify the answer utterance(s).

The main difference between a question-comment thread from a CQA forum and a live chat is that a comment from a question-comment thread is usually stand-alone, but for a live chat it could take multiple turns to form a complete meaning from each speaker. We also do not re-rank utterances like a typical CQA approach, because reorder utterances will perturb a complete answer that is spanned across multiple utterances. In addition, users in a question-common thread can up-vote a correct comment/answer but for live chats we don't have such a mechanism.

For this sub-task, an unsupervised method and a supervised method were developed. The unsupervised method selects the most similar utterances from the rep with respect to the question, an approach inspired by CQA. Our work is also related to extractive summarization where the most important sentences in a document are identified (Narayan et al., 2018; Liu and Lapata, 2019), so we include an unsupervised baseline result using Latent Semantic Analysis (LSA) for comparison.

The supervised method incorporates dialogue specific features to classify a candidate utterance, which is closer to the problem of written dialogue act classification (Kim et al., 2010), with a new set of dialogue acts for enterprise live chat.

Table 2: An example of AdaptaBERT-AUE input after pre-processing. This should output *NotAnswer*.

Input Type	Input Content
Chat Context	[CLNT] good morning , [ENTER] [CLNT] how can i get usd / jp ##y swap rate for 3 and 5 years ? [ENTER] [REP] hello there [redacted] ! [ENTER] [REP] good day to you . [ENTER] [REP] please run [redacted] [ENTER] [REP] on the lower left you can click into the different types of swap ##s . [ENTER] ...
Candidate Utterance	[REP] good day to you . [ENTER]

5.1 Question-Within Chat Utterance Similarity

This is an unsupervised method and closely related to Subtask A from SemEval–2016/2017 Task 3 Community Question Answering related work (Nakov et al., 2015, 2016, 2017; Lai et al., 2018).

We have a historical chat and its matched initial question obtained from the QQS algorithm. The initial question is then scored with all utterances from the rep using the same Word2Vec-COS model from Section 4.4. The highest scoring x rep utterances, which are the most similar utterances to the question, are assumed answers. We set x to be half of total rep utterances, with an intuition to summarize a chat by half. The indices of the x utterances in a chat are returned, subsequently can be highlighted in a chat.

5.2 Latent Semantic Analysis

For an additional comparison, we include an unsupervised baseline method's result using LSA for extractive summarization (Gong and Liu, 2001; Steinberger and Ježek, 2004), since the AUE subtask can be set up as an extractive summarization problem. We treat a whole chat conversation as a document and select the x most semantically important rep utterances from the document as the answer; and like the previous section, we set x to be half of total rep utterances.

5.3 AdaptaBERT-AUE

This supervised method takes all utterances from a historical chat obtained from the QQS algorithm, and outputs scores to indicate each utterance's probability being part of the complete answer.

We first conduct unsupervised domain-adaptive fine-tuning (Dai and Le, 2015; Howard and Ruder, 2018) on a pre-trained BERT$_{BASE}$ model (Devlin et al., 2019) to adapt to our dialogue domain, following the work in Han and Eisenstein (2019), the model is denoted as AdaptaBERT. We then perform task-specific fine-tuning on AdaptaBERT to take in a chat context and a candidate utterance as input, and classify the candidate utterance as *Answer* or *NotAnswer*, denoted as AdaptaBERT-AUE.

For both domain-adaptive and task-specific fine-tuning we extend the BERT vocabulary and procedure to include three dialogue specific tokens: (1) [CLNT] represents speaker is customer, (2) [REP] represents speaker is rep, and (3) [ENTER] refers to when a user hits the enter/return key to submit after finishing their utterance. A partial example of an input for task-specific fine-tuning can be seen in Table 2.

6 Experimental Setup

We used human annotations to evaluate our models and algorithms. Data was sampled from a large proprietary enterprise live chat dataset, containing over 3 million English chats per year. We used English chats to evaluate our methods; however the approach is not limited to English.

6.1 QQS Data and Models

Two annotation sets are used to evaluate the subtask.

QQS Pair. We have live chat questions each labeled with one of over 1,000 intents. We consider pairs of questions to be *Similar* if they have the same intent, and *NotSimilar* otherwise. The data is subsampled so there are 50% *Similar* pairs and 50% *NotSimilar* pairs. Out of these *NotSimilar* pairs, 50% are *close* negatives, defined as question-question pairs with overlapping vocabularies but were not labeled as the same intent. A total of 1 million question-question pairs are sampled, and the data is split with 80% for training and 20% for validation.

Because this data is not a random sample from live chats, it is used to train and validate the BERT-QQS model, but not for testing.

Search Result Annotation. To obtain test data, we conduct an annotation task with randomly sampled live chat first utterances. With these questions we run through the QQS algorithm until search results are returned, and questions yielding no results from the algorithm are excluded from the sample.

We design the annotation task in two parts. First, we ask annotators to evaluate if a question is clear or not, defined as whether a complete question is asked. This is to identify questions like "I have a question about excel formula" or "can you help me with my report", which are not salutations but still require clarification before they can be answered.

If a question is clear, then annotators continue to consider its search results, and select search results that are equivalent or overlapping with the question. If a question is labeled as not clear, then all search results are considered not equivalent to the question.

A total of 1,076 questions were annotated, resulting in 10,760 (question, search result) pairs with labels. Each question was annotated by 2 annotators. For inter-annotator agreement, the overall Krippendorff's Alpha is 0.46, which is considered moderate agreement (Artstein and Poesio, 2008). The final label of a (question, search result) pair is considered positive if it is selected by at least one annotator. The final label distributions are 28% positive and 72% negative.

The following three models are evaluated.

- Solr Baseline is Apache Solr with a custom indexing pipeline consists of Lucene's standard tokenizer, stop words filter, lower case filter, English possessive filter, keyword marker filter, and Porter stemmer filter. The query pipeline is the same as the indexing pipeline with an additional synonym filter factory. Document scoring uses Lucene's TFIDFSimilarity[2], where documents "approved" by Boolean model of IR are scored by tf-idf with cosine similarity. We use this as a baseline to evaluate QQS, where the Solr rank is directly used to rank results. All other similarity models are applied on top this IR method.

- Word2Vec-COS is our unsupervised baseline method. Trained with 2.8 million first utterances using Google's word2vec executable[3] with the following parameters: skip-gram architecture, window size is 5, and dimension is 300. To measure the similarity between two input texts, the text is first preprocessed to remove stop words, and words that are adjectives, nouns, proper nouns, and verbs are kept. The text is then represented as a vector by averaging over its word vectors; finally, we calculate cosine of the two vectors.

- BERT-QQS is a fine-tuned BERT$_{BASE}$ model that classifies a pair of questions to output a similarity score. We used Google's BERT code[4] to fine-tune with default hyper-parameters. Trained/fine-tuned and validated using **QQS Pair**.

6.2 AUE Data and Models

We use one dataset to evaluate this sub-task.

Utterance Annotation. An annotation task is conducted to label live chat utterances. Live chats are randomly sampled, and each utterance is labeled as one of the following dialogue acts: *QuestionStartComplete*, *QuestionStart*, *QuestionRelevant*, *QuestionComplete*, *Answer* or *Other*. We denote *Question** to include all question labels.

An utterance that is a complete question itself is labeled as *QuestionStartComplete*. A question takes multiple turns to complete is labeled as *QuestionStart* for its first utterance and *QuestionComplete* for its last utterance, and *QuestionRelevant* in-between. An utterance contributes to the solution is labeled as *Answer*, and the rest are labeled as *Other*. An example can be seen in Table 3.

There are total 656 chats and 12,310 utterances, and 21% of the chats were annotated by 2 to 6 annotators to calculate inter-annotator agreement. The Krippendorff's Alpha is 0.59, which is considered moderate agreement and close to substantial agreement (Artstein and Poesio, 2008). We take the majority vote as the final label for these utterances. The final label distributions of all utterances are 22% *Question**, 28% *Answer*, and 51% *Other*.

The following four models are evaluated.

[2] https://lucene.apache.org/core/5_5_5/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html

[3] https://code.google.com/archive/p/word2vec/

[4] https://github.com/google-research/bert

Table 3: An example Utterance Annotation. The example has been lightly edited.

Speaker	Utterance	Label
Customer	How do I setup a email thread to top coronavirus news?	*QuestionStartComplete*
Rep	Hello you have reached [redacted]. Please allow me a moment to check on this for you.	*Other*
Customer	Are you still there	*Other*
Rep	Please go to [redacted] and click into [redacted] under Sources and search for Coronavirus	*Answer*
Rep	A better alternative may actually be to check [redacted] and search "coronavirus", and subscribe to one of those	*Answer*
Rep	You can preview the kinds of stories they provide, and set up delivery preferences	*Answer*
Customer	Thanks	*Other*
Customer	Do I want deliver to alert catcher	*QuestionRelevant*
Customer	I think I'm set actually thanks	*Other*
Customer	Appreciate it	*Other*
Rep	No problem! If you have any further questions, please feel free to return to the chat.	*Other*

- `Word2Vec-COS` is the same model used in QQS, see Section 6.1. Testing is done with **Utterance Annotation** to select the most similar rep utterances to the question, as described in Section 5.1.

- `LSA-Sumy` is an unsupervised baseline method of extractive summarization using LSA. We use the sumy (Belica, 2013) Python package[5] implementation, while utilizing our own tokenization and segmentation methods. Testing is done with **Utterance Annotation** to select the most semantically important rep utterances, as described in Section 5.2.

- `AdaptaBERT-AUE` is a result of both domain-adaptive and task-specific fine-tuning, and we extended BERT_{BASE} to account for dialogue specific tokens. The model classifies a candidate utterance along with its chat context to output a score to indicate how likely the candidate utterance is *Answer*. We use 1.3 million whole chats for domain-adaptive fine-tuning. 5-fold cross-validated for task-specific fine-tuning with **Utterance Annotation**. Default hyper-parameters were used with maximum sequence length being 512 to account for chat context.

- `BERT-AUE` is `AdaptaBERT-AUE` without the unsupervised domain-adaptive fine-tuning step.

[5] https://miso-belica.github.io/sumy/

7 Results

We achieve a high F1 score of 86.83% on the AUE task, and significantly outperform the unsupervised methods on the QQS task.

7.1 QQS Evaluation

Table 4: Test on all (question, search result) pairs with different models.

Model	Threshold	Precision	Recall	F_1
`Solr Baseline`	N/A	27.87	100	43.59
`Word2Vec-COS`	0.5	28.19	100	43.98
`Word2Vec-COS`	0.7	29.78	95.10	45.36
`Word2Vec-COS`	0.9	40.51	13.80	20.59
`BERT-QQS`	0.5	44.27	67.02	**53.32**
`BERT-QQS`	0.7	47.98	54.28	50.94
`BERT-QQS`	0.9	54.77	28.54	37.53

For `BERT-QQS` the accuracy is 89% from validation of **QQS Pair**. We observed that the accuracy started at 80% with 20,000 question-question pairs and increased as the number of pair increases.

To test the QQS algorithm with different similarity models, we evaluate all 10,760 (question, search result) pairs from **Search Result Annotation**. Each pair has a prediction/score from different similarity models, and a final label to indicate positive or negative. As can be seen in Table 4, because all the pairs are search results, for `Solr Baseline` (row 1), all pairs are considered as predicted positive, therefore recall is 100% and threshold is not applicable (N/A). Similarity models like `Word2Vec-COS` and `BERT-QQS` quantify similarity with a score, and we use different predefined probability threshold values to calculate precision, recall, and F_1. `BERT-QQS` (row 5) sig-

Table 5: Ablation Study of AdaptaBERT-AUE (5-fold cross validation)

Input Features	F_1
Candidate utterance text only	79.59
Candidate utterance text and speaker	84.23
Whole chat text as context + candidate utterance text	82.98
Whole chat text as context (shuffle utterance order) + candidate utterance text	81.25
Whole chat text and speaker as context + candidate utterance text and speaker (AdaptaBERT-AUE)	86.83

nificantly improves Solr Baseline on the F_1 score for more than 9 points, indicating that it can select highly similar questions. Word2Vec-COS (row 3) performs only slightly better than Solr Baseline.

BERT-QQS with a higher threshold value can improve precision, which is a primary factor to evaluate readiness for production systems. Enterprise live chat systems often has precision requirement and sometimes are willing to sacrifice recall for precision.

7.2 AUE Evaluation

To evaluate performance of AUE, we use **Utterance Annotation**. We directly test the algorithm from Section 5.1 with Word2Vec-COS on this dataset. Basing on the output indices from the algorithm, we marked these utterances as predicted *Answer* and the rest marked as predicted *NotAnswer*. The first utterance marked as *QuestionStartComplete* or the first occurrence between *QuestionStart* and *QuestionComplete* is used as the question text.

As can be seen in Table 6 (row 1), the Word2Vec-COS attains a decent F_1 score, especially since it is an unsupervised method. For LSA-Sumy, a LSA based extractive summarization baseline method described in Section 5.2, is performing worse than the similarity based method Word2Vec-COS as can be seen in row 2 versus row 1 of Table 6.

Table 6: Unsupervised and supervised methods.

Model	F_1
Word2Vec-COS (algorithm from Section 5.1)	63.92
LSA-Sumy (algorithm from Section 5.2)	58.95
BERT-AUE (5-fold cross validation)	82.40
AdaptaBERT-AUE (5-fold cross validation)	**86.83**

For BERT-AUE and AdaptaBERT-AUE, we treat labels *Question** and *Other* as *NotAnswer*. After 5-fold cross-validation, the F_1 score is averaged from all folds and listed in Table 6. Unsupervised domain-adaptive fine-tuning accounts for more than 4 points in F_1 (row 3 versus row 4).

7.3 Ablation Study of AdaptaBERT-AUE

To understand more about how different features contribute to the AdaptaBERT-AUE model performance, we conduct an ablation study to include different features for task-specific fine-tuning.

As can be seen in Table 5, merely the text of the candidate utterance (row 1), without any context or speaker information, brings us to a F_1 score of 79.59%. With just candidate utterance text, it cannot be argued that the model is learning text similarities like Word2Vec-COS with the algorithm from Section 5.1. The bulk of the AdaptaBERT-AUE performance comes from candidate utterance text solely. Adding speaker features (row 1 versus row 2) contributes about 5 points of F_1, which is significant. The presence of chat context features (row 1 versus row 4) and the context in order or not (row 3 versus row 4) result in F_1 differences moderately. Speaker features contribute to the F_1 score more than whole chat features (row 2 and row 3 versus row 1).

8 Production System

To conclude, we describe our production system. We deployed the BERT-QQS model from Section 6.1 and used all **Utterance Annotation** to train a AdaptaBERT-AUE model for production.

A pilot application is currently employed in assisting several hundred enterprise support reps on a daily basis. This real-time application displays up to two highly similar historical questions to reps (QQS), and upon clicking into, answer utterances are highlighted with the whole chat shown (AUE).

Inference time is crucial because our production system is serving reps in real time. To harness the power of graphics processing units (GPU) for model serving, we use KFServing[6] so that different parts of the inference system can be scaled independently. When serving the models on production, each pair of texts takes about 20 milliseconds for BERT-QQS and about 40 milliseconds for

[6] https://github.com/kubeflow/kfserving

AdaptaBERT-AUE on one GPU to do inference.

8.1 User Engagement

We tracked the following user interactions after deploying the pilot application to production.

- **Weekly question volume** refers to the weekly total number of questions from customers that the reps are enabled for the application.
- **Coverage (trigger rate)** refers to the percentage of questions triggered at least one matched historical chat from the QQS algorithm. This measures the overall impact of the system.
- **Click rate** refers to the percentage of questions that the reps clicked on any suggestions (we display up to two historical chats). This is to measure the impact and performance of the QQS algorithm.
- **Paste rate** refers to the percentage of questions that the reps clicked into any suggested chat (we display up to two historical chats) and then copied/pasted from it (answer utterances were highlighted). This is to measure the impact and performance end-to-end for the 2-step method of QQS and AUE.

Table 7: User interaction statistics.

Statistic	Value
Weekly question volume	Approximately 40,000
Coverage (trigger rate)	49%
Click rate (of triggers)	37%
Paste rate (of clicks)	27%

From Table 7, we can see that our approach covers about half of the live chats (49%, row 2), and more than one in three questions (37%, row 3), our suggestions are used. In addition, for these questions their suggested chats were clicked, 27% of them the suggestions are directly copied/pasted by the reps in answering customers questions (row 4).

Click rate is related to the QQS performance, but reps may not click on a suggestion if they already knew the answer to the question. For paste rate, we observed that reps sometimes read the suggested chat/answer and type their answers to customize their response to customers, and this behavior is harder to track. Therefore the paste rate is a lower bound to reflect the actual usage.

9 Conclusion

We have demonstrated how to adapt the Community Question Answering (CQA) framework to assist question-answering live chat, effectively and efficiently. For the QQS sub-task, where we use a robust setup for live chat, attain more than 9% absolute improvement in F_1 over baseline; we achieve a high F_1 score of 87% for the newly presented AUE sub-task, using unsupervised domain adaptive fine-tuning designed for live chat. Production user engagement data gathered from our real-time application showcase how the 2-step approach can influence the enterprise customer service industry in training and staffing for the support reps.

Our approach is broadly applicable, but it may not be the most preferred solution for every type of question. Business considerations must be taken when one is selecting their QA approach. For example, a question about a specific software problem may be answered with a pre-defined multi-turn template from a goal-oriented dialogue system to help guide a customer through a re-installation process. In contrast, with our approach, the answer utterances that contain the troubleshooting steps in a historical chat will be highlighted for the rep to use and guide the customer through the installation process. A template-based goal-oriented dialogue system could cover only task-oriented questions (e.g. software re-installation question intent), and if done well does not need rep involvement. Our CQA inspired approach and goal-oriented dialogue systems complement each other.

Future work will be automating annotation process through user interactions, qualitative analysis of user engagement data, and question-answering for longer chats midstream.

10 Ethical Considerations

All the work in this paper was done using anonymized user data, to respect the privacy of both participants in each conversation.

Acknowledgments

We thank the enterprise customer service desk and our Engineering managers for the continuous support. We are grateful for Amanda Stent's guidance; and we thank Carmeline Dsilva, Steven Butler, Maria Pershina, and Ari Silburt for the invaluable feedback on earlier versions of this paper. We also thank the anonymous reviewers for their helpful comments.

References

Julian Armington. 2019. Evolving online customer service: What your company needs to know.

Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Michal Belica. 2013. Metody sumarizace dokumentů na webu.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2017. Learning end-to-end goal-oriented dialog. In *5th International Conference on Learning Representations, ICLR 2017*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Danqi Chen and Wen-tau Yih. 2020. Open-domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017b. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, volume 28.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2020. Survey on evaluation methods for dialogue systems. *Artificial Intelligence Review*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Volume 1: Long Papers*, pages 484–495.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1608–1618.

Sudeep Gandhe and David Traum. 2010. I've said it before, and I'll say it again: An empirical investigation of the upper bound of the selection approach to dialogue. In *Proceedings of the SIGDIAL 2010 Conference*, pages 245–248.

Jianfeng Gao, Michel Galley, and Lihong Li. 2019. Neural approaches to conversational AI. *Foundations and Trends® in Information Retrieval*, 13(2-3):127–298.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 19–25.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Donghoon Ham, Jeong-Gwan Lee, Youngsoo Jang, and Kee-Eung Kim. 2020. End-to-end neural pipeline for goal-oriented dialogue systems using GPT-2. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 583–592.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248.

Matthew Henderson, Ivan Vulić, Daniela Gerz, Iñigo Casanueva, Paweł Budzianowski, Sam Coope,

Georgios Spithourakis, Tsung-Hsien Wen, Nikola Mrkšić, and Pei-Hao Su. 2019. Training neural response selection for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5392–5404.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Su Nam Kim, Lawrence Cavedon, and Timothy Baldwin. 2010. Classifying dialogue acts in one-on-one live chats. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 862–871.

Tuan Manh Lai, Trung Bui, and Sheng Li. 2018. A review on deep learning techniques applied to answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2132–2144.

Sungjin Lee and Amanda Stent. 2016. Task lineages: Dialog state tracking for flexible interaction. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 11–21.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 733–743, Taiwan.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26.

Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 27–48.

Preslav Nakov, Lluís Màrquez, Walid Magdy, Alessandro Moschitti, Jim Glass, and Bilal Randeree. 2015. SemEval-2015 task 3: Answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 269–281.

Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, Abed Alhakim Freihat, Jim Glass, and Bilal Randeree. 2016. SemEval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 525–545.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759.

Gil Press. 2019. AI stats news: 86% of consumers prefer humans to chatbots.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017*.

Michelle A. Shell and Ryan W. Buell. 2019. Why anxious customers prefer human customer service. *Harvard Business Review*. Section: Customer service.

Josef Steinberger and Karel Ježek. 2004. Using latent semantic analysis in text summarization and summary evaluation. *Proceedings of the 2004 International Conference on Information System Implementation and Modeling*.

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2008. Learning to rank answers on large online QA collections. In *Proceedings of ACL-08: HLT*, pages 719–727.

Kyle Swanson, Lili Yu, Christopher Fox, Jeremy Wohlwend, and Tao Lei. 2019. Building a production model for retrieval-based chatbots. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 32–41.

Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. 2019. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 100–110.

Ryuichi Takanobu, Qi Zhu, Jinchao Li, Baolin Peng, Jianfeng Gao, and Minlie Huang. 2020. Is your goal-oriented dialog model performing really well? empirical analysis of system-wise evaluation. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 297–310.

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5878–5882.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449.

Lilian Weng. 2020. How to build an open-domain question answering system? *lilianweng.github.io/lil-log*.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77.

Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning semantic textual similarity from conversations. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 164–174.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 1–10.