Towards a Toolbox For Mining QA-pairs and QAT-triplets From Conversational Data of Public Chats

Alexander Egorov, Dmitriy Alexandrov, Nikolay Butakov
ITMO University
St. Petersburg, Russia
al.g.egorov@gmail.com, mr.alexdmitriy@mail.ru, alipoov.nb@gmail.com

***Abstract*—Communities of various specialists are using public groups on platforms like Telegram or Slack to discuss specific domain-oriented topics, for instance, Python programming language, Clickhouse database management system or even film making peculiarities. Conversations in such chats often have a form of questions and answers: someone is looking for information, and someone is giving answers. Both sides working to create a community-driven knowledge source. In a group chat, several parallel discussions on different topics can be held simultaneously, which leads to mixing messages up between dialogues and makes it difficult to get individual dialogues from the chat. In this paper, we consider the problem of data preprocessing for the automatic formation of QA-pairs and QAT-triplets from dialogues of group QA-chats that may be used to exploit information and knowledge stored in conversations. Therefore, to deal with this problem we formulate a set of related tasks and consider approaches to solve them. In particular, we highlight two of the most important tasks: identification of the start of new discussions and classification of mixed-up messages to dialogues they belong to. We perform a comparative study of proposed methods on three large groups from Telegram messenger representing user communities that have variations in communication style and patterns, moderation aspects and topics. The experiments allow us to highlight the influence of these variations on the performance of the proposed methods for the tasks and find the best alternatives among them.**

I. Introduction

Modern messengers such as Telegram provide group chats functionality (public or private), where every participant can ask questions or post some informational messages. These group chats are in extensive use by various communities and company employees for specific topics discussion, question posting, information seeking, and news sharing. For instance, there are open communities of Python developers (@ru_python) and Clickhouse users (@clickhouse_ru), which are represented by public group chats. Discussions in these public chats often consist of mixed-up questions and answers: someone is looking for information, and someone is giving answers in the form of conversation. Such conversations may contain extensive explanations and clarifications before a specific answer is laid out and thus can form a significant body of text in addition to documentation and blogs references which all may be used in training of QA-related models.

Considering the chats we can conclude that they serve as a medium to obtain knowledge, as it is a rich source of information, though without a clear structure. Thus, bringing these chats in a structured form or in the form suitable to be used for building a training dataset for question answering (QA) machine learning methods may lead to several benefits. For example, more efficient accumulation of knowledge and information that later may be represented with an alternative and more efficient way of communication. Users might be able to address their questions to QA-like or bot-like systems and receive the answers instantly. This may make help newcomers increase their proficiency in specific topics by interacting with the system.

Data extracted from the chats in the form of question-answer pairs (QA-pairs) and question-answer-text (QAT-triplets) can be used for extractive and generative question answering over text models (QAT-models [1], [2], [3]), ranking methods for matching between questions and answers [4], structured KBQA [5] with additional processing being applied. Though, additional processing may be required such as matching of QA-pairs with relevant text from official documentation and blogs using text-based ranking algorithms. Apart from that, data from the chats may be used for training other methods such as guiding the answering process with clarification questions, etc.

However, to extract these QA-pairs and QAT-triplets one needs to overcome challenges imposed by the nature of these chats. There are no explicit markings for the start of a new dialogue and multiple dialogues are often mixed-up due to the situations when users lead several conversations at the same time and discuss different questions for a long period, which makes messages sorting harder. The author of the question does not always indicate whether the provided answer helped him or not. Conversations can have a complex structure with multiple clarification questions from both sides, long explanations which may be followed by a summary with the precise answer and may also contain off-topic parts - jokes, non-thematic messages, etc.

We performed an analytical study and identified a set of tasks related to the abovementioned challenges that should be solved to preprocess data and prepare it for the mining. In this paper, we propose a set of methods to cover the identified tasks. These methods combine text features as well as metadata features to achieve better performance and accuracy.

Experimental and comparative studies performed on real public chats from Telegram messenger show that the proposed methods are effective and outperform the baselines.

Our main contributions can be summarized in three main points:

- analytical study of dialogue datasets for Russian speaking communities in Telegram chats and formulations of the tasks that need to be solved for turning data into QA-pairs and QAT-triplets;
- a set of methods for solving part of these tasks that may serve as a foundation for the toolbox;
- comparative study of proposed methods.

The rest of the work is organized as follows. In Section 2, we highlight and summarize the related works. In Section 3, we describe the collected datasets and provide results of the analytical study, formulate the tasks to process the data from chats into QA-pairs and QAT-triplets. In Section 4, we propose a set of methods to solve part of these tasks. Section 5 contains an experimental study of the proposed methods and their comparison with other alternatives. We make conclusions and propose future work directions in Section 6. The source code is available here [6].

II. Related works

Recently, there are many studies in the field of analysis and extraction of knowledge from sources of information exchange and experience sharing. Such sources include community QA (CQA) forums - for example, Stack Overflow, discussion forums that have a less formal format and do not prohibit deviation from the original topic of the thread (Slack), as well as group chats in messengers such as Telegram and WhatsApp.

A. Conversation Disentanglement

One of the main tasks in chats, since a group of people communicates in parallel on different topics, is to unravel the dialogues. Studies [7] and [8] use similar algorithms for determining the dialogue: the algorithm iteratively passes through the messages, assigning one of the existing clusters to the message or forming a new cluster. More recent studies [9] [10] use the BERT-model to encode dialogue messages, followed by the use of classification tokens [CLS] by the transformer in [9] and bi-LSTM in [10]– by the model to determine whether the last message belongs to a given message branch. A more advanced approach [11] uses the Topic-BERT model (a BERT model trained on the Same Topic Prediction-STP problem). The output of such a model determines the answer, topic, and dialogue number. It is important to note that the BERT baseline approach for splitting messages into dialogues verifies whether one pair of messages is an extension of the other.

B. Extracting QA-pair and QAT-triplets

After breaking down the conversation into clear dialogues, further analysis can be carried out. In this type of dialogue, it is useful to extract question-answer pairs or question-answer-confirmation context triplets. The authors of [12] learn separate SVM classifiers to distinguish questions and answers. They use the following features: Question Mark, 5W1H Words, N-grams, Stop Words, etc. In [13], the authors propose a framework for allocating QA-context triplets (context means clarification to the question) based on Conditional Random Fields (CRFs). They use an SVM classifier for extracting a question, the CRF for the question context, and answer extraction (worked better than SVM and Decision Trees). This framework was tested on the TripAdvisor forum.

In addition to the above-described supervised approaches for extracting QA pairs, one of the areas of research is to create semi-supervised and unsupervised methods. Therefore, in [14], the authors utilize LDA to extract topics, the cosine similarity metric to extract candidate questions and answers, and then a heuristic to select among candidates. In [15], the Co-training technique is used, which implies the usage of two classifiers using different features and applying them on unmarked examples, followed by the addition of the most confident examples to the labelled data. Unsupervised solution [16] uses an ensemble of translation model and Language Model.

It is important to note that the task of extracting QA-pairs from the dialogues is validated on the forum data, where there is a precise specificity in the topic of discussion. For effective usage of these methods in chats, a procedure for untangling dialogues is required.

C. Conclusions

First of all, in the existing works individual problems are investigated, whereas, in our work, we propose a set of methods for working with chats to solve several problems at once. Secondly, unlike other works, we analyze and perform our experiments on public chat groups which have variations in communication style and patterns, moderation aspects and topics being discussed in these communities. We aim to highlight how it may influence the tasks being solved. Thirdly, most studies use English-language chats and forums as data, while our work uses Russian-language chats from three different domains at once.

III. Data analysis

A. Datasets

We have chosen Telegram as a data source because it is one of the most popular messengers in Russia. To solve the problem of extracting and forming QA-pairs/QAT-triplets, messages from the following chats were crawled:

- @clickhouse_ru - chat is dedicated to discussing the Clickhouse DBMS;
- @ru_python - chat to discuss the usage of Python programming language;
- @kinota1k - chat for film production workers.

These are fairly large and well-formed communities that exist for several years already. During this time, they developed their own rules of conduct, accumulated a significant number of messages to be sufficient for analysis. All three communities have moderation: ads and completely irrelevant messages are removed. The chat dedicated to the DBMS Clickhouse is official and moderated by the DBMS's developers themselves. This is probably why there are fewer informal conversations. The other two communities are supported by enthusiasts and the nature of communication there is a little less formal. These communities are not entertainment in nature, but are

mainly intended to help solve problems on specific topics (indicated in the community name) - information from such chats can be used to build chatbots, so these chats were chosen for analysis as typical representatives of question-answer chats. The number of messages, as well as the time range of messages, are shown in Table I. Several examples of messages are shown in Table II.

For the analytical study, from each dataset, a sample of approximately 1500 messages were taken, see table I. Each dataset was labelled manually by three assessors. Messages, where the assessors' opinions were not consistent, were considered collectively to come for a label agreement. It has been observed that people tend to write several short messages instead of one long message, where each next message continues a thought from the previous one. Therefore, it was decided to combine pairs of messages from one author, if the interval between them is not more than 30 seconds and they do not have explicit reply-links to the other messages. Since several dialogues can be conducted simultaneously in the chat, the messages are mixed. Therefore, the first thing that was done - all messages indicating the beginning of a new dialogue were found. After that, all messages were grouped by a dialogue. Some dialogues in the selections got there without initial messages - such dialogues were not suitable for analysis and were filtered out. Not all of the remaining dialogues contained questions (especially in the film-making chat), many began with announcements, news quotes, jokes, etc. These messages are not useful for the task of QA-pairs / QAT-triplets dataset creation, so the assessors were asked to mark all initial messages as interrogative and affirmative. After the markup, the affirmative dialogues were filtered out. The remaining dialogues are potentially suitable for composing QA pairs / QAT triplets. Assessors were asked to highlight messages in which, in their opinion, answers to questions are contained. Also, the assessors were asked to highlight messages in which the authors of the questions thank the respondents or, on the contrary, write that the advice did not help. Potentially, such messages are easily searched for in the text and from them, you can judge whether the previous message is a useful or useless answer.

Many dialogues are very long. This happens mainly because, during the communication, the topic of the dialogue can change and most often it is done by the dialogue initiator. Assessors were asked to highlight messages in which the topic of the dialogue has changed. If the topic changes smoothly with no explicit message, the mark was placed in the conditional middle between the two dialogues. Additionally, the assessors marked the flood in the dialogues.

Finally, assessors marked: - initial messages in dialogues; - dialogue numbers; - was the initial message in the dialogue affirmative or interrogative; - the moment of changing the topic of the dialogue; - messages in which the author thanked for the answers; - messages in which the author wrote that the answer did not help / the problem was not solved; - non-informative messages.

B. The tasks for QA-pairs and QAT-triplets mining

After the analysis of the data, we formulated the several tasks that need to be solved to make unstructured public chats useful for QA by extracting and forming QA-pairs/QAT-triplets:

1) search for initial messages to find individual dialogues;
2) classification of messages among dialogues and thus reconstructing of the dialogues themselves;
3) highlighting messages containing answers to the initial questions. To simplify the task of selecting answers, we have added several auxiliary subtasks:
   - filtering non-informative messages: jokes, long discussions on the side topics (flood), etc;
   - splitting the dialogue into QA-sessions. During the conversation, the topic of the discussion may change. It is necessary to highlight messages within one discussion, after which the topic of the discussion changes. If the topic changes, the small probability that there are answers on the initial message;
   - search for messages in which the author of the question thanks for the answers. Such messages could help to extract the best answers;
   - search for messages in which the author says that the proposed solution did not help - the problem remained. If we find such messages we can define, that all solutions before such messages do not have good answers.

IV. Toolbox: the set of methods

In paragraph III-B the list of tasks that could help to solve the problem of extracting and forming QA-pairs/QAT-triplets was formulated. Below the solution of each task will be described.

A. Search for initial messages of dialogues

Each message contains the author, time of sending, sequence number, reply-link if any, and text information of the message. Therefore, the features that the classifier and heuristics may use can be characterized as a) metadata features; b) text features; c) combined features. Approaches to classification can be: a) purely heuristic, b) classifiers on metadata, c) classifiers on textual features, d) classifiers on all features. Further, they will be described in this order.

Two simple heuristics are used as baselines.

- Baseline 1: if the message contains a question mark, we classify it as a positive class.
- Baseline 2: if the message contains a question mark and the author was not active for the last hour, we classify it as the positive class.

Further, only one classifier was checked on the metadata:

- XGBoost on the metadata features: was author active during the last 24 hours, was author active during the last hour, was author active during the last minute, is the author new, was the previous message written by him.

The following group of classifiers uses only text message features:

- MLP classifier on the vectors of texts obtained from the pretrained BERT;

TABLE I. DATASETS AND SIZES

Name	Range	Size	Sample size	Size after combining messages	Number of dialogues	Number of interrogative dialogues	Number of uninformative messages ("flood")
@clickhouse_ru	2016-2020	152103	1437	1229	118	114	0
@ru_python	2015-2019	984361	1493	1302	89	84	469
@kinota1k	2017-2021	65888	1510	1351	236	207	584

TABLE II. EXAMPLE OF MESSAGES (TRANSLATED FROM RUSSIAN)

clickhouse_ru	ru_python	kinota1k
User1, [19.05.20 19:42] Good day to all. People, tell me, how do you parse user_agent?) We collect access logs of nginx from balancers in clickhouse, ua in a row. I wanted my own analytics on the browsers used - we do not know how it would be more correct to process the field	**User4, [23.10.19 21:35]** Hi! I use Postgres and in one of the fields I store a list of dictionaries in the format "[..., ...]". We need to add dictionaries to this list (let's call the stat column) in some optimal way. With jsonb, the request for some reason can not be built. Can you help me?	**User9, [18.12.20 17:31]** Hello everyone)! Guys, please tell me, do photographers work on the sites? If so, what do they do? I apologize for the stupid question, but it's been bubbling inside for a long time)

User2, [19.05.20 19:47] [In reply to User1]
We use DBA at one time just made a reference book and when sucking logs, we changed the found UA to its ID. But this is the SQL-way. In the Clickhouse type, there is no full-fledged FullText index, but many have already implemented something similar. Here you can try to search.
	User5, [23.10.19 21:38] [In reply to User6]
	It is better to do such way:
	for i, el in enumerate(reversed(lst)):
	if el is 1:
	return len(lst) - i
		User10, [18.12.20 17:32]
		they photograph the shooting process, the actors

User3, [19.05.20 19:52] [In reply to User1] There are many different libraries on the same github for parsing the user agent.)	**User7, [23.10.19 21:45]** If I have One to many connection in alchemy, Can I do this: parent.child.append (new_recording)?	**User11, [18.12.20 17:34] [In reply to User9]** And they take pictures of the scenery for the art workshop
User1, [19.05.20 19:55] [In reply to User2] An interesting solution. But the dictionary will be very huge...	**User8, [23.10.19 21:46] [In reply to User4]** UPDATE tablename SET jsonb_column = jsonb_column ||'...'::jsonb	**User9, [18.12.20 17:34]** That is, the photographer is also always on the site?

- MLP classifier on the vectors of texts obtained from pretrained doc2vec.

Next, we try to get the additional information from the message texts by adding binary features obtained from the texts: is there a greeting in the text, whether the author wrote about errors or exceptions in code programs, whether the author asked for help, is the question mark in a text, whether the text is empty, is there a link in the text, is there a program code in the text. These features are obtained by checking keywords in the text. We compared several classifiers:

- Logistic regression
- SVM
- XGBoost

The last group of classifiers uses all the available features:

- XGBoost trained on features obtained from messages' metadata, binary features obtained from texts and embedding vectors from BERT;
- MLP trained on features obtained from message metadata, binary features obtained from texts, and embedding vectors from BERT;
- Stacking MLP and XGBoost: MLP classifier trained on texts, after that XGBoost get the predict of MLP, features from messages' metadata and binary features obtained from the text.

The feature sets and related to the groups of methods are presented in Fig.1.

B. Classification of messages by dialogues

When the initial dialogue messages are highlighted, the next task is to assign each chat message to the corresponding individual dialogue.

We use a simple baseline which uses the ordinal number of messages:

1) Find all the initial messages of new dialogues in the dataset.

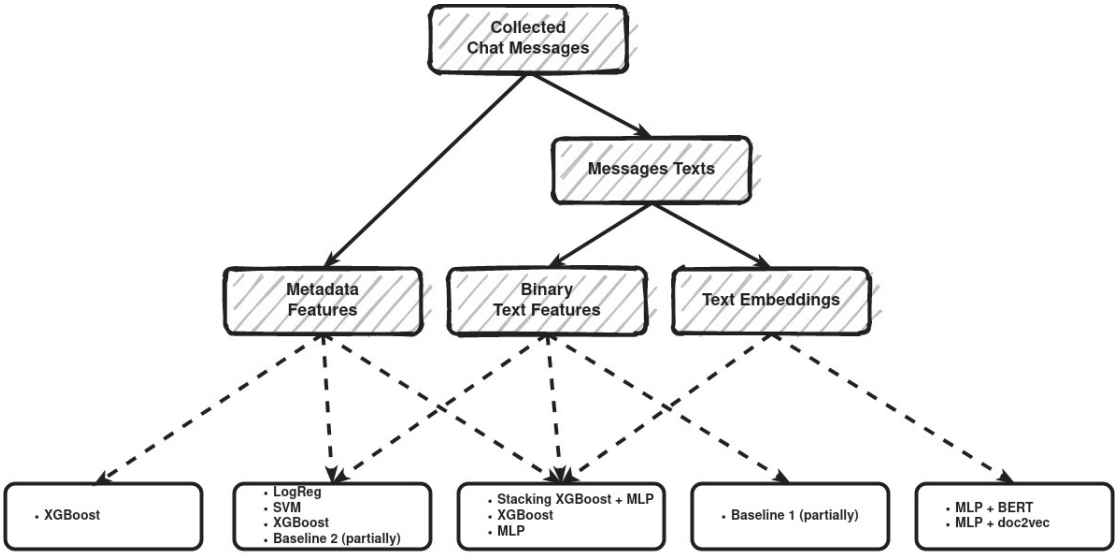

Fig. 1. Feature sets and their serving in different ML methods for the task of the search for initial messages of dialogues

2) Going through all the messages sequentially in the order of appearance. If the i-th message has a reply-link to the j-th message, then the i-th message is assigned the same dialogue number as the j-th message. Otherwise, the i-th message is assigned the same dialogue number as the i-1 message.

The disadvantage of such a classifier is that it doesn't use textual information and works on the level of individual messages. Explicit reply-links in messages play an important role in classifying messages into dialogues since such messages do not need to be classified and get additional information. We use reply-links to be able to capture as much context of the message as possible. Specifically, we combine messages into reply-chains by reply-links and further we consider not individual messages, but reply-chains. We proposed the following recovery algorithm:

1) separate the dataset into reply-chains. Order the chains by the first message;
2) take all the possible pairs of reply-chains;
3) for each such pair of chains, a binary classification problem is solved: if two reply-chains are from one dialogue we considered it as a positive class, else - as negative. Along with the classification labels we take labels probabilities from classifiers;
4) for each reply-chain we find a pair with the maximum probability and consider that these two reply-chains are from one dialogue;
5) the messages with initial questions have an ordinal number of the corresponding dialogue. Messages which have reply-links on these messages are in the same dialogues. All messages from one "positive" pair which was obtained on the previous step are in the same dialogue. If after such labelling some messages are left without the dialogue label, the classification is done with our baseline heuristic.

For each message we have a true dialogue number and predicted dialogue number, it is not a binary classification, so we use an accuracy metric. As a classification, we use stacking of XGBoost and BERT. For working with text data (i.e., with the semantic component), the BERT model was chosen, since it has proven itself well in the tasks of text classification. We use [EOT] token to combine all texts from reply-chain to one text and [SEP] to combine texts of two reply-chains into one text. As a pretrained model we use bert-base-multilingual-uncased. XGBoost accepts the following attributes as input: the output probability of the BERT model; the number of messages between the reply-chains; the minimal time between the reply-chains; the number of messages in the reply-chains; the number of authors in each message chain; the number of common authors in both chains; the time between the first message and the last message in each chain (Fig. 2).

C. Search for answers

To find answers in dialogues, it's necessary to solve several auxiliary tasks:

- splitting the dialogue into QA-sessions. During the conversation, the topic of the discussion may change. It is necessary to highlight messages within one discussion, after which the topic of the discussion changes. If the topic changes, the small probability that there are answers on the initial message;
- filtering non-informative messages: jokes, long discussions on the wrong topic (flood), etc.
- search for messages in which the author of the question thanks for the answers. Such messages could help to extract the best answers;
- search for messages in which the author says that the proposed solution did not help - the problem remained. If we find such messages we can define, that all solutions before such messages do not have good answers.

The last two tasks can be solved by searching for keywords in the message texts. The other two tasks require more detailed analysis and are not considered in this paper. The problem of finding the moment when the dialogue changes can probably be solved by using a thematic model: passing the window

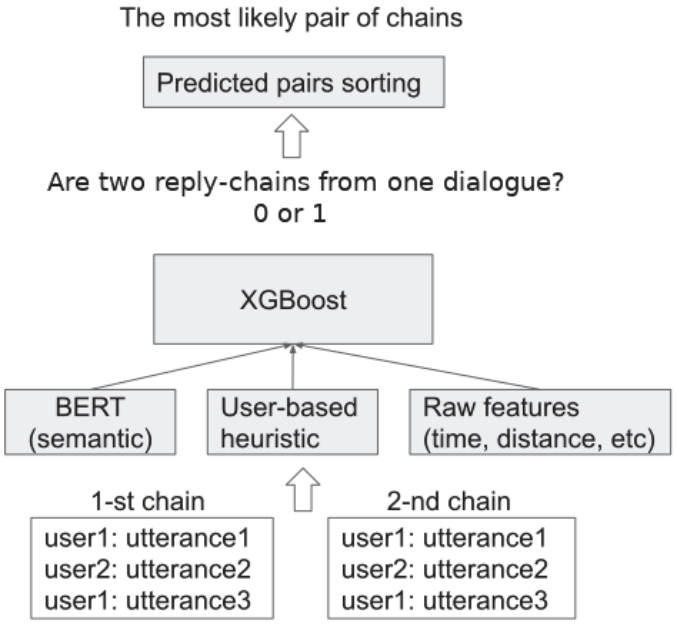

Fig. 2. Diagram of the method for restoring dialogues number

through the sequence of messages and fixing the moment when the dialogue changes. Filtering non-informative messages can probably be solved by binary classification of text messages. In this study, we will limit ourselves to checking how much the quality of finding answers can be improved by manually marking up uninformative messages, as well as highlighting the moments of dialogue changes.

Algorithm for solving the answer selection problem:

- search for the initial messages with questions;
- classification of messages by dialogues;
- separation of messages in QA-sessions;
- deleting all messages from the dialogues after the first change of the QA-session;
- filtering non-informative messages;
- search for messages in which the author says that the proposed solution did not help;
- delete co-promises and from questions to messages that the problem is not solved;
- search for messages in which the author of the question thanks for the answers;
- messages preceding messages with thanksgivings are replies;
- if there are no messages with gratitude, you can use all the affirmative messages not from the author of the question as answers.

V. Experimental study

A. Comparative study

To test the work of the proposed methods, we used three datasets described in Section III-A.

For the problem of initial messages detecting, the quality was measured by cross-validation with the leave-one-out strategy: each message was once removed from the dataset, the classifier was trained on the remaining part and the prediction for the removed message was made. After that, precision, recall and F1-score for all messages were calculated. The results are shown in Table III. For the problem of messages classification by dialogue for each message, we predicted the number of dialogue it belonged to and measured the results with an accuracy metric. Classification results can be found in Table IV. Results of searching for the messages where the question author shows gratitude for the provided solution are in Table V.

B. Discussion

The results on Clickhouse chat data show the high quality of the advanced base heuristic (Baseline 2), which is an important indicator of features such as author activity and some text aspects. The classifier with metadata features only was not able to obtain scores higher, than those that are provided by heuristic baseline (Baseline 2). Considering the text features, even the BERT-based MLP solution was not able to outperform this baseline as well. Though BERT-based MLP performs notably better than the alternative option based on doc2vec. The combination of binary text features and metadata provides a substantial improvement over the baseline - 14.1% for the XGBoost model. The results show that both metadata and text are essential for solving the task while embeddings based features do not introduce new essential information to the binary+metadata schema.

The text features may not work as expected due to several reasons. Firstly, there are cases when the message, which starts a new dialogue, is very close to or almost indistinguishable from the follow-up or clarification question. Secondly, there may be too many domain-specific words or words with domain shifted meaning (for instance, MergeTree, materialized view, table engine) that affects the performance of the pre-trained

TABLE III. COMPARATIVE STUDY OF METHODS FOR INITIAL MESSAGE EXTRACTION TASK

		clickhouse_ru			ru_python			kinota1k		
№	Name	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
1	Baseline 1	0.47	0.80	0.59	0.39	0.60	0.48	0.76	0.51	0.61
2	Baseline 2	0.86	0.71	0.78	0.75	0.43	0.55	0.85	0.40	0.55
	features from message metadata									
3	XGBoost	0.72	0.65	0.68	0.68	0.51	0.58	0.60	0.80	0.69
	features from message metadata and binary features from texts									
4	Logistic regression	0.87	0.84	0.86	0.76	0.60	0.68	0.76	0.64	0.69
5	SVM	0.87	0.84	0.86	0.78	0.60	0.68	0.83	0.67	0.74
6	XGBoost	0.87	0.90	0.89	0.80	0.58	0.67	0.83	0.68	0.75
	only texts									
7	MLP on embeddings from pretrained doc2vec	0.61	0.31	0.41	0.66	0.36	0.47	0.78	0.34	0.48
8	MLP on embeddings from pretrained BERT	0.48	0.86	0.62	0.52	0.78	0.62	0.65	0.95	0.77
	all features									
9	XGBoost	0.88	0.87	0.88	0.80	0.62	0.70	0.80	0.79	0.79
10	MLP	0.66	0.94	0.77	0.66	0.80	0.72	0.65	0.95	0.78
11	Stacking XGBoost and MLP	0.90	0.89	0.90	0.78	0.67	0.72	0.79	0.82	0.80

TABLE IV. ACCURACY METRIC FOR CLASSIFICATION OF MESSAGES AMONG DIALOGUES

№	Name	clickhouse_ru	ru_python	kinota1k
1	Baseline heuristic	0.87	0.78	0.92
2	XGBoost and BERT	0.60	0.36	0.75
3	XGBoost and BERT + heuristic	0.89	0.76	0.98

TABLE V. SEARCH FOR MESSAGES IN WHICH THE AUTHOR OF THE QUESTION THANKS

	clickhouse_ru			ru_python			kinota1k		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
With words lemmatization	0.84	0.87	0.85	0.80	0.62	0.70	0.83	0.94	0.88
Without words lemmatization	0.93	0.87	0.90	0.94	0.62	0.74	0.91	0.94	0.92

embeddings while at the same time there is only a fraction of labelled data available.

However, as conversations become (1) less formal and (2) the usage of domain-specific professional words decreases - the exploitation of raw texts leads to better results comparing to metadata + binary schema. This effect can be seen in Table III for ru_python and kinota1k groups. The methods operating with all features have improvements up to 5.8% and 6.6% respectively. Also, we can see a notable drop in quality of Baseline 2 related to the above mentioned factors while classifiers can adapt in variations among groups and yield better results.

Pretraining on a massive corpus of domain-specific text with high usage of professional words like official documentation, topic-related blogs and the chat data itself may yield a significant improvement to the results.

We tested the proposed baseline and classifier solutions for the task of message resorting among dialogues and presented results in Table IV. One can see from the table that the baseline approach works surprisingly good especially in comparison with the classifier-based solution. The groups have different levels of dialogues mixing during conversations reflected by the accuracy of the baseline. Though the classifier uses more advanced features, it has a drawback: a significant part of the reply-chains may be unlabeled. This is due to the difficulty of translating binary labels on pairs of reply-chains into dialogue numbers. We use an approach with the construction of all possible pairs of reply-chains, their classification, and then for each reply-chain, we select only one candidate with the maximum probability. This disadvantage can be overcome if chains are further marked using heuristics. The more messages could be labelled with the classifier, the better the final metrics will be: on kinota1k this approach significantly surpassed the baseline heuristic (6.5%), on clickhouse_ru - insignificantly (2.6%), on ru_python - this approach works worse (2.6%), apparently because that the classifier has labelled only a small part of the reply-chains. It should be noted that when labelling dialogues, messages that have explicit reply-links are not labelled. They are assigned the same labels as the messages they reply to. This leads to the fact that if one message in the reply-chain is classified incorrectly, and the chain is long, then all messages in the chain that follow this message are classified incorrectly and this greatly affects the resulting metric. Note also that comparison of texts suffers from the short length and often too general content of texts featured by follow-up and clarification questions that can be easily positively matched with multiple alternatives.

The proposed heuristic for finding messages where the author thanks for the answer gives the F1 metric from 0.75 to 0.9, depending on the dataset. There was an assumption that the quality of the search may decrease due to different endings

of words, so we tried two types of search: with lemmatization and without. The results show that metrics are worse with lemmatization because more false-positive cases appear.

There were only a few examples of the messages (less than 15 in each dataset) in which the author says that the problem has not been resolved. This does not allow us to build a full-fledged classifier, but we can estimate the number of answers by filtering the extra messages in our samples. In this study, we did not touch upon the problem of detecting the moments of subject change in the dialogues, as well as the problem of detecting non-informative messages, limiting ourselves only to analytical research. An additional classification of dialogues into interrogative and affirmative is not needed, since the assessors considered that all affirmative dialogues consist of non-informative messages. These are two major challenges. The first can be solved with the help of a topic model, which will walk through the sequence of messages with a window and detect the moment of a change in the subject, the second can be solved by a text classifier.

After filtering all non-informative messages and messages after the topic change in the discussion, we consider all affirmative messages posted not by the author of the original message as answers. These messages can contain explanations, partial answers, suggestions and full answers. These materials can be used to form QA-pairs and QAT-triplets and are subject to further processing.

VI. Conclusion

In this work, we proposed a set of methods for solving tasks that are required to eventually perform mining of QA-pairs / QAT-triplets. Among these tasks - searching for initial messages of dialogues, classification of messages by dialogues to which these messages belong to, extracting answers and etc. The conducted experimental study allowed comparing different alternatives to these tasks and highlight the most efficient and promising methods. For instance, the best-proposed methods for the task of the initial messages search outperform the baseline up to 14.1%. Apart from that, we compared the performance of methods on different groups which have variations in communication style and patterns, moderation aspects and topics being discussed in the community. The latter shows that these variations influence the performance of classifiers and the usefulness of different features. One needs to take into account these peculiarities to achieve better results.

In future work, we plan to extend our research to cover the rest of the tasks concerned in this work. In particular, we will add classifiers for detecting non-informative messages and a classifier for detecting the moments of change in the discussion topic. Domain-specific pretraining (using chat texts itself along with official documentation and community-driven blogs) should be considered as a way to improve the raw text contributions to the overall performance.

Acknowledgement

This research is financially supported by The Russian Science Foundation, Agreement #20-11-20270.

References

[1] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016.

[2] C. Qu, L. Yang, M. Qiu, W. B. Croft, Y. Zhang, and M. Iyyer, "Bert with history answer embedding for conversational question answering," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1133–1136, 2019.

[3] A. Hoang, A. Bosselut, A. Celikyilmaz, and Y. Choi, "Efficient adaptation of pretrained transformers for abstractive summarization," *arXiv preprint arXiv:1906.00138*, 2019.

[4] Y. Xie, W. Yang, L. Tan, K. Xiong, N. J. Yuan, B. Huai, M. Li, and J. Lin, "Distant supervision for multi-stage fine-tuning in retrieval-based question answering," in *Proceedings of The Web Conference 2020*, pp. 2934–2940, 2020.

[5] W. Cui, Y. Xiao, H. Wang, Y. Song, S.-w. Hwang, and W. Wang, "Kbqa: learning question answering over qa corpora and knowledge bases," *arXiv preprint arXiv:1903.02419*, 2019.

[6] Alexander Egorov, Dmitriy Alexandrov, Nikolay Butakov, "Toolbox For Mining QA-pairs and QAT-triplets From Conversational Data of Public Chats." https://github.com/AleksanderE/qat_miner, 2021.

[7] D. Shen, Q. Yang, J. T. Sun, and Z. Chen, "Thread detaction in dynamic text message streams," *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, vol. 2006, pp. 35–42, 2006.

[8] L. Wang and D. W. Oard, "Context-based message expansion for disentanglement of interleaved text conversations," *NAACL HLT 2009 - Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, no. June, pp. 200–208, 2009.

[9] H. Zhu, F. Nan, Z. Wang, R. Nallapati, and B. Xiang, "Who did they respond to? Conversation structure modeling using masked hierarchical transformer," *arXiv*, 2019.

[10] T. Li, J. C. Gu, X. Zhu, Q. Liu, Z. H. Ling, Z. Su, and S. Wei, "DialBERT: A hierarchical pre-trained model for conversation disentanglement," *arXiv*, 2020.

[11] W. Wang, S. C. Hoi, and S. Joty, "Response Selection for Multi-Party Conversations with Dynamic Topic Tracking," pp. 6581–6591, 2020.

[12] L. Hong and B. D. Davison, "A classification-based approach to question answering in discussion boards," *Proceedings - 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009*, pp. 171–178, 2009.

[13] S. Ding, G. Cong, and C.-y. L. Xiaoyan, "Extracting Question-Context-Answer Triples from Online Forums," *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 514–523, 2009.

[14] S. Henß, M. Monperrus, and M. Mezini, "Semi-automatically extracting FAQs to improve accessibility of software development knowledge," *Proceedings - International Conference on Software Engineering*, pp. 793–803, 2012.

[15] R. Catherine, R. Gangadharaiah, K. Visweswariah, and D. Raghu, "Semi-Supervised Answer Extraction from Discussion Forums," *Proc. IJCNLP 2013*, no. October, pp. 1–9, 2013.

[16] P. Deepak and K. Visweswariah, "Unsupervised solution post identification from discussion forums," *52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014 - Proceedings of the Conference*, vol. 1, pp. 155–164, 2014.