Email Answering by Matching Question and Context-Specific Text Patterns: Performance and Error Analysis

Eriks Sneiders[1], Jonas Sjöbergh[2], Alyaa Alfalahi[1]

[1] Department of Computer and Systems Sciences, Stockholm University
Postbox 7003, SE-164 07, Kista, Sweden
{eriks,alyalfa}@dsv.su.se
Department of Information Science, Hokkaido University
Sapporo, 060-8628, Japan
js@meme.hokudai.ac.jp

Abstract. Automated answering of frequent email inquiries is a text categorization task with narrow text categories, where all messages in one text category have the same answer. Such email categorization is optimized for high precision and at least acceptable recall. We tested matching of surface text patterns to nearly ten thousand email messages and achieved around 90% precision; the corresponding recall figures were 45-75% in different text categories. In order to achieve this performance level, the text patterns are designed to identify both the context of an email inquiry and the actual need that has created the inquiry – a question, request, or complaint. Our error analysis has pinpointed 12 reasons why text pattern matching may fail.

Keywords: Email answering, email categorization, text-pattern matching.

1 Introduction

Text pattern matching has been used in Information Extraction [1], automated question [2] and email [3] answering. Any text recognition task that involves regular expressions means matching surface text patterns.

In this paper, we explore matching of manually crafted text patterns to e-mail messages in order to assign standard answers, i.e., in order to put a message into a specific text category. In a series of tests we have investigated:

- How good email categorization-by-answer can we achieve by matching manually crafted text patterns that specify a user request and the context of the request? Assuming that such text patterns are of the highest quality, our performance figures may hint at the upper performance limits of text-pattern matching.
- How does the performance of the system change when we iteratively test and then modify the text patterns, i.e., iteratively "re-train" the system?
- When text-pattern matching fails, what is the cause of the failure? We believe the mistakes made by our system are of general nature for email answering by text pattern matching, not specific to the particular text patterns.

The paper is organized as follows. The next section introduces two vital elements of an email inquiry – context description and user request that triggers the response – which together determine the answer. Sections 3 and 4 introduce our experiment data, the email answering techniques, and the test cases. Section 5 discusses the experiment results. Section 6 does error analysis. Section 7 concludes the paper.

2 Context Description and Response Trigger in Email Inquiry

We have been working with email sent to contact centers for quite some time. Through observations of tens of thousands of email messages we have discovered that the initial inquiry consists of two parts – (i) a description of the subject or context of the inquiry and (ii) a response trigger. The response trigger can be a question waiting to be answered, a request to carry out an activity and report the results, or a statement that requires a response (e.g., a complaint). For automated email answering, requests for information and requests to complete a task are of interest – the system can deliver information or redirect the user to a self-service application.

In a *simple question*, explicit or implicit, both the context and the response trigger are placed in one sentence, both contain meaningful keywords. Example: "I wonder whether **I** **get** any **payment** in April because I haven't received any notification yet." "Get" and "payment" are the keywords that define the subject of the inquiry. "Wonder" raises the question to be answered. The second "I" disambiguates the question. "In April" specifies a detail of the inquiry pinpointed by the subject. The entire question is a response trigger.

Sometimes *relevant keywords describe the context while the response trigger does not contain any keywords*. Example: "I sent you an **application** for **parental** **benefits** more than a week ago, how is my case progressing?" "How is my case progressing" is the response trigger which does not contain any subject-specific keywords and acquires meaning only in the presence of a particular context.

Often *the context and the response trigger come in separate sentences*, which makes it more difficult to identify the inquiry. Let us consider the sentence "I **applied** for **housing** **allowance** on January 13." The statement introduces a story, yet alone it does not ask for any response. The person continues the story by telling that she has got half of the amount she expected and would like to know "When will the rest of my money come?" This is the question to be answered, the response trigger that makes sense only together with its context. The context and the response trigger are in separate sentences in a random sequence, possibly having a few other sentences in between. It may happen that the response trigger does not contain any subject-specific keywords even in a separate sentence.

A context description provides keywords that are useful for topic-related text clustering and categorization. In order to answer a message correctly, the system must take the next step and identify the request, the need, why the message was sent in the first place. So far we have found only one reference to a clear separation of context description and response trigger in automated email answering [4]: "Determining the factual content of an e-mail is not sufficient to answer it correctly – its purpose is also very important. […] In the e-mails we are analyzing, the same set of data can be extracted from e-mails that have different purposes. The answer to be generated for these emails must therefore be different.”

Resolving the purpose of an email message is not a new research problem. There exists research that applies speech-act theory in order to categorize workplace email messages according to the purpose of the message, not the topic. The purpose can be a request for action, information, permission, a proposal to meet, a promise or a commitment [5-6]. Requests and commitments may be conditional or unconditional, explicit or implicit [7]. Locus ambiguity may be a challenge: while human annotators of training data tend to agree that the message contains a request or commitment, they may not agree on exactly which utterances contain them [8]. Goldstein and Sabin [9] took a broader look at email tasks and defined 12 email genres according to their task, including also expression of feelings, document forwarding, advertising, spam, etc.

In an email message sent to a contact center, the response trigger is a task-related speech act. It may be explicit or implicit. It may be ambiguous and vague. Context is important in interpreting the request. (See Section 6 for related discussion.)

3 Experiment Data

Our test data is 9663 email messages sent by citizens to the Swedish Social Insurance Agency. We extracted the message body; if there was a dialogue thread, we took the chronologically last message. No meta-data was used. We identified the most common types of inquiries, i.e. text categories, and selected five text categories for automated answering: (Cat1) “Please send me a fill-in-form!”, (Cat2) “When will you decide my housing allowance?”, (Cat3) “How many days of parental benefits do remain for my child?”, (Cat4) “How much will I get in my future pension?”, (Cat5) “When do I get my money?”. These categories had a clear information need and could be answered by a standard answer. We manually labeled each message according to its text category. Preparation of the messages is covered in [10].

The messages came in four batches, called collections A, B, C, and D. Table 1 shows the distribution of the messages across these collections and the text categories. The sum of all messages across categories in one collection is larger than the number of messages in the collection because some messages belong to two categories. In collection A there are 6 messages that belong to two categories, in B – 9 messages, in C – 6 messages, and in D – 8 messages. The size of the messages varies. The minimum, maximum, average, and median number of words per message are 4, 321, 45.5, and 35. The minimum, maximum, average, and median number of sentences are 1, 45, 5.2, and 4.

Table 1. Number of messages by collection and by text category.

	Collection	Cat 1	Cat 2	Cat 3	Cat 4	Cat 5	Rest	Sum
Collection A	2437	94	76	51	29	362	1831	2443
Collection B	1967	76	62	49	30	269	1490	1976
Collection C	2473	109	105	53	22	387	1803	2479
Collection D	2786	148	79	45	78	393	2051	2794
Total	9663	427	322	198	159	1411	7175	9692

Table 2. Format of context description and response trigger, and the number of corresponding inquiries across a sample of email messages.

Format of the inquiry	Num	Share %
Simple question (explicit or implicit)	159	52
Simple question; response trigger without representative keywords	8	3
Simple question; vaguely written message	3	1
Context description and response trigger come in separate sentences	87	28
Context description and response trigger come in separate sentences; response trigger without representative keywords	33	11
Context description and response trigger come in separate sentences; vaguely written message	14	5
Total	304	100

In order to discover the format of the context descriptions and response triggers in our data, we examined 300 random messages in Cat1 through Cat5 in collection C. Four messages belong to two text categories, therefore we have 304 message-category pairs and the corresponding number of inquiries (Table 2).

4 Text Matching Techniques and the Test Cases

Our text-pattern matching system operates 154 manually crafted text patterns. The syntax of the text patterns resembles regular expressions. In a text pattern, we can define a number of synonyms that designate a concept; we can define the order of the words and the distance between the words. The system has spelling correction, word stems, and compound splitting built into the pattern matching process. Each text pattern is designed to match a context description and a response trigger. The text patterns are optimized for precision – if the message is answered, it must be answered correctly. Closely following precision comes recall. A complete description of the text-pattern matching technique is available in [11]; relevant statistical characteristics and lexical classes of matching text patterns are discussed in [12]. This paper complements the previous publications, it discusses the accuracy of email answering and the errors.

Table 3. Evaluation cases with “training” and test data sets.

Test name	“Training” collections	Test collection
A-B	A	B
AB-C	A+B	C
ABC-D	A+B+C	D
AB-D	A+B	D

Initially we had only collection A for “training” and collection B for testing. We write “training” in quotes because manual development of text patterns is not training as in machine learning. Over some period of time collections C and D arrived, which made it possible to measure stepwise improvement of the text patterns: we could add the test collection to the "training" data and take a new test collection. We wanted to find out how much "retraining" of the text patterns with new messages could improve the system's performance.

Different test collections, however, do not allow seeing the true effect of stepwise improvement. We fixed this in the tests AB-D and ABC-D. Unfortunately, we could not use D in order to test patterns "trained" solely on A because by the time D arrived the old A-trained text patterns were already lost.

In order to compare the pattern-matching system with a standard machine learning method we used the WEKA framework. In the previous tests with Support Vector Machine (SVM) and Naïve Bayes, and the collections A+B, SVM slightly outperformed Naïve Bayes [10]. A survey of email classification tasks [13] has also concluded that SVM and Naïve Bayes yield similar performance.

We chose SVM as the baseline. The only machine learning features were individual terms. The text pre-processing were spelling correction, lemmatization, and compound splitting. We tested them in various combinations and reached text categorization accuracy between 0.860 and 0.869; it was ten-fold cross validation on data collections A+B+C+D. We did not use advanced features such as semantic distance between terms because tools such as WordNet do not exist for Swedish. The SVM parameters were set to the WEKA default parameters. We used vectors of dimension 10 000; smaller vectors gave worse classification accuracy and longer vectors made the computer run out of memory.

SVM is a proven technique for topic-related text categorization. In our tests, identifying the response trigger is likely to be a challenge, especially because 20% of the messages are vaguely written or the response trigger does not contain representative keywords (see Table 2).

5 Experiment Results

We measured correctness of email categorization by applying the very traditional in Information Retrieval precision, recall, and F-score. The values in Table 4 and Table 5 are arranged to emphasize the trends through the consecutive tests. Figure 1 illustrates these trends.

The total precision for text-pattern matching was calculated as the number of correctly placed messages divided by the sum of correctly and incorrectly placed messages across all the categories in the test collection. The total recall was calculated as the number of correctly placed messages divided by the number of message-category pairs across all the categories in the test collection. The number of messages is slightly lower than the number of message-category pairs because a few messages belong to two categories. For SVM, "Avg" is weighted average; WEKA calculated the average value across all the text categories considering the number of messages in each category.

Table 4. Precision, recall, and F-score for the tests A-B, AB-C, and ABC-D.

	Precision			Recall			F-score		
Test	A-B	AB-C	ABC-D	A-B	AB-C	ABC-D	A-B	AB-C	ABC-D
	Text-pattern matching								
Cat1	0.91	0.92	0.93	0.54	0.63	0.59	0.68	0.75	0.72
Cat2	0.97	1.00	0.91	0.53	0.56	0.65	0.69	0.72	0.76
Cat3	0.96	0.92	0.92	0.55	0.85	0.76	0.70	0.88	0.83
Cat4	0.88	1.00	0.89	0.50	0.45	0.44	0.63	0.63	0.59
Cat5	0.84	0.91	0.88	0.41	0.63	0.59	0.55	0.75	0.71
Rest	0.86	0.88	0.87	0.99	0.99	0.98	0.92	0.93	0.92
Total	0.86	0.89	0.88	0.86	0.89	0.87	0.86	0.89	0.88
	Support Vector Machine								
Cat1	0.60	0.69	0.69	0.49	0.54	0.69	0.54	0.61	0.69
Cat2	0.73	0.85	0.69	0.60	0.77	0.86	0.66	0.81	0.76
Cat3	0.89	0.82	0.73	0.82	0.85	0.89	0.85	0.83	0.80
Cat4	0.77	0.85	0.70	0.43	0.50	0.58	0.55	0.63	0.63
Cat5	0.67	0.73	0.63	0.65	0.71	0.80	0.66	0.72	0.70
Rest	0.89	0.90	0.93	0.92	0.92	0.87	0.90	0.91	0.90
Avg	0.84	0.86	0.85	0.85	0.86	0.84	0.84	0.86	0.85

Table 5. Precision, recall, and F-score for the tests AB-D and ABC-D.

	Precision		Recall		F-score	
Test	AB-D	ABC-D	AB-D	ABC-D	AB-D	ABC-D
	Text-pattern matching					
Cat1	0.89	0.93	0.55	0.59	0.68	0.72
Cat2	0.92	0.91	0.62	0.65	0.74	0.76
Cat3	0.90	0.92	0.82	0.76	0.86	0.83
Cat4	0.91	0.89	0.40	0.44	0.55	0.59
Cat5	0.84	0.88	0.55	0.59	0.67	0.71
Rest	0.86	0.87	0.97	0.98	0.91	0.92
Total	0.86	0.88	0.86	0.87	0.86	0.88
	Support Vector Machine					
Cat1	0.75	0.69	0.55	0.69	0.64	0.69
Cat2	0.81	0.69	0.77	0.86	0.79	0.76
Cat3	0.76	0.73	0.76	0.89	0.76	0.80
Cat4	0.78	0.70	0.37	0.58	0.50	0.63
Cat5	0.73	0.63	0.72	0.80	0.73	0.70
Rest	0.89	0.93	0.93	0.87	0.91	0.90
Avg	0.85	0.85	0.86	0.84	0.85	0.85

Text-pattern matching. As expected, precision and recall mostly improve from A-B to AB-C. The test ABC-D, however, somewhat surprisingly shows a decrease in both precision and recall, despite the text patterns being presumably improved. One possible explanation of the performance drop could be over-fitting (over-training) of the text patterns. An increased number of "training" messages leads to an increased number of text patterns and an increased level of details in each pattern, which makes the set of text patterns more complex and eventually messy, which may lead to a performance drop. Another explanation could be the use of a different test collection for each test; some test data sets may be harder or easier to categorize than other test data sets.

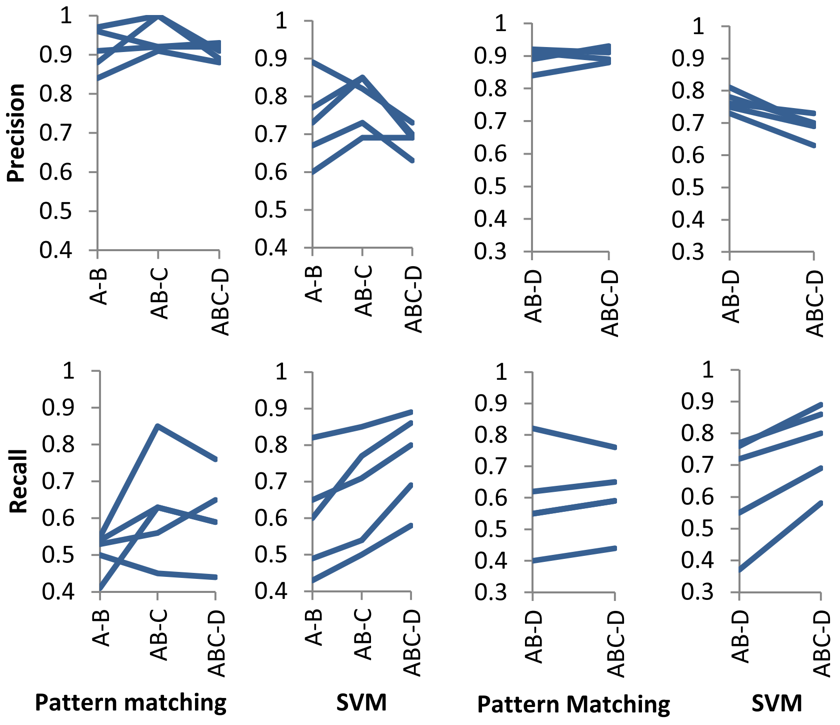

Fig. 1. Precision and recall trends for Cat1 through Cat5 between the tests A-B, AB-C, ABC-D (two left columns), and between the tests AB-D, ABC-D (two right columns). For text-pattern matching in the tests AB-D and ABC-D, the recall values in Cat1 and Cat5 are identical, therefore the lines overlap.

The tests AB-D and ABC-D use the same test collection D and demonstrate a cleaner effect of stepwise improvement between the "training" collections. The performance trends look more cheerful than those between AB-C and ABC-D. Still, precision drops for Cat2 and Cat4 while their recall rises; recall drops for Cat3 while the precision rises. These three categories are small, however, and the performance figures are influenced by only a few misplaced messages. For the entire test collection D, there is a small increase of both precision and recall between AB-D and ABC-D. McNemar's test [14] for the results of AB-D and ABC-D showed that the performance increase was statistically significant on the 5% level.

SVM. The tests AB-D and ABC-D demonstrate that more training data increase recall and decrease precision. Because Cat1 through Cat5 are relatively small text categories (e.g., Cat4 has only 59 training messages in AB-D, 81 in ABC-D), SVM gets more aggressive about classifying emails into these categories. This means more true-positive category placements – higher recall. And because the amount of the training data is still small, also more false positive category placements – lower precision.

A review of automated email answering methods [15] puts our results into context.

6 Error Analysis of Text-Pattern Matching

We manually inspected the reasons for incorrect category placements after the test AB-C. Collection C has 2479 messages. 428 messages in Cat1 through Cat5 where correctly placed, 248 were incorrectly placed into Rest. 34 messages were incorrectly placed into one of the categories Cat1 through Cat5.

Missing answer, unanswered message means that the message was incorrectly placed into Rest. We established 9 error classes for that. Table 6 summarizes the distribution of unanswered messages per error class and text category.

Err1: Unique wording. The wording in these messages does not resemble any text pattern in the system's database. There is little we can do to answer these messages. If the number of training messages rises and some of the unique messages start exhibiting common wordings, then we can create new text patterns to cover them.

Err2: Similar text pattern available. This error class is similar to Err1, except there are one or more text patterns relevant to the unanswered message. These text patterns include concepts that appear in the message, but wording of the details around these concepts is not covered by the text patterns. It is unclear how much we would benefit from modifying an existing text pattern in order to answer this particular message. If, however, the number of training messages rises, some of the unanswered messages may become useful for improving existing text patterns.

Err3: Missing synonym or word order. There exists an almost matching text pattern in the system's database, but this text pattern did not match the message because of one concept. In the text pattern, the concept was represented by a number of synonyms, whereas the message used different one. A synonym may be a word or an expression, e.g., "understand" vs. "make out". In some cases, the unanswered message used a generalization of a particular concept. If we want a text pattern to include more general synonyms, we need to redesign the text pattern and include expressions that specify the context in which these more general synonyms are used. Not specifying the context will lower precision.

In very few cases, the text pattern did not match because it stipulated the opposite sequence of two matching words.

It should not be difficult to automatically discover contextual synonyms for existing text patterns.

Err4: Missing disambiguation in the query message. This error class is similar to Err3. There exists a nearly matching text pattern in the system's database. Much of the text pattern, usually a general key phrase, did match the message. Still, the text pattern also contained a number of disambiguating expressions that specified the context of the general key phrase, and none of these expressions matched the query message. For example, the statement "I'd like to have my money" can be placed in different contexts and mean different things. For a complete match we require that a disambiguating expression such as "for March-April" matches. Then we can assume that the writer is asking about a payment of social benefit for the given period because the message is sent to an authority whose main business is paying social benefits.

Err5: Missing subject or action word in the message. This error class is similar to Err4. There exists a nearly matching text pattern in the system's database. Still, the nearly matching piece of text in the message did not contain the subject or action word required by the text pattern for a match. Taking the subject or action word out of the text pattern would require adding disambiguating context expressions. For example, the unanswered message says "I don't have my allowance", while a matching text pattern requires something like "I don't have my allowance paid".

Err6: Too nuanced message. This error class is similar to Err2. The message is rich in specific details that describe the context of the message. Nonetheless, it is difficult to match this wealth of details to recurring text patterns.

Err7: Vague message. The message may be understood by a human who knows the overall context and can reason, but the information need or the purpose of the message is not clearly expressed, i.e., the response trigger is not clear or missing.

Err8: Untreated misspelling. Although the system does correct spelling mistakes, some misspellings may be left untreated and relevant text patterns may fail to match. Some misspellings may be correctly spelled different words, such as "maid" instead of "made". Non-standard abbreviations can be considered misspellings here.

There is little we can do with text patterns in order to treat spelling mistakes missed by a spellchecker, though we can treat non-standard abbreviations as missing synonyms in Err3.

Err9: Ill-structured message. The message lacks proper sentences, or a sentence resembles a collection of words.

Table 6. Distribution of unanswered messages per error class and text category.

	Err1	Err2	Err3	Err4	Err5	Err6	Err7	Err8	Err9	Total
Cat1	19	7	7	1	2	1	1	2		40
Cat2	20	18	3	1			1	3		46
Cat3		1	5	1			1			8
Cat4	6	2	1			1	1		1	12
Cat5	41	23	27	4	9	18	10	5	5	142
Total	86	51	43	7	11	20	14	10	6	248
Share %	34.7	20.6	17.3	2.8	4.5	8.1	5.6	4.0	2.4	100

Wrong answer means that the message was incorrectly placed into one of the categories Cat1, Cat3, or Cat5; Cat2 and Cat4 did not have any wrong answers. We established 3 error classes. Table 7 summarizes the distribution of incorrectly answered messages per error class and text category.

Err10: Related context. The system has found a matching piece of text in the message, but the meaning of the text pattern there is different because of different overall context in the message, which invalidates the standard answer.

Err11: Context changes in a subordinate clause. The system has found a matching sentence, but a subordinate clause of this sentence changes the context of the text pattern and invalidates the standard answer.

Err12: Meaning changes because of a word. The system has found a matching sentence, but there is a word or a phrase in the sentence that invalidates the expected meaning of the text pattern and the standard answer.

Table 7. Distribution of incorrectly answered messages per error class and text category.

	Err10	Err11	Err12	Total
Cat1	4		2	6
Cat3		4		4
Cat5	12	1	11	24
Total	16	5	13	34
Share %	47.1	14.7	38.2	100

7 Conclusions

In automated email answering, the system's ability to identify the context and the request (we call it response trigger) stated in a query message is crucial. In about 52% of our email messages, the context and the response trigger is one question (see Table 2). For these messages, standard question-answering and text categorization techniques may work well. Some other 20% of the messages are challenging: the message is vague or the response trigger does not contain representative keywords. Having this in mind, we tested a text-pattern matching technique where each text pattern was designed to match a context description and a response trigger.

The system achieved good precision – about 90% of the messages were correctly assigned a standard answer. The recall varied, mostly lying in its 50-ies and 60-ies. The baseline system, SVM, achieved similar or slightly better recall while having considerably lower precision (see Table 4, Table 5, Figure 1). Assuming that manually crafted text patterns yield the highest quality of text-pattern matching, we believe that our performance figures illustrate the upper performance limits of text-pattern matching, or perhaps statistical (without semantic processing) text matching for a large and diverse email collection where precision holds priority.

We have observed that more "training" data, which allows improving the text patterns, does not always increase both precision and recall simultaneously for small (around 100 messages) text categories. With different test collections (e.g., continuous email flow) the precision and recall change is even more unpredictable.

We have discovered 12 reasons why text-pattern matching may fail. 9 reasons cause a "missing answer", i.e., the system fails to place a message into the category where it belongs. About 35% of the "missing answers" could be relieved by more "training" data (see Section 6 and Table 6). 3 reasons cause a wrong answer.

References

1. Downey, D., Etzioni, O., Soderland, S., Weld, D.S.: Learning text patterns for web information extraction and assessment. In: Proc. AAAI-04 workshop on adaptive text extraction and mining, pp. 50-55 (2004)
2. Sneiders, E.: Automated FAQ Answering with Question-Specific Knowledge Representation for Web Self-Service. In: Proc. 2nd International Conference on Human System Inter-action (HSI'09), May 21-23, Catania, Italy, pp.298-305, IEEE (2009)
3. Sneiders, E.: Automated Email Answering by Text Pattern Matching. In: H. Loftsson, E. Rögnvaldsson, S. Helgadóttir (eds.): Proc. 7th International Conference on Natural Language Processing (IceTAL), August 16-18, Reykjavik, Iceland, LNAI 6233, pp. 381-392. Springer, Heidelberg (2010)
4. Kosseim, L., Beauregard, S., Lapalme, G.: Using information extraction and natural language generation to answer e-mail. Data & Knowledge Engineering, vol. 38, pp. 85-100 (2001)
5. Khosravi, H., Wilks, Y.: Routing email automatically by purpose not topic. Natural Language Engineering, 5, pp. 237-250 (1999)
6. Corston-Oliver, S., Ringger, E., Gamon, M., Campbell, R.: Task-focused summarization of email. In: Proc. ACL-04 Workshop: Text Summarization Branches Out, pp. 43-50 (2004)
7. Lampert, A., Dale, R., Paris, C.: The nature of requests and commitments in email messages. In: Proc. of the AAAI Workshop on Enhanced Messaging, pp. 42-47 (2008)
8. Lampert, A., Dale, R., Paris, C.: Requests and Commitments in Email are More Complex Than You Think: Eight Reasons to be Cautious. In: Proc. Australasian Language Technology Association Workshop, vol. 6, pp. 64-72 (2008)
9. Goldstein, J., Sabin, R.E.: Using speech acts to categorize email and identify email genres. In: Proc. 39th Annual Hawaii International Conference on System Sciences, vol. 3. IEEE (2006)
10. Dalianis, H., Sjöbergh, J., Sneiders, E.: Comparing manual text patterns and machine learning for classification of e-mails for automatic answering by a government agency. In: Proc. Computational Linguistics and Intelligent Text Processing, pp. 234-243. Springer Berlin Heidelberg. (2011)
11. Sneiders, E.: Automated FAQ Answering: Continued Experience with Shallow Language Understanding. In: Question Answering Systems. Papers from the 1999 AAAI Fall Symposium. Technical Report FS-99-02, November 5-7, North Falmouth, Massachusetts, USA. AAAI Press, pp.97-107(1999)
12. Sneiders, E., Eriksson, G., Alfalahi, A.: Exploring the Traits of Manual E-Mail Categorization Text Patterns. In: Advances in Natural Language Processing, pp. 337-344. Springer International Publishing (2015)
13. Tang, G., Pei, J., Luk, W. S.:. Email mining: tasks, common techniques, and tools. Knowledge and Information Systems, 41(1), 1-31 (2014)
14. Everitt, B.: The Analysis of Contingency Tables. Chapman & Hall, London (1977)
15. Sneiders, E.: Review of the Main Approaches to Automated Email Answering. In: Proc. WorldCist'16, Recife, Brazil, March 22-24. Springer International Publishing (2016)