To Act Or Not To Act - Annotating and Classifying Email Regarding Necessary Action

Veronika Hintzen
LMU Munich, Germany
v.hintzen@campus.lmu.de

Alexander Fraser
Center for Information and Language Processing
LMU Munich, Germany
fraser@cis.lmu.de

Abstract

Every user of email is aware of the problem of reacting to emails that require a time-sensitive action by the recipient while being overwhelmed by informational emails. We define a new classification problem to capture this distinction, creating comprehensive annotation guidelines and carrying out annotation. We carry out a proof-of-concept implementation of a classifier and discuss our future research which will result in a tool that is usable in an everyday business environment.

1 Introduction

The usage of email as a major communication tool has grown over the past 20 years. As per the current email statistics report by the Radicati Group, it was estimated that 3.8 billion users would receive 281.1 billion emails per day in 2018 with an estimated growth of about 4.4 percent each year (Radicati-Team, 2018). So a user receives on average about 74 emails per day. Carreras and Màrquez i Villodre (2001) discuss how users spend too much time sorting, with one problem being spam. But whereas spam filters nowadays work more and more efficiently and instant messenger services such as WhatsApp, Signal and Threema are on the rise for private communication - and thus keep the major load of non-work-related mail from our mailboxes, and for example Googlemail already provides a topic related sorting of the remaining emails to their users - many emails people receive at work still don't require immediate attention. In business, most emails still are basically only for information purposes, such as a report of a meeting or an invitation to a workshop. While these emails might be relevant and perhaps even time critical, there is no-one waiting for the recipient's reaction to the email. One can assume that emails that contain a question or a task would need to be prioritized higher than an invitation. Sorting through those emails and setting priorities by hand often takes up a lot of time and can be seen as a major distraction in a stressful work environment. Information emails get more attention than necessary, important emails get overlooked easily and the time that could be used for working on assigned tasks or even for breaks is diminished by the sheer amount of emails one has to manage.

While email providers nowadays allow users to create simple filters based on keywords, setting up these rules still takes up a lot of time and can be difficult (Gupta and Goyal, 2018). As was noted by Carreras (2001), most users waste a large amount of time in managing their emails or they prefer not to use keyword-based rules for filtering their email inbox. So, an automated tool that classifies emails regarding the expected attention that needs to be provided to them could help with prioritizing the received emails and thus improve the efficiency of work related communication.

Text classification in general and classification of emails in particular is a major subject in computational linguistics. Sebastiani (2001) defines it to be "the activity of labeling natural language texts with thematic categories from a predefined set" and considers it to be an instance of text mining, since "'text mining' is increasingly being used to denote all the tasks that, by analyzing large quantities of text and detecting usage patterns, try to extract probably useful (although only probably correct) information." Thus, classifying emails regarding an action that is possibly expected from the recipient by the transmitter can be broken down into a bi-label or multi-label text classification problem depending on the desired degree to which the expected action should be distinguished. The general idea is to have a predefined set of labels or classes and find the class that best fits a given text. In this case, a binary label classification would simply be to sort the email into one of the two categories *ac-*

tion required and *no action required*, depending on whether there is any text in the email that indicates that the addressor of the email expects the recipient to become active in any way - for example to answer a question or to do a task. But it could also be interesting for the recipient to further discern between these two options and to have the emails labeled according to the degree of action that is required, as we will discuss later.

The paper is structured as follows. First a short overview over the annotation process will be given with an example email. Then the experiments done on these mails will be described and analyzed. This is followed by an outlook on possible improvements and a conclusion.

2 Annotation

In December 2001 the Enron corporation, one of the biggest energy companies of the US at the time, declared bankruptcy. In the ensuing investigation by the Federal Energy Regulatory Commission about 500000 emails by over 150 users were released to the public. It is perhaps the biggest publicly available email corpus and since then has been very popular with researchers. It contains a large variety of business emails as well as spam mails and private mails. A SQL-dump by Ruhe (2016) of the Enron data set was used. This is basically a "repaired" version of the MySQL-dump that was originally created by (Shetty and Adibi, 2004) but is no longer available. This MySQL-database contains all the info from the emails in a clean and easily retrievable format. For saving the annotations, the message-table was simply extended with the columns *label, notes* and *reviewed*.

Consider the email presented in table 1. The sender of this email obviously expects the recipients of this email to become active, which is implied by the following wording: "Can the two of you coordinate..." which would lead to this email being annotated with the *Action Required (AR)* category. But at the end of the email, the sender asks a question: "Can we get together that morning and review your analyses?" This means, the addressor expects a reply by the recipients, confirming this request for a meeting or maybe an alternative proposal. This leads to the annotation with *Reaction Required (RR)*.

Since the categories are considered to be hierarchical, a requested reaction is considered to be a little more important than an action, since the addressor might wait expectantly for the reply. Therefore, the possible *RR* annotation trumps the also possible *AR* annotation.

Also, as you can see, the email ends with "888-582-7421thankskh", where obviously some whitespaces went missing, leading to the weird string "7421thankskh" being considered a token in this email.

1240 emails were randomly selected and sorted into one of eight categories. Table 2 shows them in their hierarchical order with their respective abbreviation, their count and a short description.

These labels were selected for their relevance regarding a work situation. While private emails might still have some of the same cues as business emails the annotators perceived their language and subjects as so strongly differing from the business emails that it was decided to create an own label for them. Some of these private mails for example contained jokes, cooking recipes or discussions about 9/11. For further information on the definition of these categories, the annotation guidelines are available at `http://hintzenv.wordpress.com`.

While non-relevant emails were usually easy to annotate, sometimes missing context made it hard to decide on a label. These emails were annotated "Unsure" and reviewed again later. Those that still would not be clear stayed in that category. Also, the announcement of a birthday cake at a colleagues cubicle led to some discussion. It was decided it would be considered to be an invitation.

3 Experiments

3.1 Evaluation Techniques

To evaluate a classifier, after the classification of the test data, the appointed classes need to be compared with the originally annotated classes. There are several common practices to evaluate the performance of a classifier. Precision, recall and f-score were computed for each class used by each of the implemented classifiers. Confusion matrices are also presented.

3.1.1 Micro and Macro scores

For a general overview of the different models, the micro and macro averaged scores will be computed, which show a weighted (micro) and unweighted (macro) average score of the performance of a model.

The procedure for computing these scores is

subject	body
SFV Rate Design for Sun Devil	James &David –\t Can the two of you coordinate a revenue model for Sun Devil that incorporates a Straight - Fixed Variable rate design along the following parameters using 15, 20 and 25 year terms: San Juan utilization: 85% of 780,000/MMBtu/d Mainline: 85% utilization of 810,000/MMBtu/d Phoenix lateral: 75% utilization of 500,000 MMBtu/d Also, prepare some ROE sensitivities if the above utilization falls by 10% and rises by 10%. I will be out of town till Tues 11/13. Can we get together that morning and review your analyses?page me if you have questions at 888-582-7421thankskh”

Table 1: Example email from the Enron data set

category name	label	count	description
Reaction Required	RR	259	A reaction to the email is required.
Action Required	AR	87	The recipient is required to take an action.
Appointment/Deadline	AD	94	The email contains an appointment or deadline.
Invitation	I	25	The email contains an invitation.
Contains Information	CI	518	The email contains business-relevant information.
Private	P	135	The email is private, not business relevant.
Non-Relevant	NR	113	The email is business-related, but not relevant (e.g., newsletters).
Unsure	U	9	This is a catch-all category, see the discussion in the text.

Table 2: Categories with respective counts

described by Yang (1999), and Tsoumakas et al. (2010) present the respective formulas. Furthermore, for the micro average score, Asch (2013) shows, that for single-label classifiers the scores for precision and recall are equal. Since the F-Score is the harmonic average between Precision and Recall, and for the micro-average-score, those two scores are equal, so is the F-Score. In this paper, the macro averaged F1-Scores (later in this paper referred to as *macro score*) will be compared with the micro averaged F1-Scores (later in this paper referred to as *micro score*).

3.2 Preprocessing

In order to get the emails into a processable format, the bodies have to be tokenized: special characters, punctuation, tabs and newlines were filtered out and the text was split on blanks. We use Word2Vec word embeddings and create a single average vector for each document. We split the annotated data into 80 percent training data and 20 percent test data.

Because the annotated data has been labeled in a way that is quite fine grained and the counts vary greatly between the categories, the classifiers were tested on different groupings of the labels, which will be described in detail in the evaluation section. These groupings were selected by their intuitive relevance to everyday working life and in the hopes of finding a grouping that gives a balanced overall performance.

3.2.1 Word Embeddings

For this study, Word2Vec word embeddings were used. Word2Vec models with the dimensionalities of *50, 100, 200* and *300* were trained on the Enron data set and thus on about 62 Million tokens - and a vocabulary of about 650 thousand unique tokens. We do not use pre-trained Word2Vec embeddings because the Enron data set consists of emails, and emails are of a different nature than, for example, Wikipedia articles regarding the used vocabulary, syntax and the existence of many typos. The idea was to use Word2Vec embeddings trained on the Enron data set in order to better account for these errors and inconsistencies. We also tried pre-trained GloVe embeddings in initial experimen-

	micro	macro
Naïve Bayes	**0.583**	0.426
SVC baseline	0.538	0.321
Word2Vec 50	0.551	**0.501**
Word2Vec 100	0.571	0.489
Word2Vec 200	0.543	0.473
Word2Vec 300	0.575	0.499

Table 3: Micro and macro scores of all models on non-grouped classes

tation, but the results were much worse, and so we did not continue experimentation with them. We leave further study of this issue for future work.

3.3 Classification

Afterwards the classifier was trained on the vectorized texts with their respective labels from the training sets and with the learned features the vectorized texts from the test set are classified. In our study, six different classifiers (Naïve Bayes, baseline SVM on words, four SVMs for the different dimensionalities of Word2Vec embeddings) were combined with different groupings of the labels.

3.4 Evaluation

In order to evaluate the performances of the different classifiers first an overview of the micro and macro overall scores will be shown. For a more detailed look into the models, the Precision, Recall and F1-Scores of the classes in the best- and worst-performing models will be presented.

3.4.1 Overview

A first test with separate categories produced very unsatisfactory results. Table 3 shows the weighted and unweighted average F1-Scores for each of the models. For purposes of readability the scores have been rounded to the third decimal place.

According to the (weighted) micro-Score, the Naïve Bayes classifier performs best at this task, but a look on the unweighted score shows that the small classes are classified significantly worse than the larger classes. With the (unweighted) macro-Score, the model that resulted in the highest score in our tests would be the Support Vector Classifier based on the 100-dimensional Word2Vec-embeddings.

Overall, these scores are not really satisfactory. This is not surprising due to the small size of training data per class. In the test set, the smallest class only had seven occurrences. So, the idea arose to group the categories in order to get more training and testing examples for each class.

Name	**Category grouping**
7-base	RR, AR, AD, I, CI, P, NR
2-action	RR+AR, AD+I+CI+P+NR
2-timecrit	RR+AR+AD, I+CI+P+NR

Table 4: Keywords assigned to groupings

A more detailed look into the performances of the models with the base task will follow in section 3.4.2, where the best- and worst-performing models will be discussed in detail.

For an overview of the performances, tables 5 and 6 compare the micro and macro scores for each grouping in each of the models. The weighted micro scores take into account the size of the groups. The unweighted macro-scores do not do that. While usually one would think that the weighted performance of a model would give more insight into the performance of a model, we decided to add the unweighted scores since the category containing *CI* shows the highest F1-Scores due to the size of the corresponding data set but is one of the lesser important categories, and as such the weighted scores tend to skew the performances in favor of the bigger and less important categories.

We now discuss two further groupings of the labels we experimented with. Category groupings are assigned a name, leading with the number of classes the grouping results in, followed by a short keyword for the way criteria they are grouped for. In table 4 you can see these names with their respective assigned grouping.

For reasons of readability, again the scores were rounded to the third decimal places and the model names have been abbreviated: *nb* for *Naïve Bayes*, *svc bl* for *baseline Support Vector Classifier*, *wv* for the SVC using the self-trained Word2Vec embeddings. Also, for reasons of clarity, the table 5 will refer to the micro-scores (weighted), while table 6 will refer to the macro-scores (unweighted). For each grouping the best weighted and unweighted scores are underlined. The highest weighted and unweighted F1-Scores across all models are shown in **bold**.

As one would expect, the best performing groupings are those that contain only two classes and the grouping with each label on its own performs the worst. Also, as expected, the unweighted

	7-base	2-action	2-timecrit
nb	0.583	0.725	0.656
svc bl	0.538	**0.757**	0.725
wv50	0.551	0.729	0.676
wv100	0.571	0.696	0.709
wv200	0.543	0.721	0.692
wv300	0.575	0.741	0.700

Table 5: Comparison of micro-Scores

	7-base	2-action	2-timecrit
nb	0.426	0.574	0.616
svc bl	0.321	0.612	0.641
wv50	0.501	0.611	0.630
wv100	0.489	0.580	**0.667**
wv200	0.473	0.622	0.646
wv300	0.499	0.652	0.658

Table 6: Comparison of macro-Scores

		p	r	f1
svc bl	AD+I+CI+P+NR	0.77	0.95	0.85
	RR+AR	0.67	0.26	0.38
w2v100	I+CI+P+NR	0.77	0.80	0.78
	RR+AR+AD	0.57	0.53	0.55

Table 7: Scores of the best-performing models

	Actual						
	RR	AR	AD	I	CI	P	NR
RR	16	4	2	-	15	1	2
AR	1	2	-	-	5	-	-
AD	2	-	7	-	4	-	-
I	-	-	1	4	1	-	-
CI	29	12	4	3	80	4	4
P	3	-	-	-	3	13	5
NR	-	-	-	-	5	1	14
Total	51	18	14	7	113	19	25

Table 8: Confusion matrix for Word2Vec50 based SVM on base task *7-each*

scores are almost consistently lower than the micro scores while the pre-trained embeddings have an almost consistently worse average F1-Score than the Word2Vec embeddings. While the differences especially in those scores that are very close to each other cannot be considered statistically significant, this paper only strives to discuss the possibility of the task and possibly provide scores for comparison with similar future tasks.

It is noteworthy that, when comparing the micro scores for one grouping, the scores are surprisingly uniform with at best a difference of 0.081 between the worst and best performing models and even only 0.061 in the *2-action*-grouping. This can be attributed to the class sizes. In the micro score the larger a class the higher the influence on the resulting average score. A look on the respective precision and recall scores of the classes shows consistently good performance on these larger classes in all the models.

In order to see how a model improves when being trained on fewer classes with more training examples, you can compare horizontally and see a mostly consistent increase in performance from seven classes to two classes.

When looking at the two binary groupings, with micro averaged scores, it seems as though the switch of the *AD*-class from the larger class to the smaller group actually decreased the overall performance. But in table 6, you can see that there, too, is an improvement. This leads to the conclusion that the performance of the larger class drops, while the performance of the smaller class - which would be considered more important in a business environment - improves.

When looking at the micro averaged scores, the comparison of higher dimensionalities of the Word2Vec embeddings with the baseline SVC also seems notable. While with more classes the higher dimensionalities seem to add to the performance, with the binary classifiers, the higher dimensionalities perform even worse than the baseline.

With the macro averaged scores, this effect vanishes and the Word2Vec-models perform consistently better than the baseline model, again indicating that the baseline SVM has a bias towards larger classes.

3.4.2 Detailed discussion of best- and worst-performing models

For a detailed analysis of the best- and worst-performing models, table 7 shows the precision, recall and f-scores for each class. For the model performing best regarding a weighted calculation of the average value - the SVC baseline model with the *2-action*-grouping (5:2) - , the recall is very high for the larger group containing the less important categories while the recall with the important categories is very low with only about 0.26. Using the best model by macro score, you have a lot less variability within the scores.

The worst-performing model with both, the micro scores as well as the macro scores, was the SVM with the 50-dimensional GloVe-embeddings in the "7-base" task - where there were no groups, but each label for its own. In fact the labels *RR, AR* and *I* were not classified at all - with 0.00000-scores, resulting in the low macro scores. We do not present results on pre-trained GloVe embeddings in detail, leaving a study of how to adapt pre-trained embeddings to the Enron corpus for future work.

3.4.3 Evaluation and Error Analysis - Word2Vec50 with the *7-each* grouping

With the *7-each* "grouping" being the base task of this project, the best performing model of this task will be discussed here.

In order to get a better look on the distribution of the actual and predicted classes, in the following the confusion matrix for the 50-dimensional word2vec model (see table 8) will be presented.

3.4.4 Evaluation and Error Analysis - SVC baseline model with *2-action*

In order to get more detail on the performance of the model, and to get an idea of where the performance issues arise from, a detailed confusion matrix for the svc baseline model with the category *2-action* grouping is presented in table 9.

For readability the larger group (*AD+I+CI+P+NR*) will be shortened to *Other* and for reasons of space-usage, the confusion matrix will have the actual categories on the X-axis and the predicted categories on the Y-axis.

While the performance here is significantly better than with the GloVe-models and at least $\frac{2}{3}$ of the mails labeled with *Action* also really require said action, still 51 of the 69 mails will be lost - that's almost $\frac{3}{4}$.

If you look at the distribution of the *RR* and *AR* emails you can see that with 0.28 the share of correctly classified emails in the *AR* category is only slightly bigger than the 0.26 in the *RR* category. But if you consider that the *AR* category is significantly smaller than the *RR* category, with only one additional misclassified email, that percentage would have dropped to 0.22. So, it is safe to assume that both labels are classified with a comparable performance.

What is interesting to see, though, is, that within

	Actual						
	RR	AR	AD	I	CI	P	NR
AR+RR	13	5	-	-	6	3	-
Other	38	13	13	7	108	16	25
Total	51	18	13	7	114	19	25

Table 9: Confusion matrix for svc baseline model with *2-action*

	RR	AR	AD	I	CI	P	NR
AR+RR	23	6	2	-	21	-	1
Other	28	12	11	7	93	19	24
Total	51	18	13	7	114	19	25

Table 10: Confusion matrix for Word2Vec300 model with *2-action*

the group of emails that were wrongly classified as $AR + RR$, these emails originally stem only from the *CI* and *P* categories. All other categories were classified correctly. But here with three of the nine wrongly classified emails being *P* category emails, the misclassification of *P* emails is significantly higher (0.16) than of the *CI* emails (0.05). This might be traced to the fact that often private emails also contain requests for an action or a reaction.

For comparison, consider the confusion matrix of the Word2Vec300 model trained with this grouping that performed second best to the baseline svc - best with the macro Score, see table 10.

This confusion matrix produces a very different picture than the baseline svc. Here already $23 + 6 = 29$ of the $51 + 18 = 69$ actual action requiring emails are found - which is already over 40 % -, there are a lot more misclassifications towards the smaller class.

3.4.5 Evaluation and Error Analysis - Word2Vec100 model with the *2-timecrit* grouping

In order to again get a better look at the performance of the Word2Vec100 model in the *2-timecrit* grouping, see table 11.

With this SVC and grouping - while here, too, there are a lot more misclassifications toward the smaller class, instead of $\frac{3}{4}$ of the action requiring mails being "lost", of the total 82 mails regarded as *ActionRequiring*, only 38 are missed. With that being less than the half, that's already a lot less than with the baseline svc in grouping 5.

It is interesting to see that with the *AD* emails there is an unusually high recall with 10 of 13 being classified into the action requiring group.

For comparison, here, too, shall be presented the confusion matrix for the baseline svc model which performed second-best in the *2-timecrit* grouping (table 12).

Here, again, the *AD* category - while having only eight of the thirteen emails classified correctly - performs surprisingly well. Again, the misclassification counts towards the smaller class are smaller, while those towards the larger class are stronger.

This leads to the conclusion that overall the vectors produced by the CountVectorizer lead to a tendency of classifying in support of the larger class and the vectors produced, while the average document vectors resulting from the Word2Vec-embeddings lead to a tendency of classifying in support of the smaller class.

In an everyday office life the latter would probably be preferable. Consider an email account containing different folders for each class and the user - running from one appointment to another - only wanting to see the action-relevant emails when looking into the respective folder. While having emails there that don't belong would be considered a nuisance, missing emails might prove to be a problem of a lot bigger scale.

4 Outlook

In this section, an outlook on possible improvements that can be made on and with the existing models, as well as ideas for future work - i.e. possible variations of the tasks - will follow.

While none of the models implemented yet proved to be adequate for an everyday use in a work environment, there are several possibilities to improve the performance and ideas that might prove to be worth looking into.

A larger annotated data set should help greatly in the training of the used models. Language is too complex to grasp meanings just from a little over a thousand emails. With larger annotated data sets, more features can be accounted for and so the non-binary models' performance might also improve. Possible ways to achieve a larger data set include: *more time, more personnel* and using *distant supervision*. Although the annotation with distant supervision produces annotations of a relatively bad quality, it might still be better than only working with a small data set. In contrast to that, the usage of more personnel would help in ensuring a high quality of annotation. So a good compromise between the two could possibly be found, where a semi-large set of high quality annotations would be combined with a large set of low quality data and where good results might be achieved by having several training iterations with only the first iteration on the combined data set and the other iterations on only the high quality data set.

	RR	AR	AD	I	CI	P	NR
AR+RR+AD	26	8	10	3	24	2	4
Other	25	10	3	4	90	17	21
Total	51	18	13	7	114	19	25

Table 11: Confusion matrix for Word2Vec100 model with *2-timecrit* grouping

	RR	AR	AD	I	CI	P	NR
AR+RR+AD	18	3	8	-	12	3	1
Other	33	15	5	7	102	16	24
Total	51	18	13	7	114	19	25

Table 12: Confusion matrix for svc baseline model with *2-timecrit* grouping

Another possibility to get larger annotated data sets would be to use active learning and let users help with annotating emails in a run-time environment and therefore improve the used classifier according to the user's needs. This would also have the advantage of having the models trained on more contemporary business emails being adapted regarding the respective business area of the user and to the change of their vocabulary in the past years.

With larger data sets the application of deep learning models would become possible. With the *Support Vector Machines* much information is lost during the reduction of the embeddings to an average document vector, so a model that is able to properly grasp the multiple dimensions of the embeddings could possibly find more and better relations between the features and the corresponding classes of the documents and thus make better classifying decisions.

In addition, one possibility to improve the performance could be found by including the subject lines as well as info about the sender and the recipient/s of the emails into the models since these already provided relevant information about the context of the mail during the annotation process.

In emails you often have the former emails from the exchange appended to the latest email. While in the annotation process these old emails sometimes provided necessary information on the context of the email, in a bag-of-words model, that is used in

Naïve Bayes as well as in Support Vector Machines, would give these old mails too much weight. Especially in long email exchanges, with five and more emails and possibly only a short question in the latest mail. A weighting of the words depending on their occurrence location might prove to be useful.

The Enron data set has often been used for training classifiers for spam mail. Instead for this project the obvious spam mail has simply been categorized as Non-Relevant. The implementation of a spam filter based on a larger spam-specific training set that is run before our classifier to eliminate obvious spam might also improve the results (and/or help with future annotation of new data).

Several of the produced errors might have their origin in the hierarchical nature of the categories. This problem might be evaded by allowing the annotation with more than one category per email and/or by using classifiers that produce more than one label per email and then tweaking these classifiers by weighing those categories in favor of the action-inducing categories.

5 Conclusion

For this paper, a classification task was set up from scratch. The goal was to build a classifier that could distinguish between emails regarding whether a response or other action was required from the recipient. Without suitable annotated data being accessible and possibly even existent, first, a data set had to be annotated by hand.

Being probably the biggest open source data set for business emails, the Enron data set was selected as a foundation for this self-annotated data set. The annotation produced an annotated data set of 1240 emails. Due to the nature of the Enron data set, the distribution of the categories was rather imbalanced leading to very different sizes of learning and test sets for each of the categories.

For the task of building a classifier, it was decided to compare six different models: one Naïve Bayes classifier, a baseline Support Vector Classifier as well as four Support Vector Classifiers based on Word2Vec embeddings of different dimensionalities.

Since the performance of these classifiers on the base task with a class for each of the categories was not satisfying, the categories were grouped in different ways with the goal of finding a grouping that would perform better and still be of practical use in an everyday work environment.

As was to be expected, the two groupings leading to binary classifiers performed far better than the multi-class classifier. With a binary classifier, the training and test data sets were both bigger and much more balanced. Also, with a binary classifier, there are far less classes that can be taken for misclassification. Also, it was interesting to see, how the performance of the embeddings-based models changed with additional dimensions. While the Word2Vec embeddings would produce varying micro and macro scores, we noticed that the pre-trained GloVe embeddings - while over all having a worse performance than the Word2Vec based models - showed constant improvement of performance with additional dimensionalities on each of the tasks (but we omit these detailed results). Another interesting result was that the two baseline models had a surprisingly good performance overall.

In the detailed error analysis of the best- and worst-performing models, it was implicated that additionally to the small size of the data set, the hierarchical order of the categories might have been one of the major origins of misclassifications, since this led to fewer distinguishable features of said categories. The most improvement could possibly be achieved by improving the used data set regarding its size and quality. But also with the used classifying models, there are many possible tweaks and changes that could be tested and that might prove to have quite an impact on the classifying performance.

In conclusion - while at least on the binary tasks promising results could be achieved - none of the presented models has a performance that would be good enough for practical use. Too many misclassifications would make a tool based on the models used here very frustrating to work with. Also, this would probably even lead to financial risks when an email that requires a time-sensitive action by the recipient, would not be recognized as such by the classifier. But even with these not yet satisfactory results, it was shown, that this task is not impossible to achieve but rather a question of obtaining bigger data sets. Using the annotation guidelines and initial data set that we have created in this work (and make available with the publication of this paper), it will be possible for interested researchers with access to more resources to create a much larger training corpus than we were able to create. In addition, we plan to study how to incorporate *active learning* to learn from the user as they identify mis-categorizations as an additional way to obtain further supervision for this important task, see, e.g., the work of Tong and Koller (2002), as well as more recent work. Finally, once we have further supervision available for this task, we will study (data-hungry) classification models based on neural networks, from which we expect to obtain further improvements in performance.

Acknowledgments

The hardware that was used for this research was provided by the Steering Lab at Horvàth & Partners Munich.

References

Vincent Van Asch. 2013. Macro-and micro-averaged evaluation measures [[basic draft]].

Xavier Carreras and Lluís Màrquez i Villodre. 2001. Boosting trees for anti-spam email filtering. CoRR, cs.CL/0109015.

Deepak Kumar Gupta and Shruti Goyal. 2018. Email classification into relevant category using neural networks. CoRR, abs/1802.03971.

Radicati-Team. 2018. Email statistics report, 2018-2022. Technical report, The Radicati Group, Inc., March.

Arne Hendrik Ruhe. 2016. http://www.ahschulz.de/enron-email-data/.

Fabrizio Sebastiani. 2001. Machine learning in automated text categorization. CoRR, cs.IR/0110053.

Jitesh Shetty and Jafar Adibi. 2004. The Enron email dataset database schema and brief statistical report. Technical report.

Simon Tong and Daphne Koller. 2002. Support vector machine active learning with applications to text classification. J. Mach. Learn. Res., 2:45–66, March.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. 2010. Mining multi-label data. In Data Mining and Knowledge Discovery Handbook, pages 667–685.

Yiming Yang. 1999. An evaluation of statistical approaches to text categorization. Information Retrieval, 1(1):69–90, Apr.