Activity Modelling using Email and Web Page Classification

Technical Report Number 573

July 2005

Belinda Richards, Judy Kay, Aaron Quigley

ISBN 1 86487 737 5
Activity Modelling using Email and Web Page Classification

Belinda Richards, Judy Kay, Aaron Quigley
School of Information Technologies
University of Sydney
{brichard, judy, aquigley}@it.usyd.edu.au

Abstract. This work explores the modelling of a user’s current activity using a single document and a very small collection of classified documents. A model called WeMAC was developed, which accretes evidence from heterogeneous sources to give a final classification of the user’s activity. The evidence sources considered here are the different attributes of a document. We evaluate the WeMAC model using two different document types: emails and web pages; assess its performance on both tiny document sets and larger sets; and assess its performance against a “one bag” approach. The WeMAC model compared well with existing results for systems with similar tasks and larger datasets, with an average F1 value of 0.5-0.7.

1 Introduction

A byproduct of the continued growth in the use of computing technologies is that organisations and individuals are generating, gathering and storing data at a rate which is doubling every year [1]. This growth is rapidly outpacing our abilities to handle it. Clearly however, depending on ones current activities, only parts of one’s information set may be relevant at given times. We aim to predict a user’s activity based on their currently viewed document, so that other applications may use this to deliver relevant information from the user’s store of documents. This approach is embodied in our WeMAC model, and a potential application, the WeMAC Assistant (WeMACA) is illustrated in the scenario below, adapted from [2].

Carla is attending the CHI Conference with presentations related to her research. She and her colleagues have decided to visit different presentations and share their findings at the end of the day. On the first morning, her associate James sends a reminder email detailing the presentations each of them will attend that morning. Carla opens the email using her WeMACA, and on the basis of this, it determines that she is currently “attending the CHI conference”. Using this information it displays other documents related to the activity, such as web pages relating to the speakers of the presentations and emails about the conference.

The motivation behind the WeMAC model is to use knowledge from a range of different sources of evidence to a user’s activity, from both implicit (eg. currently viewed document, location) and explicit (eg. user input) sources. Combining both knowledge types, the system reasons about the evidence given to form a conclusion.
about the user’s current activity, e.g. if the user is currently in her office then from previous monitoring it can conclude that she is engaged in “research”.

There has been no work on very tiny data sets of the sizes that we want to study. However, there are strong similarities in the work on email classification and that can at least give some hints at expected performance levels and can indicate the standards and approaches that have been applied to experiments, in terms of the numbers of users studied. When presenting the user with 1 category for their mail, SwiftFile [3] achieved approximately between 50-75% accuracy, and suggest that 50% is of borderline usefulness to users, where 75% would be useful. Iems [4] uses automatically constructed rules to suggest a classification to the user. Results of between 30-67% accuracy were achieved on small datasets on one user. On approximately 500 mail items, iems achieved between 41-70% on 2 users’ data. Using a bag-of-words and word frequency approach ifile [5] achieves an accuracy score of 86-91% across 4 users. A preliminary study with a small data set for one user ignoring not yet created categories, showed an accuracy of 88% on the first 26 items.

At present, there is no work specifically in the area of user defined web page classification. Some examples of work which focuses on classifying web pages into pre-defined categories include [6] and [7]. These systems are generally evaluated on very large datasets, ranging from approximately 4000 [6] to 8000 [7]. These training sets are orders of magnitude larger than those used in the evaluation of the WeMAC model, making a comparison between the results difficult.

The accretion approach of the WeMAC model is also related to ensembles in machine learning, where the results of different learners are combined to give a typically more accurate prediction of an example’s class.

In our current implementation, the evidence sources consist of sections of a user’s heterogeneous document set. The two document types considered are emails and web pages, the components for which are outlined in Tables 1 and 2.

![Fig. 1](image-url)

**Fig. 1.** The combinations of training and testing documents to be explored. The tail of the arrow represents the training document type, and the head refers to the testing document type.

The two main problems that we address are training on different combinations of document types, and training on tiny datasets. Each user is likely to have a different collection of documents. For example, they may have mostly one type (e.g. emails), or equal numbers of both types. However, their next document may be of either type. Ideally, the system should be able to classify documents from a variety of different training document types, so that the classification of a new document is possible regardless of the types of documents in the training set. We evaluate the performance of the WeMAC model on the classifications illustrated in Fig. 2.

The WeMAC model should also be able to perform accurately from training on a tiny number of documents. This is a difficult task, as most machine learning requires large numbers of examples before it performs well [8]. However, it is important for
the WeMAC model to perform well in this situation, as it is desirable to accurately predict a new activity which has only a small number of preclassified documents, e.g. when people expressly classify documents initially.

2 WeMAC Model Overview

WeMAC stands for Web page and eMail Accretion for Context model. The web page and email refer to the document types which are currently used as evidence. Accretion refers to the method we use to predict the user’s activity. The model is ‘for context’ as a user’s activity is part of their context [9].

The accretion approach to classification involves resolving evidence from a number of different sources. The term resolve here has a broader meaning than simply the resolution of conflicts between evidence. Rather, to resolve means to make a decision based on all available evidence – which may not necessarily be in conflict [10]. An example of the accretion approach is shown in Fig. 2. The values above the lines refer to the confidence of the source’s prediction, which is the probability that the example belongs to its assigned class. The bold lines are examples of evidence sources considered in the current implementation.

Fig. 2. An example of the accretion based approach to document classification.

We consider 3 different resolvers termed the Confidence, Weighted and Majority resolvers. The Confidence resolver uses the component which predicts with the highest confidence to make its final decision. The Weighted resolver is inspired by Multi Attribute Utility Theory [11], and weights the importance of each component’s evidence based on its previous performance (we use the average F1 value). The Majority resolver chooses the activity predicted by the most components.

There are 4 main components to the WeMAC architecture: the email and web page preparation, natural language processing, machine learning and accretion components. The email and web page preparation components extract the textual information from each email and web page. The components are outlined in Tables 1 and 2. In the situations where emails and web pages are considered in the same classification task, components are collapsed in order to have components which are common to both document types.

The Natural Language Processing architectural component extracts the necessary features for classification for each component of the document. Each document is initially tokenized using a regular expression tokeniser (from the Python NLTK
library) which considers strings of alphabetic characters (upper or lower case) as
words. In initial experiments, we also used feature reduction techniques, specifically
stop lists [12] and stemming (the Porter stemmer in the NLTK library). We found,
however, that the model performed best on tiny data sets with no feature reduction,
which is unsurprising given the small training sets used. Thus the results reported in
the Evaluation section are for experiments without feature reduction.

Table 1. Components of an email

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
<th>Collapsed Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>To</td>
<td>The information about the sender of the mail item</td>
<td>Heading</td>
</tr>
<tr>
<td>From</td>
<td>The information about the receiver of the mail item</td>
<td>Heading</td>
</tr>
<tr>
<td>Subject</td>
<td>The subject line of the mail item</td>
<td>Heading</td>
</tr>
<tr>
<td>Address</td>
<td>Any URL’s within the email. These are identified by those strings which</td>
<td>Address</td>
</tr>
<tr>
<td></td>
<td>start with ‘http://’, and are terminated by an end of line or space</td>
<td></td>
</tr>
<tr>
<td>Payload</td>
<td>The main contents of the mail item, after removing any addresses.</td>
<td>Body</td>
</tr>
<tr>
<td>All</td>
<td>A combination of the features in all components. This mirrors a traditional</td>
<td>All</td>
</tr>
<tr>
<td></td>
<td>bag-of-words approach, which considers all words together. We also use</td>
<td></td>
</tr>
<tr>
<td></td>
<td>the results from this component as our ‘one bag’ comparison.</td>
<td></td>
</tr>
</tbody>
</table>

TF.IDF scores are then calculated for each word in the training corpus. ARFF [13]
files are then created for the training document and each testing document for a suite.
Each component of each testing document is then classified using the machine
learner, the Naïve Bayes classifier in the Weka Toolkit. We use Naïve Bayes as it has
been shown to work well on user-defined email classification (e.g. [5]). In the initial
tests, we also used information gain as a feature selection technique. However we did
not continue to use it as we achieved better results with all features.

The classifications for each component are used as evidence sources for the user’s
current activity. The evidence is resolved using each of the resolvers outlined above.

Table 2. Components of a web page

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
<th>Collapsed Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1</td>
<td>The information between ‘h1’ tags in a web page</td>
<td>Heading</td>
</tr>
<tr>
<td>Title</td>
<td>The information between ‘title’ tags in a web page</td>
<td>Heading</td>
</tr>
<tr>
<td>Address</td>
<td>The URL of the web page</td>
<td>Address</td>
</tr>
<tr>
<td>Main</td>
<td>Any information in a web page that is not contained in the above components. Note that the actual tag text is ignored (i.e. the text &lt;body&gt; and &lt;/body&gt; will not be considered), and text between script and style tags is ignored.</td>
<td>Body</td>
</tr>
<tr>
<td>All</td>
<td>As above in Table X.</td>
<td>All</td>
</tr>
</tbody>
</table>

Evaluation

In order to test the accuracy of the WeMAC model on small datasets on each of the
combinations of document types shown in Fig. 1, we performed cross validation
experiments and used a sliding window. Due to the small dataset size, cross
validation experiments were essential to improve the reliability of the results.
Although considering the temporality of emails may aid in the classification [14], we chose to ignore temporality to ensure our results were reliable.

Each static test suite (set of cross validation experiments) contained the same number of documents from each activity, and if appropriate, each document type. To ensure that equal numbers of documents from each activity were used in the training set, we left out a testing document from each activity and if appropriate, each document type for each experiment. This was to ensure the results could not be biased due to a larger number of training examples in one class.

A sliding window approach was used to ensure the results were stable across the corpus and not specific to the particular combination of documents chosen for a test suite. Each window contained at least half the documents from the previous window.

To evaluate the performance of the WeMAC model, we varied the resolver, the combination of training and testing document types, and the training set size, and compared the WeMAC accretion approach with a ‘one bag’ approach (which did not split the document into separate components).

Due to the private nature of email, there was no standard corpus which we could use to evaluate the WeMAC model. Also, the nature of our categories (activities) differs from categories used in previous experiments for both emails and web pages. Thus, we collected data specifically for this classification task.

We evaluated the performance of the WeMAC model on 5 users’ data, which is appropriate as most systems which have implemented a user-defined email classification system have tested its performance on approximately 5 users (e.g. [3, 5, 15]). Each user selected their activities from a predefined list, which was given to encourage users to classify their documents according to activity rather than a traditional classification scheme that they may already have in place. The list was created from the activities a number of research staff and students said they engaged in and which had related emails and web pages. It contained the following activities: Teaching, Subject Work, Admin, Conference, Seminar, Research and Project.

All users collected at least 10 emails and 10 web pages for each activity that they selected as applicable to them. Most users found it difficult to find more than 10 web pages per activity and commented that for the activities given, they tended to view only a small set of web pages. However, 2 of the 5 users were able to collect at least 50 emails per activity. This allowed us to compare the results obtained on small data sets with results from training on a somewhat larger number of documents. Four of the five users selected 5 activities, and 1 selected 6.

Our problem was significantly different from other machine learning problems, especially in terms of the tiny data sets that were used. Thus, we performed a careful
qualitative analysis of all results obtained to ensure that we fully understood the patterns of performance. This was done to ensure that the performance values obtained were representative of the performance of the system, and to determine whether or not the results are likely to be mirrored in a realistic setting. For more detailed results, we refer the reader to Appendix A. We give a brief overview of the important results below.

The performance value quoted is the average F1 value. We used this to capture the fine grained performance of the components on each activity. We found that the average F1 value actually maps quite closely to the accuracy value for each component. Thus, we were able to compare our results with previous systems, and use them to determine whether they results were comparable with similar existing systems which achieved accuracy values between 0.5 and 0.7 for small datasets.

The first evaluation task was performed on each user’s smaller data set, consisting of 10 emails and 10 web pages per activity. We conducted experiments using 4 documents per static test suite, and 3 windows, a total of 12 experiments. For all users, the best performance was achieved using the same document type for training and testing, as these documents shared the most textual information.

The pattern of resolver performance tended to differ across users for these experiments, as illustrated in Fig. 3. The Confidence and Majority resolvers tended to perform consistently well across all users. The performance of the Weighted resolver when training and testing on web pages generally depended on the performance of the Address component. If the address component performed consistently well across most classes, as for User 2 and User 3 whose web pages for each activity generally came from similar domains, the Weighted resolver could use this to its advantage. The Address component was weighted highly by the Weighted resolver due to its good performance, and as it consistently performed well, the Weighted resolver also performed consistently well for these users, and better than the other two resolvers.

Fig. 4. Comparing the performance of the WeMAC resolvers and the one bag approach.

However, if the Address component did not perform well, then the performance of the Weighted resolver did not tend to perform well, as the performance of the components on each activity tended to change over the corpus.

The effect of unstable component performance on the Weighted resolver was particularly evident for User 1 when training and testing on emails. For this user, the pattern of performance was so variable that weighting each component’s input to the decision based on its previous performance caused the Weighted resolver to make more incorrect predictions and perform particularly poorly.
As shown in Fig. 3, the performance of the WeMAC model and the one bag approach was quite similar for all users. This suggests that both are both appropriate methods for email classification on small datasets.

The WeMAC model performed poorly when training on one document type and testing on the other due to a lack of shared text between the different document types. Since the components performed poorly, the resolvers also performed poorly, as shown in Fig. 4. Although some resolvers tended to perform better than others for particular users, all resolvers struggled to reach an average F1 value of more than 0.4.

These problems were overcome by training and testing on both document types. Generally, with this combination of documents, a new email’s features were matched with the existing emails in the training set, and a new web page was matched with the existing web pages. Thus, the performance values were generally approximately equal to the average of the results obtained for that component when training and testing on the same document type. This is the case for both the components and resolvers, and the resolver performance can be seen in Fig. 4.

Generally, the WeMAC model and one bag approach performed almost exactly equally when training and testing on both document types. The largest difference between the performance of the best resolver and the one bag approach is approximately 0.07, which is very small.

To ensure that the WeMAC model’s quite acceptable performance on small datasets was not due to features unique to small datasets, we considered 2 additional sets of results. The first was obtained using 10 documents per activity and document type for each user over 1 window, a total of 10 sets of experiments. We compare the results obtained on this set with the above results. The second was obtained by using the larger datasets provided by User 1 and User 4. We use these to determine the WeMAC model’s performance growth as the number of training documents increases.

Across all users, the performance of all resolvers and components generally improved or remained the same as the number of documents increased. This is the expected result, as when the size of the training set increases, the number of features which are useful in the classification also increases. This is illustrated in Fig. 5, which is a representative example of the performance change, and illustrates the performance of the Confidence resolver when training and testing on emails.

For some users, the improvement in performance for a particular resolver was quite marked, as was the case for the Weighted resolver for Users 1 and 4. User 1’s results are shown in Fig. 5. The substantial performance increase was due to the improvement in performance for the Subject and Payload components. For this user,
the conversations within an activity were spread out across the data set (a conversation refers to an email part of a reply or forward chain). Thus, one set of 4 consecutive emails often did not include enough of the conversation to yield a real performance benefit. However, by considering all 10 documents, this data could be fully exploited. Since both the Payload and Subject components are affected by conversations, either by an almost identical subject line or repeated text from previous emails in the payload, both these components’ performance improved. Thus, since there was more consistency in these components’ performance across the different classes, the performance of the Weighted resolver also improved.

Generally, performance increased by approximately 0.1 as the size of the training set increased from 4 to 10. Thus, although our performance is certainly acceptable on tiny training sets with only 4 documents per activity, having the benefit of 10 certainly aids the classification.

The WeMAC model was also evaluated on a second larger set of documents. A summary of the number of documents per activity and the number of windows used for each experiment is given in Table 3.

Table 3. Details the documents used for each experiment

<table>
<thead>
<tr>
<th>Number of documents per activity</th>
<th>Number of Windows Per Experiment</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total # documents per activity: 100</td>
</tr>
<tr>
<td>User 1</td>
<td>User 4</td>
</tr>
<tr>
<td>4</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
</tr>
<tr>
<td>25</td>
<td>5</td>
</tr>
<tr>
<td>50</td>
<td>3</td>
</tr>
</tbody>
</table>

The results for these evaluations are shown in Figs 6 and 7. All resolvers clearly improve their performance as the number of training documents increases. The Majority resolver consistently outperformed the other resolvers for both users, followed by the Confidence and then Weighted resolvers. The Majority resolver outperformed the Confidence resolver for the following reason. If the component with the highest confidence made an incorrect prediction, the incorrect activity chosen by this component was chosen by the Confidence resolver. However, if a majority of components predicted the correct activity with slightly less confidence, then the correct activity was chosen by the Majority resolver. Although this situation was not overly common, it occurred enough to ensure that the Majority resolver consistently outperformed the Confidence resolver.

The Weighted resolver performed least well for User 1, and approximately equal to the Confidence resolver for User 4. For the Weighted resolver to perform well, it required a consistent pattern in performance for the components for the classes they predicted well. The components for User 1 did not exhibit this behaviour, as each component’s performance in predicting each activity changed over the document set. However, the components for User 4 did eventually tend to exhibit this behaviour, thus the Weighted resolver performed as well as the Confidence resolver.

The importance of resolver choice for different users is especially evident in Fig. 6. For User 1, the Majority and Confidence resolvers performed almost 0.2 above the Weighted resolver. However, all resolvers performed similarly for User 4’s data. This illustrates the fact that different methods of combining evidence may be more appropriate for different users.
For both users, the Majority resolver consistently outperformed the one bag approach, as shown in Fig. 7. Although the difference was only slight, it was consistent across both users had a number of different training set sizes. This supports the WeMAC model as not only a good predictor of activity on small datasets, but for the traditionally large datasets used in text classification systems as well, and compares well with existing similar systems such as [4].

Fig. 6. Rate of learning for the different resolvers on Users’ 1 and 4 data

The results on these larger data sets demonstrate that the WeMAC model can perform well not only on small training sets with similarities such as conversations, but also on large datasets over time. Thus, we have shown that the WeMAC model is a feasible general approach to heterogeneous document classification, here for emails and web pages.

Fig. 7. Comparison of the Majority resolver to the one bag approach for Users 1 and 4 data

Future Work and Conclusion

The development of the WeMAC model provides a strong foundation for predicting a user’s activity from a tiny number of training documents and heterogeneous document types. As it is one of the first systems to explore these research problems, there are many areas in which it can be extended including utilising information about the user’s current context and evaluating the model in a field setting.

As the WeMAC model was specifically architected to resolve evidence from a number of different sources, it would be interesting to determine whether contextual evidence sources such as location or time improved the performance of the system.

We have shown that the WeMAC model performs well within our testing framework. It would be valuable to determine how well the WeMAC model
performed in a realistic setting, where the activities are the user’s own, such as in the scenario outlined in the introduction, and perhaps a distracter category containing documents which don’t relate to any existing activity to make the task more natural. It would also be beneficial to evaluate the system on a larger number of users.

In conclusion, we achieved our initial goal of predicting a user’s activity accurately when only working with tiny and heterogeneous document sets. We found that by training on only 12 to 15 documents, we were able to achieve comparable performance with previous systems which used similar approaches and were evaluated on small datasets. As expected, the performance improved steadily as the training set size increased. This demonstrates that the WeMAC model is an accurate and useful model to predict a user’s activity.

References

APPENDIX A

Evaluation Task 1

Email to Email

Subject and Payload

For emails, the Subject component performed consistently well across most users, due to the presence of conversations. A conversation refers to an email which is a reply or forward of a previous email, where the subject line of the mails are almost identical, with a prefix of either ‘FW’ or ‘RE’. Within the small sets, there were usually at least 2 emails belonging to the same conversation. Thus, the subject component was able to perform well. For the same reason, the Payload component performed well, as emails in the same conversation usually have the text from the previous mails repeated. The performance for these components across the five users is shown in Fig. 1. This figure also shows that the performance of the Payload component was slightly less than the Subject component. Although the payload of an email in a conversation contains some common text with other emails in the same conversation, there is usually more text in addition, whereas the subject line is virtually identical. Thus, the Subject component generally performs better than the Payload component as there is more common text.

Fig. 1. The performance of the Subject and Payload components across the 5 users.

To and From

The performance of the To and From components varied across users, as shown in Fig. 2. The performance of these components depended on the spread of To and From addresses. If the From or To addresses for the emails were unique, this meant the WeMAC model was trying to classify based on evidence it hadn’t previously seen. If the emails came from or were sent to the same address across many activities (as was the case with User 2), the WeMAC model was not able to distinguish the current activities from the others. In these cases, the component performed poorly. The To component performed well when the emails were sent to mailing lists which differed between activities as was the case with User 2 (where the To component outperformed all other components), and the From component performed well when the emails for a particular activity were generally sent from one person whose emails were not classified into other activities.

Fig. 2. Performance of the To and From components for training and testing on emails.

Address
The Address component performed consistently poorly for all users, as shown in Fig. 3. Most emails did not contain a large number of URLs. Generally, if a user’s emails did contain URLs this happened only with one or two activities, for example Subject Work and Meeting for User 1. Often, the URLs within emails of the same activity were similar. However, there were not enough URLs to make the Address component an accurate predictor. The WeMAC model performed slightly better on User 1’s data due to the larger number of URLs within his emails, however, the component still performed relatively poorly.

![Performance of the Address Component](image)

**Fig. 3.** Performance of the Address component when training and testing on emails.

### Web Page to Web Page

**Address and Title**

When training and testing on web pages, the component which tended to perform best was Address. For the users whose URLs came from the same or similar domains, the URLs contained common text. Thus, the learner was able to accurately predict the activity from this information. For the same reason, the Title component tended to perform with a similar pattern to the Address component, as web pages from the same domain tended to have similar titles. However, as for the Subject and Payload components of the emails, the Title tended to perform worse than the Address as although there was common text, there was more text that was not common for the Title than the Address component of the web page. The performance of these two components is shown in Fig. 4.

![Comparing Title and Address](image)

**Fig. 4.** Performance of the Title and Address components for training and testing on web pages.

### Main

The performance of the Main component also tended to vary across users, as shown in Fig. 5. For some users, the web pages which were in the same address space contained very similar text on the pages themselves. An extreme example of this is User 1, whose pages for Seminar all pointed to different parts of the same page, thus the text on these pages was identical (this did not happen in other activities, however). The content of the Teaching web pages for User 2 tended to be quite similar, as they referred to tutorial exercises from week to week. Main performed poorly for User 3 as there was a fair amount of content overlapping between many of the activities, especially Seminars, Research and Conference, which all tended to relate to the user’s research areas. Thus, all the pages tended to be classified into one of these categories due to their similar key words. Generally, however, the Main component performed well.
Training and Testing on Different Document Types

We found that the WeMAC model performed poorly when training and testing on opposite document types. This was due to a lack of shared textual information between the components of the documents of different types belonging to the same activity. Generally, the performance values were close to random (0.2, as there were 5 or 6 activities for each user), as shown in Figs. 6 and 7.

There were a small number of components for particular users which achieved better than random performance. For example, Heading for User 2 achieved an average F value of nearly 0.4 when training on emails and testing on web pages, as shown in Fig. 6. This was due to a small number of common words in the subject line and title of Subject Work. Also, some components demonstrated different performance patterns on the different document types. For example, the Address component for training on emails and testing on web pages achieved an average F1 value of approximately 0.3 for User 2, but when training on web pages and testing on emails, the Address component achieved only slightly above 0.1. In the situations of good performance, some of the features which were indicative of a class for one type of document were present in the other type, thus the component performed better than randomly. If the performance was poor, the features which were indicative of a class for the training document type tended not to be present in the test document type.

Due to the poor component performance for these combinations of document types, the resolvers performed poorly, as shown in Figs. 8 and 9. The Weighted resolver outperformed the Majority and Confidence resolvers for some users, for example User 4. This typically occurred when one compo-
nent, in this case Address, performed relatively consistently in its predictions off each class. Thus, the Weighted resolver could exploit this pattern to its advantage. Generally, however, the 3 resolvers perform poorly when training and testing on different document types. Those resolvers that perform slightly better still struggle to reach an average F1 value of 0.4.

**WeMAC Resolvers and One Bag Approach**

![Performance of the WeMAC resolvers and the one bag approach for training on emails and testing on web pages.](image)

We found that when web pages were involved in the classification, as was the case for using different document types for training and testing, the performance of the one bag approach was dominated by the performance of the Main component of the web page. The Main component was collapsed into the Body component for these experiments. Thus, the one bag approach fared well only when Body was a good performer, which it generally wasn’t. Thus, here, the WeMAC model either outperformed or was approximately equal in performance to the one bag approach.

These problems were overcome, however, by training and testing on both document types. Generally, with this combination of documents, a new email’s features were matched with the existing emails in the training set, and a new web page was matched with the existing web pages. Thus, the performance value of a component was generally approximately equal to the average of the results obtained for that component when training and testing on the same document type. This is the case for both the components and resolvers, and can be seen in Figs 10 and 11.

**Component Performance**

![Component performance for training and testing on both types of document.](image)

This is easily seen here with User 3, whose data yielded poorer results than all other users. Although the model performed quite well on User 3’s emails, the performance on web pages was quite
poor. This explains the poorer performance of the model for User 3 compared with the other users, for whom the model tended to perform relatively well on both emails and web pages.

Fig. 11. Performance of WeMAC model resolvers and one bag approach when training and testing on both document types.

Evaluation Task 2

Performance change from 4 – 10 documents per activity

For a very small proportion of the combinations of documents, performance decreased slightly as the size of the training set increased. This typically occurred when training and testing on different document types, as a result of a greater spread of features within each type, but no increase in the overlap between them. Thus, performance of these combinations often suffered as a result.

For other combinations of documents, if performance declined as the training set increased in size, it was typically only minimal. For example, the average F1 value for the Majority resolver for User 2 decreased by approximately 0.07 when training and testing on both types of documents as the training set size increased, as shown in Fig. 12. This was due to the clumping of conversations in this user’s data. Usually, the conversations formed a set of consecutive emails. Thus, the small sets performed very accurately as the subject lines were almost identical. As the size of dataset increased, the clumps were not as helpful in the classification since all 10 documents were considered, thus there was more spread which led to poorer performance overall. However, this performance decrease was only slight.

Fig. 12. Illustrating the slight decrease in performance of the Majority resolver when considering combinations involving emails, User 2’s data.