Results

Analysis of typical user-based usability testing

Typical usability testing was user-based performance that was evaluated based on total time to complete tasks, number of errors, frequency of assists, and the percentage of completed tasks. Table1 shows the result of total time to complete task. The first column shows each participant and the first row shows the each website. Red texts show shortest time that participants had to finish all tasks from the websites.

	1	2	3	4	5	6	7	8	9	10
SM5	8.15	9.26	12.27	11.58	5.54	11.25	11.59	14.24	16.11	13.48
Tosca	6.54	7.37	7.44	8.05	5.54	8.34	8.35	7.13	16.08	12.19
Loche	**4.29**	**5.31**	**5.23**	**5.02**	**4.09**	6.25	**3.60**	**5.30**	**9.00**	**9.07**
Sonata	6.25	5.58	6.15	10.35	4.28	**6.00**	5.19	6.56	9.21	9.39

Table 1 Total time to complete tasks

	1	2	3	4	5	6	7	8	9	10
SM5	2	**0**	4	3	**0**	3	13	**1**	**2**	1
Tosca	2	4	3	**0**	5	8	5	3	5	**0**
Loche	**1**	3	2	**0**	3	**0**	**2**	3	3	5
Sonata	6	3	**1**	2	**0**	**0**	7	2	4	6

Table 2 Number of errors (Red texts show the lowest error)

	1	2	3	4	5	6	7	8	9	10
SM5	1	**0**	1	1	**0**	1	1	**0**	1	**0**
Tosca	1	1	1	**0**	**0**	1	1	**0**	**0**	**0**
Loche	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Sonata	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Table 3 Frequency of assists (Red texts show the lowest assist)

	1	2	3	4	5	6	7	8	9	10
SM5	100	100	100	100	100	100	100	100	100	**67**
Tosca	100	100	100	100	100	100	**67**	100	100	**67**
Loche	100	100	100	100	**67**	**67**	**67**	100	100	100
Sonata	100	100	100	**67**	100	100	100	**67**	100	100

Table 4 Percentage of completed tasks (Red texts show the less than 100%)

After analyzing the user-based usability testing results, we concluded the participant's preference of web interface design (Table5).

1	2	3	4	5	6	7	8	9	10
Loche	Loche & SM5	Loche & Sonata	Loche	Sonata & SM5	Sonata	Loche & Sonata	Loche	Loche	Loche & SM5

Table 5 Estimated participant's preference of web interface design from user-based usability testing

Lastly, Table6 shows the results from the user survey that participants directly answered the one, which they liked most among the four web interface designs. The results of Table 5 and Table 6 are mostly similar with few differences; therefore the user-based usability testing was successful.

1	2	3	4	5	6	7	8	9	10
Loche	SM5	Loche	Loche	Sonata	Sonata	Loche	Loche	Loche	Tosca

Table 6 Participant's preference of web interface design from user survey

Analysis of usability test using bio-signals (EEG, ECG)

First, we analyzed the ECG electric waves of SDNN, the standard deviation of the normal RR intervals, and the ratio of high frequency (HF) and low frequency (LF) to check emotional status. The higher number of SDNN means the participant's emotion is more active and positive. The lower number means there is no emotional feeling change. The higher number of HF means the participant's heartbeat is stronger which shows more active and positive feeling. Therefore, the smaller ratio of LF divided by HF is their preference website.

	A	B	C	D	E	F	G
1	date	subject	task	SDNN	LF	HF	LF/HF
32	060 250	6. □□□	Open	44.8	75.6	24.4	
33			SM5	40.4	78	22	3.545455
34		X	Tosca	55.1	80.1	19.9	4.025126
35			Loche	47.3	81.9	18.1	4.524862
36			Sonata	60	81.1	18.9	4.291005
37							
38	060 260	7. □□□	Open	53.2	77	23	
39			SM5	41.3	50.4	49.6	1.016129
40		X	Tosca	48.1	62.7	37.3	1.680965
41			Loche	31.7	74.8	25.2	2.968254
42			Sonata	33.9	79.4	20.6	3.854369
43							
44	060 260	8. □□□	Open	52.9	40.5	59.5	
45			SM5	74.6	37.1	62.9	0.589825
46			Tosca	64.8	38.7	61.3	0.631321
47			Loche	79.8	51.1	48.9	1.04499
48		X	Sonata	59.5	43.3	56.7	0.763668
49							
50	060 260	9. □□□	Open	52.1	74.5	25.5	
51			SM5	39.7	56.9	43.1	1.320186
52		X	Tosca	41.1	75.1	24.9	3.016064
53			Loche	60.8	73.8	26.2	2.816794
54			Sonata	58.6	72.7	27.3	2.663004
55							
56	060 290	10.□□□	Open	66.3	84.4	15.6	
57			SM5	64.1	69.1	30.9	2.236246
58			Tosca	58.1	78.1	21.9	3.56621

Figure 3 Example of analyzing ECG data

	1	2	3	4	5	6	7	8	9	10
SDNN	Tosca	SM5	Sonata	Loche	Sonata	Sonata	Loche	Loche	Loche	SM5
LF/HF	Sonata	Loche	SM5	SM5	Loche	SM5	SM5	SM5	SM5	SM5

Table 7 Estimated participant's preference of web interface design from ECG

Second, we analyzed the EEG data, which a frequency of 4~30 Hz was selected to remove the noise from raw data, and the values of RPS (relative power spectrum) were found in the frequency band of brain waves, theta (4~8 Hz) and beta (13~30 Hz), using FFT (Fast Fourier Transform). As we mentioned previously, brain waves were measured by QEEG-4 (LAXTHA) and the position of electrodes went by the international 10-20 electrode arrangement, using the channels of F3 and F4 for emotional control areas.

Higher numbers of theta mean that people are a in comfortable and positive status, while higher numbers of beta mean that people are a in nervous and anxious thinking status. Thus, the F3 position is located in the left side of the brain, an area that processes optimistic thinking, and the F4 position is located in the right side of the brain, an area that processes pessimistic thinking. Therefore, we need to focus more on analyzing the F3 area for decision of preference of web interface design.

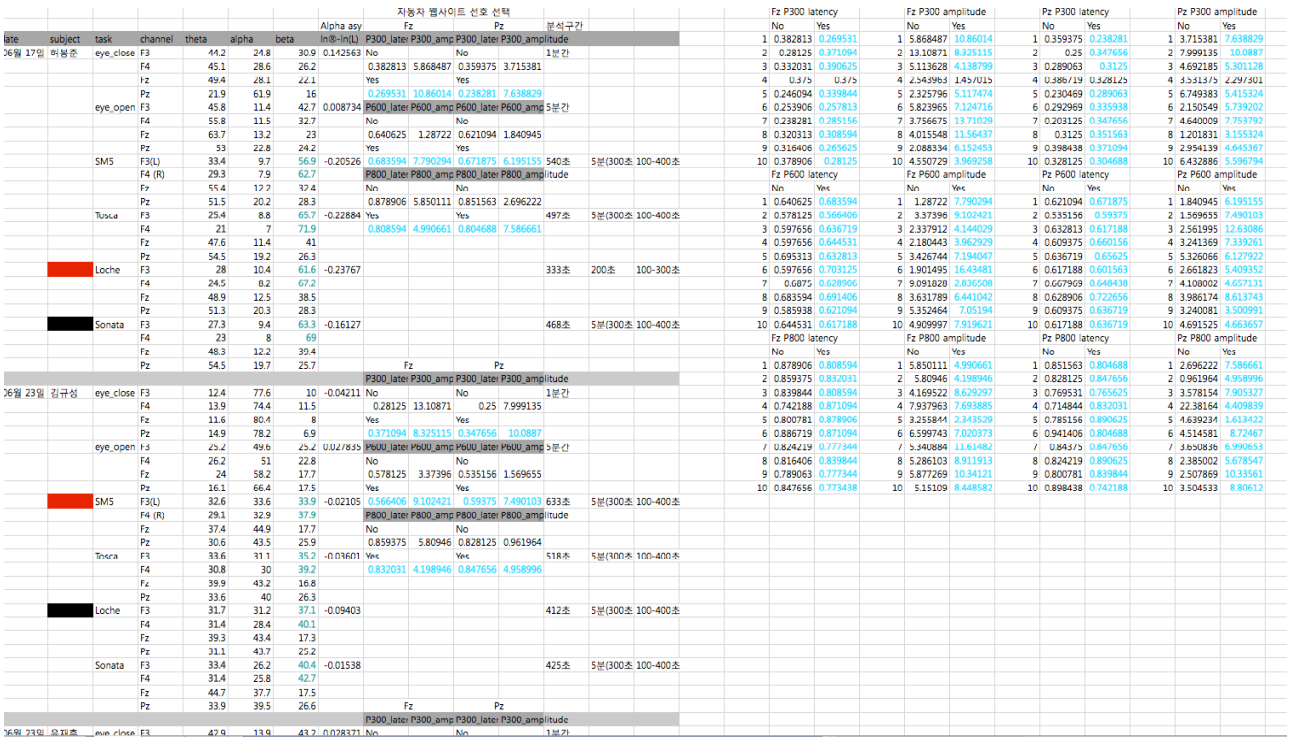

Figure 4 Example of analyzing EEG data

Figure 5 shows the result of beta power of F3 and F4. The red squares are the preference of web interface designs that participants chose from the user survey, and the black squares are the ones that they do not like. According to the figure of beta power of F3, most of their preference websites of beta power are lower than non-preference websites. As we mentioned previously, beta power shows the meaning of nervous and anxious thinking status, so this result is what we anticipated.

ever, the disjointness of the labels is no longer valid, in the sense that a single music sound may be classified into multiple emotional categories. This stipulation seems to make the problem significantly more complicated. Unfortunately, the area is yet to be explored. The sparse literature on this subject is primarily geared toward text classification and, to our knowledge, no prior work exists in the music information retrieval domain.

We resort to the scarcity of literature in multi-label classification by decomposing the problem into a set of binary classification problems. In this approach, for each binary problem a classifier is developed using the projection of the training data to the binary problem. To determine labels of a test data, the binary classifiers thus develop are run individually on the data and every label for which the output of the classifier exceeds a predetermined threshold is selected as a label of the data. To build classifiers we used Support Vector Machines (SVM for short) and our implementation is based on the LIBSVM [1].

The SVMs were trained using features extracted from the sounds. To extract features we used MARSYAS [Tzanetakis and Cook,2000]. The extracted features are divided into three different categories: timbral texture features, rhythmic content features, and pitch content features. The dimension of the final feature vector is 30.

The accuracy of the classifiers is measured using *precision*, *recall*, *break-even point* and *F1-measure*. Since the precision and the recall can be averaged over the classifiers with or without weighting, we use both *micro-averaged precision* P_{micro}, *micro-averaged recall* R_{micro}, *macro-averaged precision* P_{macro} and the *macro-averaged recall* R_{macro}. In addition, we also compute the Hamming accuracy (denoted by HA), which is defined to be the simple unweighted accuracy, that is the unweighted ratio of the total correct to the total input size. These are the performance measures that are widely used in information retrieval literature [Yang and Liu,1999].

4 Experiments

Our SVM-based multi-label classification method was tested for two problems: classification into the thirteen adjective groups and classification into the six supergroups. There was significant difference in the distribution of the positive data for some of the adjective groups (e.g., "bluesy" not appearing in the classical category). We constructed the supergroup classifier for each of the four styles. Due to the space limitation, we only include the results of all the thirteen adjective groups on all four styles. We divided the 499 sounds into training data and testing data by a random $50\% - 50\%$ split.

The accuracy measures on each of the thirteen classes are shown in Table 2. The overall accuracy for the two experiments are shown in Table 3. The breakeven point, i.e. the half-way point between the precision and the recall, was 46% in micro-avering and 43% in macro-avering. In our six-supergroup experiment the breakeven point was 50% in micro-avering and 49% in macro-averaging, so the overall accuracy was improved when the number of categories is reduced.

The overall low performance can be attributed to the fact that there were numerous borderline cases for which the labeler found it difficult to make decision. Also, the frequency of the labels was not equal across music types. We actually carried out another set of experiments, emotion detection for supergroups within each music type. We observed improvements especially, Performance stood out on supergroup 2 for classical and supergroup 4 for fusion. This may suggest that the use of genre information might improve emotion detection.

Our experiments show that emotion detection is a rather difficult problem and improvement of performance is the immediate issue. This can be resolved by: expanding the sound data sets, collecting labeling in multiple rounds to ensure confidence in labeling, using different sets of adjectives, incorporating style and genre information, and using different types of features.

[1] Available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

Group	TP	TN	FP	FN	P	R	HA
A	12	132	81	22	0.1290	0.3529	0.5830
B	3	189	44	11	0.0638	0.2143	0.7773
C	96	70	45	36	0.6809	0.7273	0.6721
D	53	106	64	24	0.4530	0.6883	0.6437
E	46	81	61	59	0.4299	0.4381	0.5142
F	43	102	73	29	0.3707	0.5972	0.5870
G	26	127	78	16	0.2500	0.6190	0.6194
H	28	156	40	23	0.4118	0.5490	0.7449
I	56	135	41	15	0.5773	0.7887	0.7733
J	10	178	47	12	0.1754	0.4545	0.7611
K	15	161	51	20	0.2273	0.4286	0.7126
L	13	144	70	20	0.1566	0.3939	0.6356
M	18	181	43	5	0.2951	0.7826	0.8057

Table 2: Accuracy measures on adjective group classification.

Measure	P_{micro}	R_{micro}	B_{micro}	F_{micro}
Values	0.3621	0.5893	0.4757	0.4486
Measure	P_{macro}	R_{macro}	B_{macro}	F_{macro}
Values	0.3247	0.5411	0.4329	0.4058

Table 3: Overall accuracy measures.

Acknowledgments

The authors thank Diane Cass for helping us in finding references. This work is supported in part by NSF grants EIA-0080124, DUE-9980943, and EIA-0205061, and in part by NIH grants RO1-AG18231 (5-25589) and P30-AG18254.

References

[Farnsworth,1958] Paul R. Farnsworth. *The social psychology of music*. The Dryden Press, 1958.

[Hevner,1936] Kate Hevner. Experimental studies of the elements of expression in music. *American Journal of Psychology*, 48:246–268, 1936.

[Huron,2000] D. Huron. Perceptual and cognitive applications in music information retrieval. In *International Symposium on Music Information Retrieval*,2000.

[Tzanetakis and Cook,2000] George Tzanetakis and Perry Cook. Marsyas: A framework for audio analysis. *Organized Sound*, 4(3):169–175, 2000.

[Yang and Liu,1999] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR*, 1999.

using it to open your new trees. In your new "true" trees, notice the .1 or .2 following the sample number, these designate the population of origin of each sample. How many of your ~20 simulated trees show reciprocal monophyly of the two populations? How many show paraphyly of one population (with the other population monophyletic)? How many show polyphyly? Enter your results in the third column (I did 30 simulations):

Topology	Div time = N AJC's results	Div time = N Your results	Div time = 3N AJC's results	Div time = 3N Your results
Reciprocal monophyly	**7**		**23**	
Paraphyly of 1 population	**14**		**6**	
Polyphyly	**9**		**1**	

Now look at the trees inferred from the mutational history. Can you determine the proportion of monophyletic, paraphyletic and polyphyletic trees? Are these results different from the "true" genealogy file? Why or why not?

Now copy your infile to the other folder (e.g., "2pop_old_OUTFILES"), and save under a new name. Edit the age of the historical event (population splitting event in forward time, coalescent event in backwards time) from 10,000 to 30,000 (3***N***) generations. Review your resulting genealogies. Tabulate the frequencies of monophyletic, paraphyletic and polyphyletic genealogies. Can you explain WHY you (probably) observed more monophyletic trees when the splitting even happened longer ago in the past? If you have any non-reciprocally monophyletic trees, look at the ancestral lineages. Do you have >2 lineages extending close to the root of the tree? Is there any reason why you might expect more ancestral lineages extending farther back in the tree for those trees that are **not** reciprocally monophyletic?

IV. Migration vs. lineage sorting

Lack of monophyly between two sister populations could be due to incomplete lineage sorting, or it might also be due to migration. In this example, we will simulate 2 sister populations that diverged 5***N*** generations in the past. In the absence of migration, these populations are likely to be reciprocally monophyletic. However, in this simulation we will add a recent dispersal event from one population to the other at time = (***N***/10) generations ago. During the dispersal event, each member of one population has a 0.2 probability of migrating to the other population, but population sizes will remain the same. Modeling migration with population sizes that remain constant is referred to as "conservative migration" (e.g., Nagylaki 1998).

Create a new directory, labeled e.g., "2pop_MIG_OUTFILES"), and copy the executable plus the infile (".par" file) into it. Leave most of your simulation parameters the same, but change the historical events as follows:

```
//historical event: time, source, sink, migrants, new deme size, new growth rate, migration
matrix index
2 historical event
1000 0 1 0.2 1 0 0
50000 0 1 1 1 0 0
```

and PDF text do not always match exactly (e.g., em dash in PDF vs. a hyphen in XML), we use dynamic programming to find the substring in the PDF text with smallest Levenshtein distance to the caption text in the XML file. We modify the standard Wagner-Fischer dynamic programming algorithm for edit distance [25] by setting the cost for starting (and ending) at any position in the PDF text to 0. This modification maintains the time complexity of $O(mn)$, where m and n are the string lengths.

Labeling figures:

Once we have identified the page that a figure is on, we render that page as an image and then use multi-scale template matching [2] to find the position of the figure on the page. We use the figure image as a filter and cross-correlate it with the page to produce a spatial map of image correlations. Template matching is typically done using image representations such as edge detections or oriented gradients [2], but because we do not have to deal with typical conditions present in natural images such as variations in lighting or pose, we find that template matching in raw pixel space works best. We use OpenCV's `matchTemplate` implementation with the similarity metric `CV_TM_CCOEFF_NORMED`, matching at 45 scales where the figure's largest dimension relative to the page takes up between 10% and 95% of the page.

In rare cases, the provided figure images do not match the figures as they appear in the PDF (e.g., subfigures may be laid out horizontally on the PDF but vertically in the provided image file). If template matching yields a similarity below 0.8 for any figure, we exclude the paper from our dataset to reduce the risk of inaccurate training data.

Labeling tables:

Tables are sometimes provided as images in the same way figures are. However, it is more common for tables to be represented directly in the XML with tags for each table cell. We first tried using the textual edit distance to identify table coordinates in the PDF file (similar to captions), but we found that the order of table cells often differs between PDFBox's extracted text and the XML (e.g. table cells may be extracted from the PDF in column-major order while the XML is row-major). Therefore, we instead use a bag of words similarity.

We find the token sequence in the PDF that has the highest similarity to the set of words in the XML table. We can find the optimal sequence in the PDF text efficiently by maintaining a word difference counter. For a given start position in the PDF stream, we initialize the counter to the bag of words from the XML table. For each token following this position, we decrement the counter for the word at the current position (while allowing negative counts to represent words that occur more in the PDF than in the XML). This procedure is repeated for each start position on the PDF page. The following pseudo-code illustrates our algorithm for finding the interval on the page with the lowest bag-of-words distance to the table:

```
best_dist <- math.inf
for start_word in page_words:
  diff_counter <- table_words
  cur_dist = sum(table_words)
  for end_word in page_words from start_word:
    diff_counter[end_word] -= 1
    if diff_counter[end_word] >= 0:
      cur_dist -= 1
    else:
      cur_dist += 1
    if cur_dist < best_dist:
      best_dist <- cur_dist
      store start_word and end_word positions
```

If m is the length of the XML table and n is the length of the PDF, generating or copying the initial word counter is $O(m)$ and iterating over ending words on the PDF text is $O(n)$. Both of these occur for every starting word, for a total time complexity of $O(n(n+m))$. We identify the minimum axis-aligned bounding box containing all caption tokens as the table caption region.

Dataset	Manually-labeled		Induced labels	
name	CS-Large [7]	PubMed	LaTeX	XML
# papers	346	104	242,041	791,381
# figures	952	289	1,030,671	3,064,951
# tables	282	124	164,356	1,267,464

Table 1: Number of papers, figures, and tables in the manually-labeled datasets (left) and our datasets of induced labels (right).

3.3 Comparison to Manual Annotation

In this section, we proposed a method for automatically inducing labeled data for figure extraction in scientific documents. An alternative approach is to train annotators to sift through a large number of research papers and label figure and table coordinates and their captions. While this approach typically results in high quality annotations, it is often impractical. Manual annotation is slow and expensive, and it is hard to find annotators with appropriate training or domain knowledge. With limited time and budget, the size of labeled data we can collect with this approach is modest.[9]

Scalability of induced labels:

In contrast to manual annotation, our proposed method for inducing labels is both scalable and accurate. We compare the size of our datasets with induced labels to that of manually labeled datasets in Table 1. We compare with two manually labeled datasets:

- The "CS-Large" dataset [7]: To our knowledge, this was previously the largest dataset for the task of figure extraction. Papers in this dataset were randomly sampled from computer science papers published after the year 1999 with nine citations or more.

[9] Another alternative is to use crowdsourced workers (e.g., using Amazon Mechanical Turk https://www.mturk.com/ or CrowdFlower http://www.crowdflower.com/) to do the annotation. Although crowdsourcing has been successfully used to construct useful image datasets such as ImageNet [11], [20] found that crowdsourcing figure annotations in research papers yielded low inter-annotator agreement and significant noise due to workers' lack of familiarity with scholarly documents.