NTCIR-18: Proceedings of the 18th NTCIR Conference on Evaluation ofInformation Access Technologies, June 10-13, 2025, Tokyo, Japan
DOI: https://doi.org/10.20736/0002002090

Biting into SUSHI: The University of Maryland at NTCIR-18

Douglas W. Oard
Shashank Bhardwaj
University of Maryland, College Park, MD, USA
(oard,sbhardw7)@umd.edu

Emi Ishita
Kyushu University
Fukuoka, Japan
ishita.emi.982@m.kyushu-u.ac.jp

Abstract

The University of Maryland participated in both subtasks of the SUSHI Pilot Task. This paper describes the design of the systems used for each task, and it presents some preliminary analysis of the available results. The generation of data that has been shared with other participating teams is also described.

Keywords

Archival search, Inference, Text classification

Team Name

UMCP

Subtasks

A (Folder Ranking), B (only Archival Reference Detection)

1 Introduction

The University of Maryland team participated in both the Folder Ranking task (Subtask A) and the Archival Reference Detection task of Subtask B. We did not participate in the Archival Reference Boundary Detection task of Subtask B. Our team included two organizers of the SUSHI Task, plus one member (the second author of this paper) who was not a task organizer but who contributed to the development of the Subtask A relevance assessment system. Results from this team should therefore be caveated with an understanding that the team members had an unusual degree of insight into the design details of the task, some of which may not have been known as early or in as much detail by other SUSHI participants.

2 Subtask A: Folder Ranking

In our early work on box ranking [5] we had developed quite a simple strategy—use the query to first rank the training documents, then replace each Document ID with the Box ID for the box in which that document was found, then remove duplicate Box ID's, working down from the top of the ranked list. This works because documents in a box are to some extent similar to each other, so finding a good training document in a box suggests that there may be other good documents in that same box. A similar intuition can be applied to folders, so that approach (with Folder ID's substituted for Document ID's) became our first baseline for folder ranking. We could represent documents using title metadata, OCR text, or folder labels. The OCR text has many character errors, but Tokinori Suzuki from the Kyushu University team created a summary for each document using GPT4o and shared those with us, so those became a fourth possible document representation. In our run names, we abbreviate these fields T, O, F, and S, respectively. When searching using a single field we used BM25; when using multiple fields we used BM25F (in both cases, using PyTerrier[1] [4]). We included no special handling for withdrawal sheets; they were indexed in the same way as any other document. All documents have title metadata; eight documents have no OCR text, and thus no summary.[2]

We used the document title metadata in exactly the form it appears in the metadata file, and the summaries in exactly the form that we obtained them. For OCR text, we use the OCR text only from the first page (because in our prior work we had found that (with default BM25 parameters) variations in page count hurt more than the additional evidence helped [5]). For folder labels, we performed Subject-Numeric Code (SNC) translation using the translation table provided by the organizers, using the 1965 translation when that was available and otherwise the 1963 translation. When a scope note was present, we truncated that note at the first instance of any term from a list of terms that generally indicated the end of the descriptive part of the scope note.[3] The format of the NARA SNC codes is quite consistent, but Brown SNC codes appear in a wide range of formats; we wrote some rather brittle code that successfully handled most of those variations. However, we did not perform SNC translation for Brown folder labels that contained 20 or more characters because we found that relatively long Brown folder labels were essentially already translated. We tuned the parameter 20 (and all of our other parameters) to optimize nDCG@5 on the Dry Run test collection.

The simple expedient of ranking the training documents can find only folders that contain training documents, which in both the Dry Run and the final official test collections are just under half the folders in the test collection. To rank other folders, we performed what we call "expansion," in which for each other folder (i.e., each folder with no training documents) we first selected a set of training documents we expected would help to predict what the score for that folder would have been, and then we computed a score for the "expansion" folder based on the scores of that selected set of training documents. We implemented four ways of selecting training documents, which we called CloseDate, SameSNC, SimilarSNC, and SameBox. We defined SameSNC as any training document in any folder that had the same SNC in its folder label, regardless of what box that folder was in. For SimilarSNC we modified our SameSNC test to generate any match of first and second level for POL codes (i.e., POL 3-2 would match POL 3, and it would match POL 3-1) or any match at first level for other codes (e.g., SCI 7 would mach SCI 2 or SCI). For SameBox all we required was that the training documents be from the same box, regardless of folder label.

[1]https://github.com/terrier-org/pyterrier
[2]S32480, S32974, S35646, S37271, S37758, S37933, S37966, S40974
[3]These case sensitive terms were: 'SEE', 'for which ', 'Exclude ', 'exclude ', 'Subdivide ', 'subdivide ', 'Other than ', 'other than ', 'For ', 'Except ', 'except '.

For CloseDate we required SameSNC and that the training document's date be within the expansion folder's date range. To support this, we calculated an end date for every folder by first sorting the start dates for all folders that shared an SNC and then defining the end date of one folder to be the start date of the next folder in that sequence. When there was no subsequent folder, or when two or more folders in the sequence had the same start date, we assigned no end date. Folders without end dates (which are actually the majority of all folders) are excluded from CloseDate matches, so CloseDate is very highly selective. It turned out that the greatest challenge for checking CloseDate was that document dates in the collection exhibit far greater variability than we had first expected. Ultimately we were able to process the vast majority of those dates correctly, although with rather brittle (and slow) code. Because we expected our date processing to be useful to others, we shared JSON objects with other participating teams for the full set of documents in the collection and for the full set of folders in the collection. The document JSON also includes the OCR text for every page, Tokinori Suzuki's summaries, and our translated folder labels.[4]

Once we had a set of training documents for an expansion folder, we first averaged the BM25 or BM25F scores for those documents. Often there was only a single training document in the set, and in such cases this average was simply the score of that one training document. This tended to produce overly optimistic scores (ranking expansion folders as highly as a folder that had actually contained a training document), so we then subtracted a score adjustment factor. We set this adjustment factor to be the minimum value that would place the highest scoring expansion folder for a query at rank 2 (i.e., placing one folder that actually contained a training document ahead of it). As with all our parameters, we selected 2 by optimizing nDCG@5 on the Dry Run collection.

BM25 is limited to exact match on term stems, so we also ran ColBERT (Contextualized Late Interaction over BERT) in order to take advantage of more nuanced matching using term embeddings. To do this, we inserted separator tokens between the title, translated folder label, and first-page OCR text (with OCR text trimmed back from the end to fit all of that in 512 tokens). We ranked the folders in the training set using ColBERTv2 [7],[5] with no additional collection-specific pre-training or task-specific fine-tuning, using the same process for ranking folders in the training set that we used for BM25F. To rank folders that are not in the training set, the minimum score from folders in the training set that share either the same box or the same SNC code was used. This differs from the expansion process that we used in our BM25F runs in two ways: (1) it was based on folder scores rather than document scores, and (2) it was based on SameBox or SameSNC (rather than the SameBox and SameSNC we used for BM25F expansion). This resulted in our UMCP-COLBERT-* runs.[6]

Because we expected ColBERT to find things that BM25F had missed, and vice versa, we also implemented system combination using Reciprocal Rank Fusion (RRF) [1]. When using RRF, we performed expansion first, and then we applied RRF on the ranked list that expansion had produced. Cormack's RRF implementation includes a parameter $r = 60$ that controls the emphasis given to items lower in the list (higher r gives relatively more emphasis to lower items). Because our nDCG@5 measure is focused on only the top five folders, we set $r = 0$. We also found that it was useful to prefer the BM25F ranking (after expansion), so we gave ColBERT 65% of the weight of the BM25F score. Again, both parameters were tuned on the Dry Run collection.

The folder labels indexed by BM25F included only folders that contained training documents, but folder labels are available for every folder. To allow all folder labels to influence our results, we performed full-collection BM25 search of only the translated folder labels, and then combined that with our initial RRF result, this time giving folder-labels 38% of the weight of the first RRF result (again, tuned on the Dry Run collection). This two-stage RRF process is a bit of a hack, but this was our last enhancement, and it was easier to do it this way than to rework our code base and then do the necessary regression testing. We indicate the use of RRF by '-CF' in the run name, indicating that RRF was run first with ColBERT and then with full-collection folder label search results.

2.1 Formative Evaluation

We focused our formative evaluation principally on Folder nDCG@5, using the Dry Run collection. The Dry Run collection has a couple of peculiarities that limited its fidelity for this purpose. First, 155 of the 200 topics have only one relevant document. If the Dry Run test collection had a very large number of topics, that might not have been a limiting factor. But with only 200 topics, this posed some risk of our tuning to noise rather than to signal. Second, the queries in the Dry Run collection had been created by selecting document titles, and we had seen in our earlier work that searching document titles did far better than searching OCR [5]. We were concerned that this might be a collection artifact, and thus skeptical about the effect of tuning in favor of document titles.

Nonetheless, the Dry Run test collection was the best we had, so we set aside those concerns and simply tuned every parameter using that collection. In addition to the parameters noted above, this included 8 BM25F parameters (called 'c' and 'w' by Terrier) that controlled field weights and length normalization for the four BM25F fields. This tuning process ultimately led to our UMCP-*-TOFS-L2-CF run. Surprisingly, that run did actually turn out to be the best of the 25 runs that we submitted when scored by nDCG@5 on the final official test collection, although we don't yet know whether different tuning might have done even better.

During our tuning process we saw that SameSNC expansion and SameBox expansion each yielded some improvement (by occasionally highly ranking folders that could otherwise not have been found), but that neither SimilarSNC nor CloseDate was helpful (presumably because Similar SNC let in too much noise, and CloseDate very rarely actually did any expansion). This led us to implement a conjunctive combination of SameSNC and SameBox, performing expansion only when the SNC was the same and the training document was in the same box as the expansion folder. This looked good on the Dry Run test collection, so this is what we used in the runs that contain '2' in their name (indicating SameSNC+SameBox expansion with a maximum rank of 2).

We were a bit skeptical about whether our BM25F parameter tuning might be unhelpfully overtuned to the peculiarities of the

[4]Available on the SUSHI website: https://sites.google.com/view/ntcir-sushi-task/
[5]https://github.com/stanford-futuredata/ColBERT
[6]Unlike our other submissions, there is also a UMCP-ColBERT-D run.

Table 1: Subtask A folder ranking results, 95% conf intervals.

	Run	**nDCG@5**	±	MAP	±	MRR	±	S@1	±
RRF	UMCP-T-TOFS-L2-CF	0.226	0.08	0.150	0.06	0.455	0.13	0.378	0.14
	UMCP-TD-TOFS-L2-CF	0.228	0.08	0.151	0.06	0.434	0.13	0.333	0.14
	UMCP-TDN-TOFS-L2-CF	0.229	0.08	0.165	0.06	0.490	0.13	0.400	0.14
Expansion	UMCP-ColBERT-T	0.185	0.08	0.113	0.06	0.377	0.13	0.311	0.14
	UMCP-ColBERT-TD	0.183	0.08	0.116	0.06	0.347	0.11	0.222	0.12
	UMCP-ColBERT-TDN	0.173	0.08	0.103	0.06	0.313	0.12	0.244	0.13
	UMCP-ColBERT-D	0.163	0.07	0.096	0.05	0.327	0.12	0.267	0.13
	UMCP-T-TOFS-L2	0.211	0.08	0.128	0.06	0.416	0.14	0.356	0.14
	UMCP-TD-TOFS-L2	0.213	0.08	0.129	0.06	0.412	0.13	0.333	0.14
	UMCP-TDN-TOFS-L2	0.214	0.08	0.143	0.06	0.462	0.14	0.400	0.14
Single-Field	UMCP-T-O-B	0.191	0.08	0.119	0.06	0.398	0.14	0.356	0.14
	UMCP-TD-O-B	0.183	0.08	0.117	0.06	0.378	0.13	0.311	0.14
	UMCP-TDN-O-B	0.200	0.08	0.131	0.07	0.425	0.14	0.378	0.14
	UMCP-T-S-B	0.176	0.08	0.114	0.07	0.376	0.14	0.356	0.14
	UMCP-TD-S-B	0.180	0.08	0.119	0.07	0.377	0.14	0.333	0.14
	UMCP-TDN-S-B	0.188	0.08	0.130	0.07	0.419	0.14	0.400	0.14
	UMCP-T-T-B	0.154	0.08	0.086	0.07	0.292	0.13	0.244	0.14
	UMCP-TD-T-B	0.160	0.08	0.088	0.06	0.297	0.12	0.222	0.12
	UMCP-TDN-T-B	0.183	0.08	0.097	0.06	0.339	0.13	0.289	0.13
	UMCP-T-F-B	0.045	0.03	0.019	0.02	0.133	0.12	0.111	0.12
	UMCP-TD-F-B	0.046	0.03	0.020	0.01	0.129	0.09	0.089	0.09
	UMCP-TDN-F-B	0.059	0.04	0.036	0.02	0.173	0.09	0.089	0.08
Baseline	UMCP-T-TOFS-U2-CF	0.210	0.08	0.152	0.06	0.432	0.13	0.333	0.14
	UMCP-TD-TOFS-U2-CF	0.212	0.08	0.153	0.06	0.428	0.13	0.333	0.14
	UMCP-TDN-TOFS-U2-CF	0.210	0.07	0.159	0.06	0.446	0.12	0.333	0.14
	TerrierBaseline-T	0.204	0.08	0.125	0.06	0.410	0.13	0.333	0.14
	TerrierBaseline-TD	0.197	0.08	0.122	0.06	0.384	0.13	0.289	0.13
	TerrierBaseline-TDN	0.203	0.08	0.132	0.06	0.417	0.13	0.333	0.14

Dry Run test collection, so we also submitted an untuned BM25F with all parameters set to Terrier's default values. The untuned runs are indicated by '-U' in the run name. Other expansion cases are denoted '-L', indicating learned parameters. The third alternative is '-B', which indicates the use of BM25 (with default parameters) rather than BM25F.

Because the Dry Run test collection contains no useful Description or Narrative fields (in the that collection, those fields repeat the topic's Title field), we submitted Title (T), Title+Description (TD), and Title+Description+Narrative (TDN) runs for each of our system variants in order to characterize the effect of query length using the final official test collection. For ColBERT we also submitted a D run using only the topic's Description field as the query.

2.2 Summative Evaluation

Table 1 shows folder ranking results for the 25 University of Maryland runs and the 3 baseline runs. Unlike the task overview paper [9], here the runs are grouped into sets that show results for T, then TD, then TDN queries, with the T row highlighted. Those sets are then grouped by one of four run types (Baseline, Single-Field, Expansion, or RRF). We focus our analysis on nDCG@5.

Focusing first on the Single-Field results, we see that OCR > Summary > Title > Folderlabel. Comparing this to what we saw on the Dry Run collection, Title is doing less well. This tends to confirm our long-standing concern that using document titles as queries, as was the case in the Dry Run test collection, tends to produce overly optimistic results when the title field is indexed. Moreover, we now see that OCR text is doing better than any other single field. That comports well with our intuition, which is that OCR text is far more extensive than title metadata, and thus should have a better chance of matching. Finally, we see that Summaries are a bit below OCR text, which is consistent with what we saw on the Dry Run collection. We note, however, that these are GPT4o summaries for which only a single prompt was tried, that other LLMs could also be tried, and thus that improved performance from LLM-generated summaries may be possible.

Table 2 shows per-topic results for the same runs, organized and highlighted in the same way as Table 1. Topics are ordered left to right as in the task overview paper [10] (i.e., from easiest to hardest, when averaged across all runs, including runs submitted by other teams). Topics 16 and 28 are clearly unusual, with almost every system placing the only relevant folder at rank 1 (which is the only way to get an nDCG@5 score of 1). Looking at them, these are simply easy topics for a term matching system. Their titles are "Amateur radio" and "Spread of equine influenza" for Topic 16 and Topic 28, respectively. There's not much to be learned here; in each case there was a relevant document in the training set with the key query terms in both the Title, the OCR and the Summary. As these are highly discriminative terms, highly ranking the best training document (and thus its folder) is easy. However, neither folder has the associated query term in its folder label, even after SNC translation, so our runs with BM25 search of only folder labels (UMCP-*-F-B runs) consistently fail on those two topics.

More interesting is to look at the six topics that have no document from any relevant folder in their training set. These are Topics 8, 12, 26, 33, 40 and 43. For five of those six topics, none of the 28 runs (including the 3 baseline runs) ranked any relevant folder anywhere in their top five. The one exception is Topic 33, where matching on a full-collection folder label search seems to have worked. This is easily seen from the fact that all six UMCP-*-*-CF runs found one or more relevant folders in the top five, but that no other run did so. Since COLBERT consistently missed all the relevant folder(s) in its top five, it must be the full-collection folder label search that explains this result. Topic 33's title is "Coffee Rust Eradication". There are three highly relevant folders (and no other relevant folders) for that topic, one of which has the Brown folder label "AGR4-2 Coffee Rust." So that one folder label is easily found.

Another question we can ask is how often our "expansion" technique finds folders that don't contain any training document(s). Focusing on Title queries, we see TerrierBaseline-T (which has no expansion) places a relevant folder in the top five ranks for 22 of the 45 topics. Disappointingly, our expansion run with Title queries (UMCP-T-TOFS-L2, which omits any counfounding effects from RRF) places relevant folders in the top five ranks only for that same set of 22 topics. However, repeating the same analysis with TD or TDN queries found one additional topic (Topic 38 for TD, Topic 9 for TDN) for which a relevant folder was placed in the top 5 ranks by UMCP-*-TOFS-L2 that had not been found in the top 5 ranks of the corresponding Terrier Baseline. More detailed analysis is needed, but there is at least some hint that our approach to expansion could have occasionally achieved its intended effect.

Finally, we can also ask whether adding ColBERT helps. ColBERT alone yielded results that were notably below BM25F, but for TD or TDN (although not for T) queries ColBERT ranked relevant folders in the top five for different topics than did BM-25F. This relative outperformance of TD and TDN over T is unsurprising, since ColBERT uses contextual embeddings that are pre-trained on text expressed using sentences, and both the description and the narrative fields contain sentences. For example, ColBERT with TD queries adds four topics with relevant folder(s) in the top five

Table 2: Folder ranking, per-topic results, nDCG@, topis ordered as in task overview paper [9].

Topic	16	28	36	2	19	44	23	15	5	20	35	37	11	1	18	22	21	7	4	24	32	39	33	27	38	42	45	41	29	40	34	9	12	6	3	8	10	13	14	17	25	26	30	31	43	Mean
Highly Relevant (HR) Folders	1	1	4	5	6	2	6	4	15	7	3	9	10	6	3	1	1	1	7	2	2	1	3	4		3	5	3	8	2	3	2	2	3	3	1	2	13	1	2	1	1	1	3		3.6
Relevant (R) Folders			5	5	9	16	1	3	11	3	4	5		7	2	5	2	4	3	3	4	1		5	7		2		4	1	6	6	1	8		2	1	10	1	1	1		8	6	1	3.6
No Train Folder is R or HR																							✓							✓			✓			✓						✓			✓	
UMCP-T-TOFS-L2-CF	1	1	0.54	0.61	0.55	0.48	0.55	0.37	0.55	0.55	0.42	0.49	0.49	0.44	0.39	0.20	0.24	0.13	0.28		0.22		0.23	0.08	0.17		0.17																			0.226
UMCP-TD-TOFS-L2-CF	1	1	0.54	0.56	0.55	0.56	0.55	0.53	0.51	0.47	0.42	0.49	0.55	0.32	0.47	0.13	0.24	0.13	0.28	0.19			0.23	0.08	0.32		0.17																			0.229
UMCP-TDN-TOFS-L2-CF	1	1	0.54	0.56	0.55	0.56	0.55	0.61	0.47	0.34	0.42	0.47	0.35	0.44	0.35	0.20	0.24	0.09	0.28		0.16		0.20	0.08	0.17		0.17	0.3			0.05	0.16														0.229
UMCP-ColBERT-T	1	1	0.72	0.61	0.55	0.48	0.55	0.53	0.51	0.34	0.42	0.17	0.17	0.19	0.18	0.20		0.13	0.11	0.21	0.16			0.08																						0.185
UMCP-ColBERT-TD	1	1	0.72	0.50	0.55	0.55	0.55	0.37	0.35	0.32	0.42	0.15	0.34	0.05	0.26	0.20		0.13		0.30		0.17		0.08			0.17							0.05												0.183
UMCP-ColBERT-TDN	1	1	0.56	0.50	0.47	0.48	0.55	0.19	0.32	0.30	0.42	0.15	0.34		0.18	0.20				0.24							0.17	0.47			0.25															0.173
UMCP-ColBERT-D	0.39	1	0.72	0.46	0.51	0.56	0.51	0.53	0.51	0.32	0.42		0.34	0.07		0.20				0.48		0.28		0.06																						0.164
UMCP-T-TOFS-L2	1	1	0.58	0.61	0.55	0.55	0.55	0.37	0.55	0.55	0.42	0.55	0.55	0.41	0.26	0.13	0.24	0.21	0.11		0.23			0.08																						0.211
UMCP-TD-TOFS-L2	1	1	0.72	0.6	0.55	0.56	0.55	0.56	0.55	0.34	0.42	0.55	0.55	0.33	0.26	0.1	0.24	0.21	0.11		0.08			0.08	0.21																					0.213
UMCP-TDN-TOFS-L2	1	1	0.72	0.61	0.55	0.56	0.55	0.61	0.55	0.34	0.42	0.55	0.38	0.41	0.30	0.20	0.24	0.13	0.11		0.16			0.08								0.16														0.214
UMCP-T-O-B	1	1	0.58	0.61	0.55	0.55	0.55	0.37	0.55	0.55	0.42	0.47		0.13	0.07	0.20	0.24	0.20	0.06		0.16								0.34																	0.191
UMCP-TD-O-B	1	1	0.72	0.58	0.38	0.48	0.55	0.37	0.55	0.55	0.42	0.34	0.13	0.05	0.09	0.20	0.24	0.20	0.06		0.16								0.13																	0.182
UMCP-TDN-O-B	1	1	0.72	0.58	0.49	0.48	0.55	0.37	0.55	0.55	0.42	0.47	0.21		0.14	0.20	0.24	0.33	0.06	0.48	0.16																									0.200
UMCP-T-S-B	1	1	0.69	0.58	0.60	0.48	0.55	0.37	0.51	0.34	0.42	0.53			0.09	0.2	0.24	0.20	0.11																											0.176
UMCP-TD-S-B	1	1	0.70	0.58	0.60	0.48	0.55	0.52	0.55	0.21	0.42	0.40	0.21		0.09	0.20	0.24	0.20	0.11																											0.179
UMCP-TDN-S-B	1	1	0.70	0.58	0.61	0.48	0.55	0.52	0.55	0.34	0.42	0.41	0.15	0.11	0.14	0.20	0.24	0.33	0.11																											0.188
UMCP-T-T-B	1	1	0.61	0.34	0.55	0.39	0.34	0.61	0.13		0.42	0.34	0.51	0.13	0.26	0.09		0.13	0.11																											0.155
UMCP-TD-T-B	1	1	0.42	0.64	0.51	0.47	0.34	0.61	0.34		0.42	0.21	0.49	0.13	0.26			0.13	0.07						0.17																					0.160
UMCP-TDN-T-B	1	1	0.86	0.70	0.55	0.58	0.34	0.61	0.34		0.42	0.34	0.49	0.15	0.21	0.13		0.13	0.05						0.34																					0.183
UMCP-T-F-B					0.34	0.37			0.13	0.34			0.21	0.5										0.12																						0.045
UMCP-TD-F-B					0.17	0.37		0.23	0.13	0.34			0.21	0.5										0.12																						0.046
UMCP-TDN-F-B			0.41		0.17	0.37		0.23		0.34		0.13	0.21	0.47								0.17		0.12																						0.058
UMCP-T-TOFS-U2-CF	1	1	0.54	0.61	0.55	0.55	0.55	0.37	0.34	0.55	0.42	0.47	0.13	0.31	0.30	0.20	0.24	0.20	0.24		0.22		0.23	0.08	0.17		0.17																			0.210
UMCP-TD-TOFS-U2-CF	1	1	0.54	0.56	0.55	0.55	0.55	0.37	0.32	0.55	0.42	0.47	0.34	0.26	0.43	0.20	0.24	0.20	0.22		0.16		0.23	0.08	0.17		0.15																			0.212
UMCP-TDN-TOFS-U2-CF	1	1	0.54	0.56	0.55	0.48	0.55	0.52	0.28	0.21	0.42	0.34	0.34	0.40	0.30	0.20	0.24	0.20	0.17		0.16		0.23	0.08	0.17		0.17	0.3			0.06															0.210
TerrierBaseline-T	1	1	0.72	0.61	0.55	0.55	0.55	0.37	0.51	0.55	0.42	0.51	0.17	0.36	0.09	0.20	0.24	0.13	0.07		0.16			0.08					0.34																	0.204
TerrierBaseline-TD	1	1	0.72	0.61	0.51	0.48	0.55	0.37	0.38	0.55	0.42	0.51	0.30	0.19	0.25	0.20	0.24	0.13	0.07		0.16			0.08					0.13																	0.197
TerrierBaseline-TDN	1	1	0.72	0.58	0.55	0.48	0.55	0.23	0.55	0.55	0.42	0.51	0.17	0.21	0.14	0.20	0.24	0.23	0.07	0.48	0.16			0.05																						0.202

beyond those found by Terrier-Baseline-TD. Using RRF to combine BM25F with ColBERT picks up two of those, Topic 24 and Topic 45.

Looking broadly at these results, we see good results from indexing uncorrected OCR text, some benefit from searching multiple fields, some benefit from ColBERT's contextual embeddings, and some benefit from indexing folder labels for the full collection. We also see that trends among the summative evaluation results on the full collection resemble trends that we saw in formative evaluation on the Dry Run collection, with the notable exception of the Dry Run collection inflating the results from matching title metadata to queries that had also been derived from (other) title metadata. We have seen little benefit from our expansion approach that sought to use SameSNC and SameBox relations to find folders that had no training examples, although we note that the comparisons that we can make from this set of runs have some confounds that we might be better able to explore in future experiments.

2.3 Next Steps for Folder Ranking

The most striking result we have seen is that no system we have been able to build has been able to place any relevant folder in the top five for about one-third of all topics (14 of the 45 topics). It remains to be seen whether some combination of expansion and better ranking techniques can improve our performance on those difficult topics. Now that we have a richer and more representative test collection than the Dry Run test collection, we can perform more nuanced formative evaluation. Of course, once we do so we will then need a new test collection on which to perform summative evaluation. Perhaps we will be able to create a second set of topics for this collection at NTCIR-19.

Another question that we could now explore is the effect of different training sets on the evaluation results. So far we have explored only uniform sampling at the rate of five documents per box, but of course real archival collections are very often unevenly digitized. So there is a substantial scope for experimentation with different training sets, either uniformly or unevenly sampled.

Finally, we have not yet looked at jointly optimizing box and folder ranking. Our present systems rank folders, but they do so without regard to how many boxes would need to be requested in order to get each of those folders. In practice, however, the cost of a request depends on how many boxes are requested, and on the physical proximity of those boxes. If we are to account for this in our evaluation, we will need to develop new evaluation measures.

3 Subtask B: Archival Reference Detection

Our interest in Archival Reference Detection is motivated by a long term goal of using the detected archival references as additional training data for a Subtask A system, particularly for archival content that is not otherwise well described. We imagine that the documents being cited have been well enough described in the text of the paper at or near the point of citation, and that the principal value to us of a footnote or endnote is therefore to indicate where in an archival repository the item that is being cited can be found. That perspective shaped our work in two ways. First, we are not interested in using a footnote or endnote to learn only of the existence of a document that might be in an archive – our interest when processing a footnote or an endnote is to learn where some specific document can be found in some specific archival repository. That's not to say that we would not use additional descriptive text in a footnote or an endnote, just that descriptions alone do not (in our opinion) an archival reference make. This decision was important to our work because the Dry Run collection that was available for training was rather small, so some additional annotation would be needed in this first year of the task. Of course, doing annotation requires that that an annotator know what they are looking for, and the Subtask B guidelines did not include specific guidance on how to recognize an archival reference.

The second way in which our perspective shaped our work is that to use Subtask B results in a Subtask A system, we would want our Subtask B system to find a lot of archival references. Prior work had found that about 1% of footnotes or endnotes contained archival references [10], and that many of those references were to one archival repository (The UK National Archives). If those were accurate estimates, then we might expect to find about 8,500 archival references in the final Subtask B test collection. But if many of those pointed to materials in one archival repository, we would need vastly more than 8,500 archival references if we wished to use Subtask B results as a basis for training Subtask A systems for other archival repositories. So at this early stage in our work, the simplest guideline is that the more we can find, the better. This led us to view our goal as being primarily recall-oriented, although of course all else equal we would prefer to have the highest reasonably possible precision as well.

With that as background, we chose to investigate the degree to which we could maximize recall (with at least moderately good precision) by using active learning. Active learning has in recent years been used for recall-oriented tasks such as discovery of digital evidence in civil litigation [6] or selection of articles for systematic reviews in medicine [2]. So there was at least some reason to expect that active-learning might be useful for recall enhancement in this setting as well. The principal difference between our work and those other applications of active learning is that for our experiments we have far fewer annotation resources. For that reason, we viewed our Subtask B participation as a pilot study, exploring the behavior of active learning on this task early in the process.

We made one other compromise in our work—we performed the annotation ourselves (by the first and third authors of this paper) rather than hiring and training an independent annotator. This was an entirely practical decision because, as organizers, we were already helping to design and manage the annotation process for the final official test collection, and it simply was not possible for us to manage a second independent annotation process in the time that was available. As it turned out, this was a fortuitous decision because what we learned by doing annotation ourselves helped us to improve the independent annotation process for the final official test collection. More on that below.

The organizers provided 1,836 annotated footnotes and endnotes (collectively, “citations”) as the Dry Run test collection. The collection included 671 positive examples and 1,165 negative examples. Our first step was, therefore, to train a classifier on that training data, using passive learning as a single batch. We did this using a scikit-learn SVM.[7] From a cursory examination of the results, it was clearly awful. This led us to look at the training data. It was (in our opinion), just as bad. So our first step was to clean up the training data. To do this, we manually reannotated all 671 of the positive training examples in the collection.

Our definition of what constituted a positive example actually shifted quite a lot during this reannotation process as we encountered the need to make decisions that we had not previously considered. Three of those turned out to be quite consequential.

[7]https://scikit-learn.org/stable/modules/sgd.html

Documents. Early on, we encountered several descriptions of human skulls. Later we found descriptions of jade jewelry. We have in recent papers been writing about looking for documents because the more generic term archivists use for individual elements in their collection (“items”) had confused our readers (is a box an item? Not to an archivist...). But coming face to face with a skull made us decide what we meant by document. It turns out that one of the jade items was a book with the pages etched in jade. That one is pretty clearly a document, but we decided that the skulls and the jewelry were not. But then we found oral history recordings. That fit our world view of what a document is, so we decided recordings were documents. But what about recordings of instrumental music? Here we just sought to keep it simple, and decided that all recordings were documents. Then we encountered some transcribed inscriptions from statues and we needed to decide if the statue itself that had the inscription was a document. We decided it was not, because its primary purpose was not as a document, but that a transcription of its inscription would be a document. We don’t mean to claim that any of these decisions are objectively correct—another researcher with different interests might reasonably make different decisions. But we did discover that all of these decisions needed to be made. So we started writing down our decisions, and our basis for making them. This evolved into an annotation guide.

Archival Repositories. When we write carefully we write archival repository rather than archive because the archival collections that interest us go by many names. For example, sometimes they are a part of a research library (e.g., a special collections unit) and sometimes they are a part of a different type of organization that has a similar collecting mission (e.g., a historical society). But then we encountered cases that had no collecting organization, such as “available from the author.” Is the author an archival repository? We decided no, because we actually care about things that are collected by institutions. So to be an archival repository you need to be an institution. This led to the question of whether the institution must have a place. We decided not — the Internet Archive is an institution in the sense we mean, regardless of where in cyberspace or the physical world it actually exists. There’s a bit of a “we’ll know it when we see it” character to that definition, but enumerating every possible institutional form did not interest us, so we accepted the imprecision. But then we found institutions such as academic departments that you could contact to get a copy of something that someone who worked there had written. We decided that such cases were not archival repositories because they lacked curatorial intent (which we also did not sharply define, but by which we meant something along the lines of collect for posterity). So in the end an archival repository is a collection maintained by an institution that has curatorial intent.

Location. The trickiest decision we needed to make was what it meant for an archival reference to contain a location. In the Dry Run collection, many things marked as archival references lacked information about the location of the item (for example, letters written 200 years ago were marked as archival references, perhaps because the annotator inferred that they must be in some archive somewhere). Such cases were easy for us; we marked them as not archival references. We found many cases with only the name of an archive. We marked those as archival references because we thought they would be useful to a searcher as a starting point for finding a document (essentially, this means that we defined a fourth level for a Subtask A system: document, folder, box, and repository; we were in essence imagining some future repository ranking task). But the harder question was what to do when the repository was not mentioned but some location in a repository was (e.g., Box 14). We decided that in such cases it was reasonable to assume that the author of the paper containing the footnote or endnote we were seeking to classify would not have included such a reference unless somewhere else in their paper the archival repository had also been described. Essentially, this involved accepting a limitation of the Subtask B setup, which is that footnotes and endnotes were being annotated in isolation. So if we could recognize that a footnote or endnote referred to a location in an archive we would mark it as an archival reference, even if we could not tell from that one footnote or endnote what archival repository they were talking about.

There were many other decisions to be made (e.g., what level of certainly was required when something was not absolutely clear, or what to do about words in languages other than English that the annotator may not be able to interpret correctly), and we included guidance on those things in the annotation guide as well. But the three big problems we had not completely thought through before the start of the process were whether skulls are documents, whether an author's personal files are archival repositories, and what constituted a location. Reannotating the Dry Run test collection forced us to make each of those decisions. Of course, each decision must also have been made when the training set was originally annotated, but they were simply made differently than we now did for our current research purpose.

We therefore reannotated all 671 positive examples in the Subtask B Dry Run test set, finding 238 to be positive examples by our definition. Because we considered the other 433 originally positive examples in the Dry Run test set to be close negatives, we used those 433 cases as negative examples in our experiments, and we made no use at all of the negative examples that had been distributed with the Dry Run collection. From here forward, when we refer to positive examples, we mean examples that are positive by our definition.

Training an SVM on this set of 238 positive and 433 close negative examples and then running that classifier on the final official Subtask B test collection resulted in our first submitted run, UMCP-SGDC. We then implemented active learning using uncertainty sampling with the ModAL python package.[8] Straightforward implementations of uncertainty sampling tend to select duplicates, and there are many duplicates in the collection. To minimize human effort, we recorded our decisions and and then automatically assigned the same decision to future exact duplicates. This decision also had the effect of creating a growing cache of human annotations that we could use in subsequent experiments. We therefore kept all annotations that we generated during system development and we used a classifier trained on those annotations as a starting point in our active learning experiments. Because the initial annotations were made on the Dry Run collection and all later annotations were made on the final official test collection, we concatenated those two collections for system training, but for testing we ran only on the final official test collection. This concatenation also simplified feature generation since all features for the final official training set had been used when training the original classifier.

Our decision to record annotations made during active learning had an additional advantage: we could correct some classifier errors prior to submission. After training a classifier and then using that classifier to make decisions on every item in the test set, we simply replaced the classifier's decision with the annotator's decision for any item that the annotator had annotated. This is a simple hack for converting an inductive classifier (i.e., one designed to generalize to unseen content) into a transductive classifier that works over a set of items that was already known at training time.

We experimented with seven scikit-learn classifiers, all with their default parameters. The only one we used in any submission was the one we call SGDC, which was a Support Vector Machine (SVM) with a linear kernel, trained using Stochastic Gradient Descent (SGD), with score calibration performed using cross-validation. We also tried the same classifier without score calibration, a second SVM classifier (SVMlite) using either a linear or a Radial Basis Function (RBF) kernel, logistic regression, and two variants of Naive Bayes (a straightforward multinomial Naive Bayes implementation, and a second implementation that scikit-learn calls ComplementNB that is designed for tasks with high class skew). We consistently saw the best results (as defined below) from SGDC.

We tried three approaches to select examples for annotation: relevance sampling, uncertainty sampling, and ModAL's uncertainty batch sampling. We found uncertainty sampling to be the best. Relevance sampling is generally preferred when positive examples are relatively rare [3], but once we initialized a classifier using our manually reannotated Dry Run set, positive examples were easily found. Relevance sampling thus resulted in our annotating large numbers of positive examples, which was not particularly helpful. We avoided some of the problem with uncertainty sampling selecting many duplicates by automatically replicating the annotation for previously seen exact duplicates (although we still had to deal with near duplicates), so uncertainty sampling was a reasonable choice. ModAL's uncertainty batch sampling seeks to maximize the utility of an entire set of annotations, but we found that it had a penchant for showing us Cyrillic or Arabic scripts, neither of which we could read. In an effort to avoid this, we used the Python alphabet detector package[9] to remove 35,716 footnotes and endnotes in which the characters were more than 75% Cyrillic or Arabic from the final official test collection. We still found that ModAL's uncertainty batch sampling was turning up what seemed to us to be corner cases more often than it found what seemed to likely be broadly useful training examples, so ultimately we gave up on uncertainty batch sampling and simply used uncertainty sampling. As a nice side

[8]https://github.com/modAL-python/modAL

[9]https://github.com/EliFinkelshteyn/alphabet-detector

Table 3: Subtask B results. Top block used active learning.

UMCP- Run Name	Positive Trng Examples	System Yes	Estimated Correct Yes	Precision	Recall	F_1
GPT-SGDC-U-2-30	620	7,804	5,424	0.695	0.119	0.155
GPT-SGDC	560	7,219	5,061	0.701	0.110	0.145
SGDC-U-3-75	463	5,060	3,916	0.774	0.081	0.111
GPTauto-SGDC	455	4,857	3,623	0.746	0.074	0.099
SGDC	238	4,464	2,973	0.666	0.061	0.082

effect, ModAL's implementation of uncertainty sampling is very much faster than its implementation of uncertainty batch sampling, which was helpful when doing interactive active learning.

Once we settled on SGDC and uncertainty sampling, all that remained was to select a batch size and number of iterations. In general we selected a batch size that allowed us to finish the number of annotations we wished to make using just a few batches. This then led to our second submitted run, UMCP-SGDC-U-3-75, in which we did three batches of 75 each.[10] For this run we used all of the annotations that we had accumulated during system development for initialization.

After submitting those two runs, Tokinori Suzuki shared the results of GPT4o annotation on the final official test collection with us. One potential weakness of active learning is that it can iteratively learn one part of the space well, but it may take quite a lot of annotation before it discovers positive examples that use entirely different vocabulary. We speculated that GPT might thus help with diversity. Looking at the GPT results, we saw many cases in which (in our opinion) GPT had gotten it wrong, so we simply took the GPT positive examples, reannotated them ourselves as a batch, and then added our annotations for what GPT had originally called positive (i.e., our positives, and what we then treated as our close negatives) to the training data that had accumulated from all of our prior runs (including, by that point, test set annotations from UMCP-SGDC-U-3-75). This produced UMCP-GPT-SGDC. Note that we classify that run as active learning because although it was actually run using our passive learning configuration, it uses training data that had previously been selected for annotation using active learning. We then sought to further improve the resulting classifier by doing two rounds of 30 test set annotations using active learning with uncertainty sampling, thus producing our UMCP-GPT-SGDC-U-2-30 run, which turned out to be our best run by every measure except Precision.

Finally, we wondered what would have happened if we had simply trusted GPT as a basis for training a classifier. To answer that question, we ran passive learning using our initial manually annotated training set plus GPT's original positive decisions (i.e., as they had originally been provided, before we did manual reannotation). This produced our UMCP-GPTauto-SGDC run.

3.1 Formative Evaluation

Because we had used all of the Dry Run test collection for training, for formative evaluation we had to rely on manual inspection of the classifier's results. Because of this, all of our runs (including our passive learning runs) are classified as "manual runs" according to the Subtask B guidelines. Ultimately what we wanted was to find as many correct positive annotations as possible. That led us to establish as our formative evaluation measure to be optimized maximizing the number of positive decisions made by systems whose positive decisions were usually correct. In other words, we sought to maintain a relatively high precision while maximizing relative recall. We initially simply displayed a random sample of positive classification results for a single classifier. If the vast majority of them looked good then we deemed the precision to be acceptable. This is how we learned that SGDC was consistently doing reasonably well at precision, and that it was consistently finding the greatest number of positive examples. Later in our work we ran every classifier every time, and we were then able to display random samples from what one classifier called positive that another classifier had called negative; we typically showed three randomly selected examples from each, for all possible classifier pairs. We used this display to look for cases in which system combination might be useful, but we never found such a case. So all of our submitted runs are single-system SGDC runs.

[10]Fewer than 75 manual annotations were actually needed per batch because memorized annotations could be applied when exact duplicates were found.

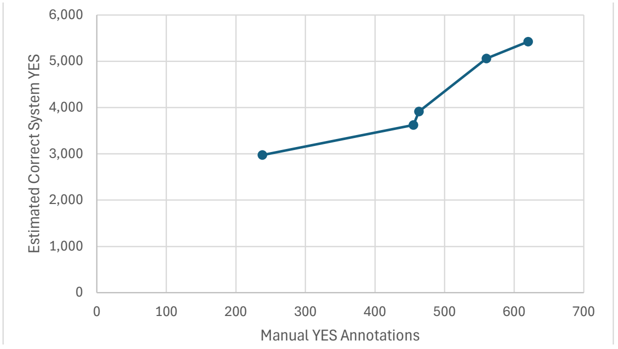

Figure 1: Amplification of manual positive annotation.

3.2 Summative Evaluation

Table 3 summarizes results for our five runs. One clear conclusion is the more positive training examples you have, the better. We see Precision near 0.7 for all runs, with the best recall (and hence the largest number of positive cases found) coming with the largest number of training examples. The raw GPT results seem to be only modestly beneficial (compare UMCP-SGDC to UMCP-GPTauto-SGDC), but manually checking those and adding the positive and close negatives to the training set was a big win (compare UMCP-SGDC-U-3-75 to UMCP-GPT-SGDC). Active learning was only modestly beneficial (compare UMCP-SGDC-U-3-75 to UMCP-SGDC and compare UMCP-GPT-SGDC-U-2-30 to UMCP-GPT-SGDC), although we note that we performed very few active learning iterations. All of these results use only SGDC, but in formative evaluation (which was on the same collection) we saw that no other system returned as many positive classification decisions, so it seems unlikely that any other classifier would have done as well by Recall. Because F_1 is always closer to the lower of Precision and Recall, and because Recall is much lower than precision, it thus seems unlikely that any other classifier would have done as well by F_1.

Figure 1 shows another way to look at these results. The horizontal axis in that figure is the number of positive examples that were found through manual annotation, and the vertical axis is the total number of true positive examples that a system trained with that number of manually annotated positive examples was able to find. As can be seen, our best classifier produces about a 10:1 correct positive annotation ratio. We should be careful to note, however, that our horizontal axis is based on positive manual annotations, and that the total number of manual annotations that are needed to find those positive examples is about twice that value. We should also note that we would expect diminishing returns with addition training data, although we do not yet see that effect over the limited range of training data that we have looked at.

One important caveat on our summative evaluation results on the final official test collection is that the annotation guidelines used by the independent annotator who annotated the final official test collection were essentially the same as the annotation guidelines that we had developed for our own work. That's both good and bad—good in that the annotator had well worked out annotation guidelines; bad in that we can't now know how well our systems would have done in an arms-length summative evaluation. For a pilot task in which the only participants were the organizers, this seems like a fine tradeoff to have made. But in future evaluations we will want to be careful to freeze the annotation guidelines in advance and share them with all participants.

3.3 Next Steps for Archival Reference Detection

The quite low recall of all of our systems is in one sense good news, since it indicates that there is much more to be found than our present classifiers are able to find. We have looked at the annotations for the "bottom" stratum (i.e., the stratum that was sampled very sparsely from cases that no run had classified as positive), and we agree that some of the footnotes or endnotes in that stratum with positive annotations are truly archival references. Given the sparse sampling rate, each positive sample in that stratum suggests the presence of more than 500 actual archival references in the full collection [9], so it seems reasonable to expect that there are thousands, and perhaps tens of thousands, as yet undetected archival references that are still to be found. Although our best system found less than 1% of the collection to contain an archival reference, the true number may be closer to 3%. That's good news, since as we said at the outset, the more the better from the perspective of our ability to ultimately use these archival references in systems like those in Subtask A.

That, however, leaves open the question of how to find many of those undetected archival references. We now have a much larger number of positive training examples available, notably including the positive examples in the bottom stratum—some of which may be quite unlike the ones that we have found to date. So training on more data is an obvious next step. Another obvious thing to try is a neural classifier, some variant of BERT. Such classifiers are data hungry at training time, but we now have more training data. Moreover, such classifiers can also be pre-trained using large unannotated collections, and we now have that as well. Finally, we suspect that if we could do extraction from archival references well (e.g., to find the archive name and the location within that archive) then we could co-train classifiers and extraction systems in such a way that a classifier could use the extraction system's results as a clue for when it is on the right track. Of course, the extraction task is still some distance in the future for us, but a good first step in that direction is the Archival Reference Boundary Detection Task, and we note that there is now some training data for that task.

4 Conclusion

Looking back over our work on the two SUSHI subtasks, we have learned quite a lot. First, we had the opportunity to look closely in Subtask A at how inference might be done, and at least on the Dry Run collection it seems that the combination of same box and same SNC may be a useful signal. More work is needed to see how best to use that signal. We also now have a much better test collection with which to look into other signals, such as the box sequence, that we have not yet looked at in any detail. Subtask B turned out to be richer than it first appeared, with many consequential decisions, and ultimately with better results than we have previously seen. Our original motivation for participating in Subtask B had been to contribute to the stratified sampling that was used to guide assessment, but we got much more out of it than that. It will also be interesting in the future to do analysis on the output of these classification systems, since they can, for example, also be used to characterize the impact of archival collections on scholarship in much the same way as bibliographic citations can be used to characterize the impact of publications [8]. In the end, SUSHI has opened up as many new questions as it has answered. Which seems appropriate for a pilot task.

Acknowledgments

We thank Tokinori Suzuki for sharing GPT summaries for Subtask A and GPT classification results for Subtask B with us. Having access to those GPT results helped to enrich our experiments.

References

[1] Gordon V Cormack, Christopher R Palmer, and Charles LA Clarke. 1998. Efficient Construction of Large Test Collections. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* 282–289.

[2] Wojciech Kusa, Guido Zuccon, Petr Knoth, and Allan Hanbury. 2023. Outcome-Based Evaluation of Systematic Review Automation. In *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval.* 125–133.

[3] David D Lewis. 1995. A sequential algorithm for training text classifiers: Corrigendum and additional data. In *ACM SIGIR Forum*, Vol. 29. 13–19.

[4] Craig Macdonald and Nicola Tonellotto. 2020. Declarative Experimentation in Information Retrieval using PyTerrier. In *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval.* 161–168.

[5] Douglas W. Oard. 2023. Known by the Company It Keeps: Proximity-Based Indexing for Physical Content in Archival Repositories. In *27th International Conference on Theory and Practice of Digital Libraries.* 17–30.

[6] Douglas W Oard and William Webber. 2013. Information Retrieval for E-Discovery. *Foundations and Trends in Information Retrieval* 7, 2–3 (2013), 99–237.

[7] Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. 2022. ColBERTv2: Effective and Efficient Retrieval via Lightweight Late Interaction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics.* 3715–3734.

[8] Donghee Sinn. 2013. The Use Context of Digital Archival Collections: Mapping with historical research topics and the content of digital archival collections. *Preservation, Digital Technology & Culture* 42, 2 (2013), 73–86.

[9] Tokinori Suzuki, Douglas W. Oard, Shashank Bhardwaj, Emi Ishita, and Yoichi Tomiura. 2025. NTCIR-18 SUSHI Pilot Task Overview. In *Proceedings of NTCIR-18.*

[10] Tokinori Suzuki, Douglas W. Oard, Emi Ishita, and Yoichi Tomiura. 2023. Automatically Detecting References from the Scholarly Literature to Records in Archives. In *25th International Conference on Asia-Pacific Digital Libraries, Part II.* 100–107.