We describe the Webis group’s participation in the TREC 2022 Deep Learning and Health Misinformation tracks. Our runs submitted to the Deep Learning track focus on improving the pairwise retrieval model duoT5 by combining a greedy aggregation algorithm with document augmentation. In the Health Misinformation track, our submissions to the Answer Prediction task exploit pre-trained question answering and claim verification models, whose input is extended by evidence information from PubMed abstracts. For the Web Retrieval task, we explore re-ranking based on the closeness of the predicted answers for each web document in the initial ranking to the predicted “true” answer of a topic’s question.

1 INTRODUCTION

We participated in two TREC 2022 tracks: Deep Learning and Health Misinformation. As for the Deep Learning track, with our four runs we investigate whether the default aggregation of pairwise preferences in duoT5 can be further improved. The default implementation of duoT5 already is very effective by deriving and sum-sum-wise aggregating pairwise preferences for all possible pairs of documents. We investigate whether a different greedy aggregation scheme or whether deriving each pairwise preference multiple times using perturbed variants of the query or the documents can help. Our results show that a greedy aggregation improves the effectiveness of duoT5 substantially, but calculating the pairwise preferences multiple times with perturbations does not improve the effectiveness.

As for the Health Misinformation track, in our 20 runs (10 for each task) we use several pre-trained question answering (QA) and scientific claim verification systems to predict a “correct” yes/no answer to a topic’s question. As input to the systems, we use PubMed abstracts to add evidence information (context) from trustworthy sources to the questions. The predicted answer is then used as an estimated true answer to construct rankings where documents are simultaneously sorted by their topical relevance and the predicted correctness of the contained information.

2 DEEP LEARNING TRACK

We submitted the results of four approaches to the TREC Deep Learning track. All four systems are implemented in PyTerrier [16] where we first re-rank the task’s official top-100 document candidates using monoT5 [19]. Then, we calculate duoT5 preference scores for all pairs (sometimes in multiple variants) of documents in the top-50 of the monoT5 ranking and compare the official duoT5 aggregation of the duoT5 ranking with a greedy aggregation algorithm. Two variants calculate for each document pair multiple pairwise preferences where we create augmentations of query-document-document triples (e.g., replacing the original query with queries generated via docT5query [17] pre-calculated by Ma et al. [14]). We use ir_datasets [15] to access the passages for re-ranking. We used existing models from Hugging Face for MonoT5 and DuoT5 which are trained on version 1 of MS MARCO (i.e., we do not train models). Using models trained on version 1 is recommended [5] (e.g., version 2 has more noisy labels [6]). Calculating all pairwise preferences (including all augmentations) took roughly 5 hours on a single core of an A100 GPU.

We submitted four runs out of which three were pooled and one is the baseline:

Webis-dl-duot5. We re-rank the top-100 candidates of the official baseline with monoT5 and re-rank the top-50 of the monoT5 ranking with duoT5. We use the implementation of monoT5 and duoT5 in PyTerrier with the models trained on version 1 of MS MARCO mentioned above.

Webis-dl-duot5-g. We re-rank the top-100 candidates of the official baseline with monoT5 and re-rank the top-50 of the monoT5 ranking using duoT5 and greedy aggregation. We use the implementation of monoT5 and duoT5 in PyTerrier with the models
we selected the combination with the highest nDCG@10 on the TREC 2020 DL data. This hyperparameter optimization yielded an approach proposed by Cohen et al. [4], as previous experiments showed that greedy aggregation is more effective than the default sym-sum aggregation [9]. The greedy aggregation algorithm is proven to closely approximate the best total order in terms of the number of violated preferences [4].

Webis-dl-duot5-aug-1. We re-rank the top-25 results of our webis-dl-duot5-g run by aggregating multiple pairwise preferences obtained via duoT5 on augmented document pairs. Out of 9 augmentation patterns (1) no augmentation, (2, 3, 4) using only the first one, two, or three sentence(s) of the passages, and (5, 6, 7, 8, 9) expand the passages with a query obtained via docT5query), two methods to aggregate pairwise scores (greedy and sym-sum), and five methods to aggregate augmented scores (min, max, mean, median, sum), we selected the combination with the highest nDCG@10 on the TREC 2020 DL data. This hyperparameter optimization yielded an approach that used five augmentations (1) no augmentation, (2, 3, 4) by only the first two sentences, (5-9) variants of passage expansions that are aggregated into a single pairwise score using min aggregation, and the pairwise scores are aggregated into retrieval scores using greedy aggregation.

Webis-dl-duot5-aug-2. We re-rank the top-25 results of our webis-dl-duot5-g run by aggregating multiple pairwise preferences obtained via duoT5 on augmented document pairs. Out of 9 augmentation patterns (1) no augmentation, (2, 3, 4) using only the first one, two, or three sentence(s) of the passages, and (5, 6, 7, 8, 9) expand the passages with a query obtained via docT5query), two methods to aggregate pairwise scores (greedy and sym-sum), and five methods to aggregate augmented scores (min, max, mean, median, sum), we selected the combination with the highest MRR on the TREC 2020 DL data. This hyperparameter optimization yielded an approach that used six augmentations (1) no augmentation, (2, 3, 4) by only the first two sentences, (5-9) variants of passage expansions that are aggregated into a single pairwise score using sum aggregation, and the pairwise scores are aggregated into retrieval scores using greedy aggregation.

### 2.1 Evaluation

Table 1 shows the effectiveness of our four approaches in terms of nDCG@10 and the mean reciprocal rank (MRR). We follow the recommendation of the organizers of the shared task and report evaluation results with duplicate documents (even when not removing duplicates might come with disadvantages [7, 8]). Both augmentation runs decrease the effectiveness of duoT5. However, the greedy variant substantially improves upon the original duoT5.

### 3 HEALTH MISINFORMATION TRACK

In our 10 runs submitted to the Answer Prediction task, we use various pre-trained QA and claim verification systems to infer correct answers to the yes/no health questions from 50 topics by aggregating the answers predicted for the top-k PubMed abstracts retrieved when using a topic’s question field as a query. We also use the predicted answers as candidate “true” answers in our 10 runs submitted to the Web Retrieval task. In our ranking approaches, we order documents by combining document topical relevance with the predicted correctness of the contained information.

#### 3.1 Answer Prediction Task: Our Runs

To predict a “correct” answer to the 50 yes/no health questions like “Are vaccines linked to autism?”, we test pre-trained question answering and claim verification models. As input to the models, we use a topic’s question field and relevant evidence information extracted from trustworthy sources like PubMed.3

**Runs using QA models.** For each topic, we first retrieve 20 or 1000 PubMed abstracts as evidence candidates by submitting topics’ questions to one of the following retrieval systems: (1) PubMed API, (2) Google Custom Search API, or (3) BM25 retrieval (Elasticsearch implementation) on an index of 33.5 million PubMed abstracts.7 For each question–abstract pair, we then let a QA model predict an answer score between 0 (no) and 1 (yes). As QA models, we use (1) a BioLinkBERT-large model [25] pre-trained on PubMed QA [11], (2) a Roberta-BoolQ-base model [13] pre-trained on the BoolQ dataset [2], and (3) a UnifiedQA-T5-large model [12] pre-trained on various question answering datasets.10 We do not fine tune the models. As the final answer score for the topics’ questions (in the range from 0 to 1), our runs use different ways of aggregating the predicted answer scores for each evidence document. Following the task requirements, we include in each run a numerical score and a binary yes/no answer label (using a decision threshold of 0.5). Our submitted runs are:

- **Webis-goo-boolq-abs.** For each topic task, we use the question field as a query to the Google Custom Search API (limited to a search in PubMed). For each of the top-20 returned PubMed abstracts, the pre-trained RobERTa-BoolQ-base model predicts the probability of whether the abstract is on the ‘yes’-side of an answer. The final answer prediction score for a topic is the average of the individual answer scores across all abstracts.

- **Webis-goo-lbq-abs.** Analogous to the previous run, but using the pre-trained BioLinkBERT-large QA model.

---

4. [https://www.elastic.co/](https://www.elastic.co/)
5. [https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7021193/](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7021193/)
6. [https://huggingface.co/Tejas21/bio-linkbert-large-finetuned-boolq](https://huggingface.co/Tejas21/bio-linkbert-large-finetuned-boolq)
7. [https://huggingface.co/allenai/unifiedqa-t5-large](https://huggingface.co/allenai/unifiedqa-t5-large)
We do not fine-tune the models. Even though the models were castorini/duot5-3b-med-msmarco a claim and a text passage), we take these predictions as yes/no originally trained to predict the support/refute probabilities (given on the data from the TREC 2019 Health Misinformation track [1].

We add PubMed abstracts from the PubMed abstracts index (topic question fields are used as queries). Then, we re-rank the results with monoT5 [19] and again re-rank the top-50 results (of the first re-ranking step) with duoT5 [19].11 We use PyTerrier [16] to implement re-ranking. The predicted answer scores returned for 1000 question–abstract pairs using the UnifiedQA-T5-large model are aggregated in the topic's final answer prediction score by discounting ranking positions, assuming that answers from higher ranked (i.e., more relevant) abstracts might be closer to the true answer and thus should contribute more to the final topic's answer score.

We aggregate the topic answer score as follows: (1) Given the predicted answer scores $score_i$ for the abstract at rank $i$ we compute the discounted cumulative answer score DCA for top-$k$ documents:

$$DCA_k = \sum_{i=1}^{k} \frac{score_i}{\log_2 (i + 1)}$$

Then, (2) the normalization factor is computed as the maximum achievable (ideal) discounted cumulative answer score IDCA:

$$IDCA_k = \sum_{i=1}^{k} \frac{1}{\log_2 (i + 1)}$$

Finally, (3) we use the normalized discounted cumulative answer score $nDCA_k$ (with $k = 1000$) as the prediction score for the questions from each topic:

$$nDCA_k = \frac{DCA_k}{IDCA_k}$$

**Runs using claim verification models.** For each topic's question, we first retrieve 1000 abstracts from the index of 33.5 million PubMed abstracts using BM25 (topic question fields are used as queries). Then, we re-rank the top-1000 results with monoT5 followed by re-ranking with duoT5 the top-50 results from the first re-ranking step (analogous to the Webis-uniqa-dis run). For each question–abstract pair, we collect predicted answer scores returned by the claim verification model LongChecker [23, 24] pre-trained on the FEVER dataset [22]12 or the Vera model [18] pre-trained on the data from the TREC 2019 Health Misinformation track [1]. We do not fine-tune the models. Even though the models were originally trained to predict the support/refute probabilities (given a claim and a text passage), we take these predictions as yes/no

---

**3.2 Answer Prediction Task: Evaluation**

The results for our runs submitted to the Answer Prediction task as provided by the track organizers are reported in Table 2. Overall, we observe that discounting the answer prediction scores based on the rank of retrieved evidence documents more often than simple

<table>
<thead>
<tr>
<th>Run</th>
<th>AUC</th>
<th>Acc.</th>
<th>FPR</th>
<th>TPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Webis-verasent-dis</td>
<td>0.81</td>
<td>0.70</td>
<td>0.40</td>
<td>0.80</td>
</tr>
<tr>
<td>Webis-longck-dis</td>
<td>0.79</td>
<td>0.64</td>
<td>0.36</td>
<td>0.64</td>
</tr>
<tr>
<td>Webis-nlm-boolq-abs</td>
<td>0.69</td>
<td>0.52</td>
<td>0.96</td>
<td>1.00</td>
</tr>
<tr>
<td>Webis-longck-uniqa-dis</td>
<td>0.66</td>
<td>0.62</td>
<td>0.48</td>
<td>0.72</td>
</tr>
<tr>
<td>Webis-uniqa-dis</td>
<td>0.66</td>
<td>0.62</td>
<td>0.48</td>
<td>0.72</td>
</tr>
<tr>
<td>Webis-longck-uniqa-ax-dis</td>
<td>0.66</td>
<td>0.60</td>
<td>0.48</td>
<td>0.68</td>
</tr>
<tr>
<td>Webis-longck-uniqa-ax-dis</td>
<td>0.65</td>
<td>0.52</td>
<td>0.96</td>
<td>1.00</td>
</tr>
<tr>
<td>Webis-goo-boolq-ax-dis</td>
<td>0.48</td>
<td>0.50</td>
<td>0.88</td>
<td>0.88</td>
</tr>
<tr>
<td>Webis-goo-lbert-title-ax-dis</td>
<td>0.48</td>
<td>0.50</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>Webis-nlm-lbert-ax-dis</td>
<td>0.48</td>
<td>0.50</td>
<td>0.80</td>
<td>0.80</td>
</tr>
</tbody>
</table>

Median all participants 0.71 0.64 0.48 0.80

---

11https://huggingface.co/castorini/monot5-3b-med-msmarco, https://huggingface.co/castorini/duot5-3b-med-msmarco
12Using fever_aci checkpoints https://github.com/dwadden/multivers
averaging yields higher AUC and accuracy scores. Similarly, using claim verification systems is more accurate than QA systems for predicting an answer. These differences however might also be caused by the retrieval approaches used to find evidence documents (at this point we have not evaluated their retrieval effectiveness) or by the differences in datasets both types of systems were trained on. The answer predictor that is based on the Vera claim verification model that uses only the “most relevant” sentence selection from PubMed abstracts is the most accurate in predicting correct answers. However, the approach that uses LongChecker has the lowest false positive rate, worth considering because this error type is most harmful for health-related questions.

### 3.3 Web Retrieval Task: Our Runs

After predicting the answer to a topic’s question, we retrieve documents with BM25 from one billion documents of the C4 corpus [20] that we indexed with Elasticsearch. Then, we re-rank the results with monot5 and again re-rank the top-50 results (of the first re-ranking step) with duoT5 (the same models as in Section 3.1). The answer score for each retrieved document from C4 is then predicted in a similar way as described in Section 3.1. To combine a document’s answer score with the retrieval score for the final ranking, we first calculate the difference of the predicted answer scores between the topic T (predicted “true” answer) and each document D:

$$\Delta\text{answer}(D) = |\text{answer}(T) - \text{answer}(D)|$$

Our runs further use the closeness $1 - \Delta\text{answer}$ to the predicted “true” topic answer to boost the initial retrieval scores.

**Runs with a linear score boosting.** For each run in this group, we use BM25 to retrieve 1000 abstracts from the PubMed abstracts index (topic question fields are used as queries). Then, we re-rank the results with monot5 and again re-rank the top-50 results (of the first re-ranking step) with duoT5 using PyTerrier [16]. Answer conflicts are resolved with axiomatic re-ranking (more recently published abstracts are ranked higher). For each of the 1000 question–abstract pairs, we collect answer prediction scores returned by pre-trained claim verification and/or QA models for each retrieved abstract and then aggregate the final topics’ answer score by discounting ranking positions with nDCGA900. We then retrieve 1000 documents from C4 using Elasticsearch’s BM25, re-rank with monoT5, and the top-50 of the first re-ranker with duoT5. Document answer scores are predicted with the same claim verification and QA models as in Section 3.1. Using the aggregated topic answer score and the individual answer scores for each retrieved C4 document, we boost the retrieval score linearly based on the closeness between a re-ranked document’s answer score and the predicted topic answer:

$$\text{score}_{\text{lin}}(D) = \text{score}_{\text{duoT5}}(D) + (1 - \Delta\text{answer})$$

**Webis-longck-uniqa-ax-lin.** Predict answer scores using the averaged UnifiedQA and LongChecker scores (abstract texts, abstract titles (only LongChecker), and document text as a context input).

**Runs with a polynomial score boosting.** Using the aggregated topic answer score and the individual answer scores for each retrieved from C4 document, we boost retrieval scores as follows:

$$\text{score}_{\text{pol}}(D) = \text{score}_{\text{duoT5}}(D) \cdot (1 - \Delta\text{answer}^2)$$

**Webis-longck-ax-pol.** Predict answer scores for abstracts and documents using the pre-trained claim verification model LongChecker (fever_sci checkpoints, again) with abstract texts, abstract titles, and document text as a context input.

**Webis-uniqa-ax-pol.** Predict answer scores using the pre-trained QA model UnifiedQA-T5-large (abstract texts and document text as a context input).

**Webis-longck-uniqa-ax-pol.** Predict answer scores using the averaged UnifiedQA and LongChecker scores (abstract texts, abstract titles (only LongChecker), and document text as a context input).

**Webis-longck-uniqa-ax-lin.** Predict answer scores using the averaged UnifiedQA and LongChecker scores (abstract texts, abstract titles (only LongChecker), and document text as a context input).

**Webis-longck-uniqa-ax-pol.** Predict answer scores using the averaged UnifiedQA and LongChecker scores (abstract texts, abstract titles (only LongChecker), and document text as a context input).

**Webis-longck-uniqa-ax-lin.** Predict answer scores using the averaged UnifiedQA and LongChecker scores (abstract texts, abstract titles (only LongChecker), and document text as a context input).

**Webis-longck-uniqa-ax-pol.** Predict answer scores using the averaged UnifiedQA and LongChecker scores (abstract texts, abstract titles (only LongChecker), and document text as a context input).

### 3.4 Web Retrieval Task: Evaluation

For the Web Retrieval task, the retrieval effectiveness of the submitted runs is evaluated using DCG, precision, and a compatibility measure [3] w.r.t the usefulness, correctness, and helpfulness and harmfulness of documents. The results for our runs in Table 3 show that none of our approaches significantly outperforms the median retrieval effectiveness (some runs are, however, significantly worse). The runs featuring linear or polynomial score boosting (cf. runs (a)–(g) in Table 3) have significantly worse effectiveness (both nDCG and precision) on binary ‘useful’ and ‘correct’ relevance judgments as well as for graded ‘usefulness’ judgments. They are also significantly less compatible with ‘helpful’ results. Runs featuring a score combination (with trade-off $\alpha = 0.75$, cf. runs (h)–(j) in Table 3) achieve a significantly improved effectiveness (nDCG) and are significantly more compatible with ‘helpful’ results compared

---

13Using fever_sci checkpoints from https://github.com/dwadden/multivers
14https://huggingface.co/allenai/unifiedqa-t5-large
to the runs with linear and polynomial score boosting. Axiomatic re-ranking at the answer prediction stage does not significantly change compatibility or effectiveness (cf. runs (f) and (g) in Table 3). Compared to the runs featuring the LongChecker model [23, 24], the runs using UnifiedQA [12] or a combination of both have slightly improved effectiveness and compatibility with linear or polynomial score boosting, but the opposite effect can be observed when using a weighted score combination. Thus, whether the claim verification or QA models are better suited for the task is inconclusive. A weighted combination of the topical relevance with an answer closeness to the predicted “true” is so far the most promising.

4 CONCLUSION

In the Deep Learning track, we investigated the effectiveness of four duoT5 variants. We found that a greedy aggregation is substantially more effective than the original duoT5 at the same efficiency. In the Health Misinformation track’s Answer Prediction task, we investigated the effectiveness of pre-trained question answering and scientific claim verification models in predicting correct answers to yes/no health questions. As input to the models, we used questions and retrieved PubMed abstracts that potentially contain trustworthy evidence information. The experimental results show the investigated claim verification models to be more effective for the task than the investigated question answering models (a possible reason might be the different datasets that were originally used for training). For the Web Retrieval task, all our runs are not particularly effective (rather below the median across all evaluation metrics) but a weighted combination of the topical relevance with documents’ closeness to the predicted “true” answer is significantly more effective than our linear or polynomial score boosting. As for using a predicted “true” answer during retrieval, there was no significant difference between using question answering or claim verification models even though in the Answer Prediction task the claim verification is more effective. One of the possible reasons might be that our current re-ranking approaches do not consider borderline answer predictions (close to the 0.5 answer threshold), which needs further investigation and is an interesting direction for future work.

ACKNOWLEDGMENTS

This work has been partially supported by the DFG in the project “ACQuA 2.0: Answering Comparative Questions with Arguments” (project 376430233) as part of the priority program “RATIO: Robust Argumentation Machines” (SPP 1999).

REFERENCES


Table 3: The Health Misinformation track’s official effectiveness results for our runs. U: useful, Co: correct, Incor.: incorrect. Significant differences to other runs are marked with superscripts (Student’s t-test, $p < 0.0009 = 0.05/55$, Bonferroni-corrected).

<table>
<thead>
<tr>
<th>Run</th>
<th>Compatibility</th>
<th>nDCG (binary)</th>
<th>P@10 (binary)</th>
<th>nDCG (graded)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Help</td>
<td>Harm</td>
<td>U &amp; Co</td>
<td>P@10</td>
</tr>
</tbody>
</table>
| (a) Webis-longck-ax-lin | 0.11ghi | 0.07fjkl | 0.43abcd | 0.27 | 0.09 | 0.49 | 0.49defg
| (b) Webis-uniqua-ax-lin | 0.15ghijkl | 0.12 | 0.50 | 0.31 | 0.13 | 0.56 | 0.56abc
| (c) Webis-longck-uniq-ax-lin | 0.14ghijkl | 0.07fjkl | 0.48 | 0.34 | 0.08 | 0.52 | 0.8defg
| (d) Webis-longck-ax-pol | 0.15ghijkl | 0.09 | 0.47 | 0.32 | 0.11 | 0.54 | 0.8ehjk
| (e) Webis-uniqua-ax-pol | 0.18ghijkl | 0.14abc | 0.52 | 0.36 | 0.15 | 0.58 | 0.8defg
| (f) Webis-longck-uniq-ax-pol | 0.17ghijkl | 0.08 | 0.51 | 0.38 | 0.10 | 0.57 | 0.8ehjk
| (g) Webis-longck-uniq-pol | 0.17ghijkl | 0.08 | 0.52 | 0.38 | 0.10 | 0.57 | 0.8ehjk
| (h) Webis-longck-ax-com | 0.23abcdefghij | 0.15 | 0.58 | 0.55 | 0.18 | 0.66 | 0.8defg
| (i) Webis-uniqua-ax-com | 0.23abc | 0.17abc | 0.58 | 0.52 | 0.23 | 0.66 | 0.8defg
| (j) Webis-longck-uniq-ax-com | 0.23ab | 0.17abc | 0.57 | 0.48 | 0.23 | 0.65 | 0.8defg
| (k) Median all participants | 0.24 | 0.13 | 0.61 | 0.53 | 0.16 | 0.69 |