MDC at SemEval-2023 Task 7: Fine-tuning transformers for textual entailment prediction and evidence retrieval in clinical trials

Robert Bevan and **Oisín Turbitt** and **Mouhamad Aboshokor**
Medicines Discovery Catapult
Macclesfield
United Kingdom
{robert.bevan,oisin.turbitt}@md.catapult.org.uk

Abstract

We present our entry to the Multi-evidence Natural Language Inference for Clinical Trial Data task at SemEval 2023. We submitted entries for both the evidence retrieval and textual entailment sub-tasks. For the evidence retrieval task, we fine-tuned the PubMedBERT transformer model to extract relevant evidence from clinical trial data given a hypothesis concerning either a single clinical trial or pair of clinical trials. Our best performing model achieved an F1 score of 0.804. For the textual entailment task, in which systems had to predict whether a hypothesis about either a single clinical trial or pair of clinical trials is true or false, we fine-tuned the BioLinkBERT transformer model. We passed our evidence retrieval model's output into our textual entailment model and submitted its output for the evaluation. Our best performing model achieved an F1 score of 0.695.

1 Introduction

1.1 Background

A large number of clinical trial reports are freely available online; for example, the National Library of Medicine's clinical trials repository[1] lists over 440,000 clinical trial studies. These represent a valuable resource for researchers and medical practitioners but leveraging this data can be challenging due to the large volume of historical clinical trials and the rate at which new trials are conducted (Bastian et al., 2010).

Efforts to efficiently extract key information from clinical trial reports have been numerous and varied. Several studies aimed to streamline the systematic review process with a view to transitioning experimental medicines into the clinic more quickly. Blake and Lucic (2015) built a system to automatically identify clinical trial interventions and comparators along with their associated endpoints. Kiritchenko et al. (2010) created a system combining a statistical text classifier for relevant sentence identification with a rule-based information extractor to automatically extract 21 key pieces of information from clinical trial records. The system included a web interface to allow users to review a modify the extracted information. Summerscales et al. (2011) addressed the challenge of efficiently extracting and reporting clinical trial research outputs to medical practitioners. Their system extracts key information from research abstracts and calculates summary statistics relevant to evidence based medicine.

Other groups have built resources to facilitate the development of tools to streamline the analysis of clinical trial reports. Nye et al. (2018) annotated character spans corresponding to the population, intervention, comparator, and outcomes (PICO) in a set of 5000 clinical trial abstracts and mapped the interventions to a medical vocabulary. Lehman et al. (2019) presented a data set of full-text clinical trial articles paired with over 10,000 hypotheses about an intervention's effect with respective to a given outcome and comparator alongside several baseline systems for predicting whether an intervention significantly reduced, significantly increased, or had no effect with respect to an outcome. DeYoung et al. (2020) extended the data set by 25%, introduced an abstract only version, and provided stronger baseline models.

1.2 NLI4CT data set

The NLI4CT data set (Jullien et al., 2023) comprises 1000 clinical trial records along with 2400 hypotheses, relevant trial record section labels, relevant subsection evidence labels (i.e. which lines in the corresponding section contain evidence), and entailment labels (i.e. is the hypothesis true or false). The data set is divided into training, development, and test sets. The training set contains 1700 statements referencing 852 trials in total; the development set contains 200 statements referenc-

[1] https://clinicaltrials.gov

Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023), pages 1287–1292
July 13-14, 2023 ©2023 Association for Computational Linguistics

Hypothesis Type	Section	Count
Single	**Results**	86
	Eligibility Criteria	181
	Adverse Events	207
	Intervention	251
Comparison	**Results**	292
	Eligibility Criteria	361
	Adverse Events	341
	Intervention	181

Table 1: Combined training and development set hypothesis and section type statistics.

ing 101 trials; and the test set contains 500 statements referencing 250 trials. Note that the clinical trials included in the training and development sets overlap completely, whereas the test set references an additional 148 trials that do not feature in either the training set or the development set.

Each clinical trial record is split into four sections: eligibility criteria, which outlines prospective participants' attributes and medical history constraints; intervention, which describes any treatments received by the participants; results, which details the outcome of the trial; and adverse events, which lists any observed harmful outcomes as a result of an intervention. Each example in the training and development sets includes a hypothesis which makes some claim about either a single clinical trial or a pair of clinical trials; the relevant section label which defines the clinical trial record section containing the evidence required to confirm or deny the hypothesis; relevant line indices, which define which lines in the relevant section contain evidence; and a hypothesis entailment label. The test set does not contain the relevant line indices or entailment labels. Table 1 lists relevant clinical trial section and hypothesis type statistics.

1.3 Tasks

1.3.1 Task 1: Textual entailment

The goal of the textual entailment task was to build a system that, when presented with either a single clinical trial record or pair of clinical trial records, a relevant clinical trial section label, and a hypothesis about the trial - or pair of trials - is able to predict if the hypothesis is true or false.

1.3.2 Task 2: Evidence retrieval

The goal of the evidence retrieval task was to build a system that is able to identify which lines in a relevant clinical trial record section are relevant to a given hypothesis. Again, the system may be presented with either a single clinical trial record or a pair of clinical trial records.

2 Method

Our approach focused on fine-tuning pre-trained BERT-based transformer models for both tasks (Devlin et al., 2019). BERT based models can process sequences of up to 512 tokens. The text associated with each trial typically corresponds to a significantly larger number of tokens. Fortunately, in both tasks we only needed to consider one section of the clinical trial record which greatly reduced the amount of text the system must process.

Even so, as Figure 1 shows, if we concatenate the hypothesis with the relevant clinical trial record section for each example, the token count exceeds the BERT token limit for ~21% of the examples; this issue is particularly pronounced in the eligibility criteria section where nearly half of the examples exceed the token limit. If we consider only the relevant parts of each relevant trial record section, concatenate them with the hypothesis and tokenize the combined text, ~5% exceed the token limit (Figure 2). Therefore, if we are able to accurately identify the relevant evidence prior to making an entailment prediction, we don't need to worry about the token limit in the vast majority of cases. Consequently, we decided to tackle the evidence retrieval task before attempting the textual entailment task in order to filter the amount of text input into the model.

All models were trained using the transformers library (Wolf et al., 2019) with the PyTorch (Paszke et al., 2019) back-end. Table 2 lists the hyper-parameters that were tuned for each model. Training was repeated 3 times - with different random seeds - for each hyper-parameter combination and the final model weights were chosen according to the validation set macro-averaged F1 score.

2.1 Textual entailment

We fine-tuned the BioLinkBERT-base transformer (Yasunaga et al., 2022) for textual entailment prediction. BioLinkBERT builds on previous domain specific transformer models by incorporating PubMed citation information during pre-training. BioLinkBERT performs strongly across a range of biomedical natural language processing tasks and, pertinently for this task, has shown improved

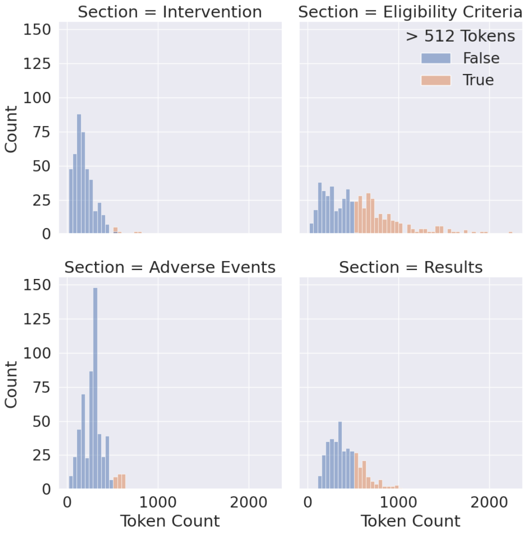

Figure 1: Combined training and development set concatenated hypothesis and section text token count distributions for each clinical trial record section.

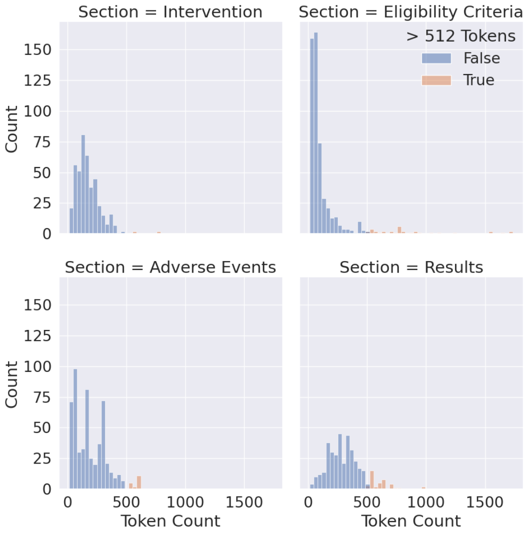

Figure 2: Combined training and development set concatenated hypothesis and relevant subsection text token count distributions for each clinical trial record section.

multi-hop reasoning when compared with other high performing transformer models (Yasunaga et al., 2022). Clinical trial records were processed following Algorithm 1 before they were passed to the BioLinkBERT model. In summary, the hypothesis, primary trial evidence, and secondary trial evidence were concatenated, and prefixed with "Hypothesis:";"Primary Evidence:"; and "Secondary Evidence:", respectively.

We fine-tuned BioLinkBERT to minimise the negative log likelihood loss using the AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$) optimiser (Loshchilov and Hutter, 2017). We trained the model with early-stopping - using the development set loss as the stopping criteria - for a maximum of 30 epochs with a warm-up ratio of 0.1 and an early stopping patience of 8; the model's development set performance was evaluated every 25 training steps. We trained the model using mixed precision training on a pair of NVIDIA V100 Tesla GPUs.

2.2 Evidence retrieval

We fine-tuned PubMedBERT (Gu et al., 2020) for the evidence retrieval task. In order to train a model that is able to identify relevant evidence given a hypothesis, we pre-processed the training and development sets according to Algorithm 2. In summary, for each hypothesis we iterated through each associated trial record and section, building examples of up to 512 tokens containing the hypothesis and concatenated lines taken from the section. These were tokenized and token-level relevance labels were generated - tokens associated with relevant and irrelevant lines were assigned 1 and 0 labels, respectively; special tokens were labelled with -1 (our chosen loss function ignored these).

Algorithm 1 Procedure for pre-processing hypotheses and evidence before passing them to BioLinkBERT.

Input: Hypothesis + evidence lines.
Output: BioLinkBERT input string.

```
▷ Add the hypothesis.
BioLinkBERT_input ← ["Hypothesis:", hypothesis]

▷ Add primary trial evidence.
BioLinkBERT_input ← [BioLinkBERT_input, "Primary Evidence:"]
for evidence in primary evidence do
    BioLinkBERT_input ← [BioLinkBERT_input, evidence]
end for

▷ Optionally add secondary trial evidence.
if ∃ secondary evidence then
    BioLinkBERT_input ← [BioLinkBERT_input, "Secondary Evidence:"]
    for evidence in secondary evidence do
        BioLinkBERT_input ← [BioLinkBERT_input, evidence]
    end for
end if
```

At training time the model expects a sequence of 512 tokens with corresponding labels and attempts to predict each token's relevance. We fine-tuned our model to minimise the negative log likelihood loss using the AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\lambda = 0.01$) optimiser (Loshchilov and Hutter, 2017) on a single NVIDIA V100 Tesla GPU. We trained the model with early-stopping - using the token-level Matthews Correlation Coefficient as the stopping criteria - for a maximum of 30 epochs, with an early stopping patience of 15; the model's performance

Textual Entailment Parameter Grid	
Parameter	**Values**
`Batch size`	**8**, 16
`Learning rate`	1e-5, 2e-5, 3e-5, **5e-5**
`AdamW weight decay`	0, **0.01**, 0.1

Evidence Retrieval Parameter Grid	
Parameter	**Values**
`Batch size`	**16**, 32
`Learning rate`	1e-5, **3e-5**, 5e-5

Table 2: Hyper-parameter values used during grid-search. Optimal values are printed in boldface.

Model	Precision	Recall	F1
Textual entailment	0.668	0.724	0.695
Evidence retrieval	0.814	0.795	0.804

Table 3: Test set performance for best performing models in each sub-task.

was evaluated using the development set every 200 training steps.

Given our model was trained to predict relevant tokens - as opposed to relevant line indices - we had to further process the model's output when generating predictions for the final evaluation. We experimented with assigning relevant line indices according to the following schemes: a line is predicted to be relevant if any of its constituent tokens are predicted to be relevant; a line is predicted to be relevant if the majority of its tokens are predicted to be relevant; and finally, a line is predicted to be relevant if all of its constituent tokens are predicted to be relevant.

3 Results

3.1 Evidence retrieval

Table 3 lists the best performing evidence retrieval model's test set performance. Table 4 shows the evidence retrieval performance for each of the dif-

Section	Precision	Recall	F1
Results	0.868	0.948	0.906
Eligibility Criteria	0.563	0.611	0.586
Adverse Events	0.812	0.973	0.885
Intervention	1.0	0.988	0.994

Table 4: Evidence retrieval model development set performance for each record section.

Algorithm 2 Procedure for generating evidence retrieval task training and validation examples.

Input: Hypotheses, clinical trial section text, relevant line labels, tokenizer.

Output: List of token/token label list pairs.

```
token_limit ← $512$
special_token_count ← $3$
tokens_and_labels ← []

▷ Iterate through the examples.
for each example do
    token_limit ← token_limit − special_token_count
    hypothesis_token_count←# tokens in hypothesis
    token_limit ← token_limit − hypothesis_token_count

    ▷ Examples may refer to one or two trials - iterate through them.
    for each clinical trial listed in example do

        ▷ Clinical trial records contain 4 sections - iterate through them.
        for each section do
            lines ← lines in section
            relevant_line_indices ← relevant line indices
            line_labels ← []
            starting_line_index ← 0
            line_index ← 0
            token_running_total ← hypothesis_token_count

            for each line in lines do
                line_token_count←# tokens in line

                if line_index in relevant_line_indices then
                    line_labels ← [line_labels, [$\vec{1}^{\text{line\_token\_count} \times 1}$]]
                else
                    line_labels ← [line_labels, [$\vec{0}^{\text{line\_token\_count} \times 1}$]]
                end if

                ▷ Check if token limit exceeded or final line reached.
                if line_token_count+token_running_total>token_limit
                or final line reached then

                    ▷ Concatenate hypothesis with lines and tokenize.
                    example_tokens←tokenizer([
                      hypothesis,
                      lines[starting_line_index:line_index]
                    ])

                    ▷ Assign token-level labels.
                    token_level_labels←[
                        $-\vec{1}^{\text{hypothesis\_token\_count}+2 \times 1}$, ▷ [CLF]/
                                                              hypothesis/
                                                              [SEP] tokens.
                      line_labels[starting_line_index:line_index], ▷ Trial
                                                              text token labels.
                        $-\vec{1}^{1 \times 1}$ ▷ [SEP] token.
                    ]

                    ▷ Pad labels to maximum model length.
                    pad_token_count←max(
                      0,
                      token_limit - (line_token_count+token_running_total)
                    )
                    token_level_labels←[
                      token_level_labels,
                        $-\vec{1}^{\text{pad\_token\_count} \times 1}$
                    ]

                    ▷ Store tokens and token labels.
                    tokens_and_labels ← [
                      tokens_and_labels,
                      (example_tokens, example_labels)
                    ]
                end if

                ▷ Reset token running total and starting line index.
                if line_token_count+token_running_total>token_limit then
                    token_running_total←hypothesis_token_count
                    starting_line_index←line_index
                end if

                ▷ Update token running total and increment line index.
                token_running_total←line_token_count
                line_index←line_index+1
            end for
        end for
    end for
end for
```

ferent clinical trial record sections. There's a large discrepancy in performance across the different sections. The model performs poorly when identifying evidence in eligibility criteria, with an F1 score of 0.586; conversely the model does exceptionally well when identifying evidence in the intervention section, with an F1 score of 0.994. Figure 3 helps explains this discrepancy - the box plot shows the distribution of the difference between the total number of lines in each example and the number of relevant lines in each example for each section. The box plots generated for the results and eligibility criteria sections are very different: in the results box plot, any non-zero differences are considered outliers, whereas the eligibility criteria plot has a median difference of 15 and exhibits much greater variance. The task of predicting which lines are relevant in an intervention section is almost trivial, doing the same for the eligibility criteria section is much more difficult. The results and adverse event box plots suggest they may be easier for the evidence retrieval model to process than the eligibility criteria. The results in Table 4 reflect this with the model performing well on both sections. Of the three token prediction to line prediction conversion procedures we tried, we found that predicting a line to be relevant if any of its constituent tokens were predicted to be relevant worked best, although the performance difference between the approaches was minimal.

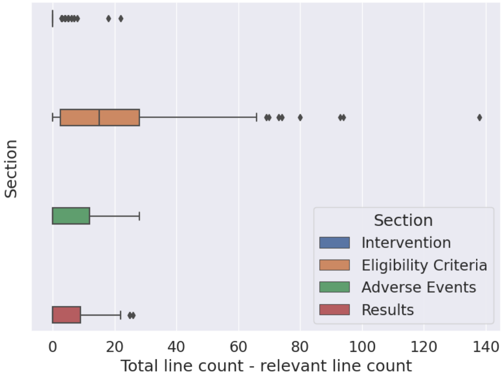

Figure 3: Box plot showing difference in section line count and relevant line count.

3.2 Textual entailment

Section	Precision	Recall	F1
Results	0.719	0.714	0.713
Eligibility Criteria	0.843	0.839	0.839
Adverse Events	0.693	0.692	0.692
Intervention	0.722	0.722	0.722
Combined	0.746	0.745	0.745

Hypothesis Type	Precision	Recall	F1
Single	0.743	0.743	0.743
Comparison	0.752	0.750	0.749

Table 5: Textual entailment model development set performance (macro average). Top: performance for each record section. Bottom: performance for each hypothesis type.

Table 3 lists the best performing textual entailment model's test set performance. Table 5 lists the model's development set performance for each clinical trial section type. Note that the model used the ground-truth evidence to generate the predictions, which explains why the scores are significantly higher than the test set scores. The model performs significantly better when evaluating hypotheses relating to eligibility criteria than any of the other section types; the model performs similarly well across each of the remaining section types. Table 5 lists the model's performance when evaluating hypotheses concerning a single clinical trial, and hypotheses concerning a pair of trials; the model performs equally well in both settings.

3.3 Conclusion

We implemented a pipeline approach to textual entailment prediction in clinical trials data. Our chosen method was primarily motivated by the fact a significant proportion of hypothesis statement - relevant trial record section pairs included in the data set exceed the maximum token limit specified by high-performing domain specific transformer models. Identifying evidence containing lines in relevant sections, concatenating them and combining these with the hypothesis greatly reduced the number of examples that exceeded the token limit. We fine-tuned PubMedBERT for evidence retrieval. Our best performing model achieved an F1 score of 0.804 (10th place). We observed evidence retrieval performance varied significantly across section types; this is a reflection of the data set characteristics: the distribution of relevant line counts varies significantly by section. We fine-tuned BioLinkBERT - a model that has demonstrated improved multi-hop reasoning when compared with PubMedBERT - to perform textual entailment prediction. Our entailment prediction model achieved an F1 score of 0.695 (9th place). The entailment prediction model performed best when evaluating hypotheses focusing on clinical trial eligibility criteria (its performance did not vary significantly across the other section types). While the pipeline approach reduces the number of data set examples that exceed the token limit, ~5% of examples still exceed the token limit. We truncated evidence token sequences that exceeded the token limit before passing them to the textual entailment prediction model. In future work we'd like to include sequences that exceed the token limit - perhaps by applying the model in a strided fashion and averaging its outputs, or using a model with a longer attention span. While the sequential pipeline model is simple and effective, a single model - trained to perform both tasks - would be more efficient and possibly more robust to mislabelled evidence data as the model could learn to identify relevant evidence using the textual entailment signal in addition to the relevant line indices. We plan to investigate this in future work.

References

Hilda Bastian, Paul Glasziou, and Iain Chalmers. 2010. Seventy-five trials and eleven systematic reviews a day: How will we ever keep up? *PLOS Medicine*, 7(9):1–6.

Catherine Blake and Ana Lucic. 2015. Automatic endpoint detection to support the systematic review process. *Journal of Biomedical Informatics*, 56:42–56.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jay DeYoung, Eric Lehman, Benjamin E. Nye, Iain James Marshall, and Byron C. Wallace. 2020. Evidence inference 2.0: More data, better models. *CoRR*, abs/2005.04177.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *CoRR*, abs/2007.15779.

Mael Jullien, Marco Valentino, Hannah Frost, Paul O'Regan, Donal Landers, and André Freitas. 2023. Semeval-2023 task 7: Multi-evidence natural language inference for clinical trial data. In *Proceedings of the 17th International Workshop on Semantic Evaluation*.

Svetlana Kiritchenko, Berry de Bruijn, Simona Carini, Joel Martin, and Ida Sim. 2010. Exact: Automatic extraction of clinical trial characteristics from journal publications. *BMC medical informatics and decision making*, 10:56.

Eric Lehman, Jay DeYoung, Regina Barzilay, and Byron C. Wallace. 2019. Inferring which medical treatments work from reports of clinical trials. *CoRR*, abs/1904.01606.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Benjamin Nye, Junyi Jessy Li, Roma Patel, Yinfei Yang, Iain Marshall, Ani Nenkova, and Byron Wallace. 2018. A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 197–207, Melbourne, Australia. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703.

Rodney L. Summerscales, Shlomo Argamon, Shangda Bai, Jordan Hupert, and Alan Schwartz. 2011. Automatic summarization of results from clinical trials. In *2011 IEEE International Conference on Bioinformatics and Biomedicine*, pages 372–377.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. 2022. Linkbert: Pretraining language models with document links.