PIO extraction on RCT studies

Luke van Leijenhorst
Supervisor: Martha Larson
iCIS - Radboud University Nijmegen

October 2021

1 Introduction

Knowing whether a certain medical intervention has a desired outcome is crucial for determining the right intervention for a patient. Using the best evidence available to make the decision about the treatment of patients is at the core of Evidence-Based Medicine (EBM).

A SR (Systematic Review) of RCT (Randomized Controlled Trial) studies is considered to provide the highest amount of evidence for medical decision making. This evidence is hidden in lengthy RCTs that are time-consuming to analyse and also costly since they have to be analysed by medical professionals due to the technicality and complexity of the texts. On top of that, the number of RCT studies and SRs published in the top 30 contributing countries has been increasing rapidly between 1995 and 2015. [1] By (partially) automating this process, time will be saved and it will allow for faster incorporation of new knowledge into the knowledge base for EBM.

The identification of the elements of the PICO (Population, Intervention, Comparator and Outcome) framework in abstracts of RCT studies could play an important role in identifying the important elements of a study and can assist in the selection procedure of studies for a SR.

In this research we will focus on the extraction of the *PIO* elements subdivided into a total of 16 sub-categories for more detailed identification. The Comparators (C) and the Interventions (I) are both encapsulated into the I class. We will look at a transformer-based approach to this entity extraction task and report on the results. With the resulting model we explore whether data saturation has been achieved. Afterwards, we point out problems with the ground truth and perform an error analysis. In this analysis, we will look at ways in which new data should be selected to increase the quality of the data for the task.

The research question that will be answered is: "How can spans of text containing PIO elements be extracted from RCT studies to speed up systematic reviews for Evidence-Based Medicine (EBM)?".
In the process of answering this research question we will also answer the following sub-questions:

- How well do transformer-based models perform on entity extraction of detailed PIO values on RCT studies in comparison to earlier attempts?
- To what extent does the accuracy measurement of the model reflect the usefulness of the output to medical professionals?
- How to sample additional RCT abstracts for annotation to improve the classifier?
- In what way can data be annotated more accurately to achieve data saturation?

2 Literature review

In this literature review we will first give an overview of larger scale systems with the goal of automating systematic reviews. Then we will look at previous work on the task of PICO extraction and the EBM-NLP dataset that I will be building on in this research. Lastly, we will look at Conditional Random Fields (CRFs), Long short-term memory (LSTMs) networks and transformer models.

(Semi-)Automated SR

For the task of (partially) automating SR we will review two different systems: Evidence Inference and RobotReviewer.

Evidence Inference

In two papers by DeYoung et al [2] [3] a task called *Evidence Inference* was introduced. This task consists of two sub-tasks:

1. Identifying spans of text containing PICO elements in a RCT study
2. Determining the effect of the **I**ntervention to the **O**utcome for a certain **P**opulation in comparison to the **C**omparator.

The papers showed the complexity of the task with f1-scores of around 0.52 on a BERT pipeline for the full task. The f1-scores for the same model with the correct PICO spans already marked, is around 0.77. This significant difference in scores shows the importance of correct PICO element extraction out of RCT studies in the performance of larger automation systems.

RobotReviewer

The RobotReviewer system [4] takes the automation one step further and has additional components that also perform a risk of bias review to analyse the quality of the study and determine whether the given text is actually a RCT study. The system also extracts other relevant information like sample sizes and key findings of the study in addition to the PICO elements.

Trialstreamer

A recent addition to the RobotReviewer system was the Trialstreamer database [5] which automatically updates itself by adding new RCT studies from PubMed and the International Clinical Trials Registry Platform of the World Health Organization to the database. The RobotReviewer system is then run on this database to extract all the information. The results of the system's individual components are promising but still require human verification since errors are still too common.

PI(C)O extraction

In the domain of PICO element extraction there are two different approaches to classify the entities in a text: Sentence classification and Named Entity Recognition (NER) of the PICO entities. We first look at current work in the area of sentence classification

PICO sentence classification

Sentence classification of PICO elements has seen some promising results. A LSTM model trained on abstracts achieved f1-scores of around 0.82 for the PIO elements.[6] Another study [7] applied a transformer-based model to the same problem and achieved f1-scores of around 0.88. The limitations with sentence classification however, is that sentence level classification does not provide information that is detailed enough for complex QA systems or summarization tasks. On top of that, subsequent steps in SR automation systems could benefit greatly from more detailed PI(C)O elements.

PICO-NER

Now we look at PICO extraction as a NER task where the text is first tokenized. Afterwards, each token gets an entity assigned to it. At first, due to a lack of labeled data, larger datasets had to be derived through distant supervision which resulted in noisy labels. [8] To overcome this problem the EBM-NLP dataset was introduced.

EBM-NLP

The EBM-NLP dataset was introduced in an accompanying paper to provide a large labeled dataset for the task.[9] The dataset consists of 5000 abstracts

of RCT studies annotated with spans of text containing PIO elements. 4800 of those annotations were annotated by crowd workers and 200 were annotated by medical professionals. After identifying the PIO elements, the annotators were also asked to sub-divide the PIO elements in more fine-grained entities. After filtering out abstracts that described non-RCT studies or were missing data, 4641 abstracts remained. The full list of sub-categories and the counts for each category in the used development set (10% of the crowd workers' annotations) can be found in figure 1. Along with the dataset, the authors also provided two baseline models for the task: A Logistic regression model and a Bidirectional LSTM-CRF.

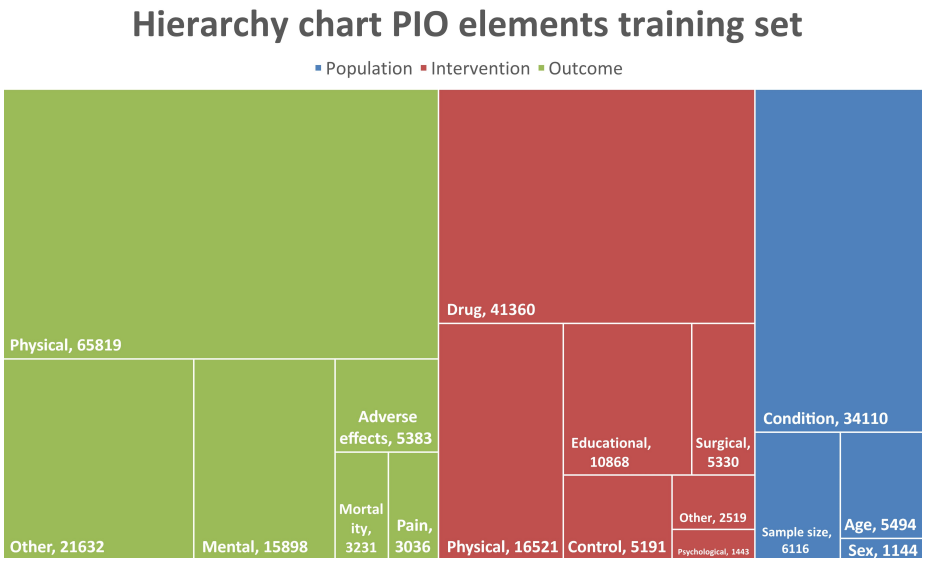


Figure 1: Hierarchy plot of the PIO sub-categories together with their counts in the development set.

Models

Logistic regression

Logistic regression [10] models predict the probability of an outcome based on the feature vector that is given to the model. It does this by multiplying the feature vector by a weight vector and adding a bias term. If the probability for a class is over a certain threshold, it predicts this class. The weight vector and bias term are trained through an optimization algorithm like gradient descent. For handling text (NLP), obtaining feature vectors is often not trivial and you have to manually generate input values from the text. Features often include POS (part of speech) tags, n-grams, character information (e.g. number of special characters or capital letters) and more.

Conditional Random Fields

In a 2001 paper by Lafferty et al. [11] Conditional Random Fields (CRFs) were introduced. A CRF simply applies logistic regression to sequential data and makes use of the fact that the predictions of the model depend on each other. In NLP, this is useful because context carries a lot of information about the meaning of a word or sentence.

LSTM

LSTM [12] is a type of Recurrent Neural Network (RNN) architecture. Just like CRFs, RNNs make use of the dependencies between the models' output. A big difference however, between RNNs and CRFs is that RNNs can find non-linear relations and scale much better. As a result LSTM networks often perform much better than CRFs.

One of the limitations of standard RNN architectures is that long term dependencies often get lost. LSTMs solve this by having input gates, output gates and forget gates. These gates allow the networks' cell to keep or drop information from its context. As a result, LSTM can capture both long term and short term dependencies.

Bidirectional LSTM-CRF

A bidirectional LSTM runs the input both from beginning to end and end to beginning. Because of this, it can capture both dependencies to the left in the sequence of input but also to the right.

A Bidirectional LSTM-CRF [13] is simply a Bidirectional LSTM with a CRF layer on top. This CRF layer makes use of additional features like POS tags and token-level information (e.g. special characters, capitalization). For POS tagging and NER tasks, this model seems to outperform the regular Bidirectional LSTM and LSTM-CRF models by a small margin. However, this model does require you to manually generate features for the CRF layer.

Transformer models

LSTM networks and its variants still come with some limitations. Some of these limitations are solved by transformer models [14].

The first limitation is that in order to capture the dependencies, input sequences have to be processed input by input. As a result, training can not be done in parallel which increases training time vastly. Transformer models pass all arguments of the sequence at once so computations can be done in parallel.

Another limitation in the LSTM network, is that in order to get information about dependencies that occurred much earlier or much later in the sequence,

many computations have to be made. During these computations information loss can occur. In transformer models you have something called a self attention layer which has direct access to the dependencies. By solving these limitations, transformers have been shown to outperform LSTM networks in many NLP tasks.

For the task of detailed PIO extraction, no transformer has been trained. We believe models for this task can greatly benefit from the improvements a transformer gives in comparison to the Bidirectional LSTM-CRF model which is the current best-performing model for this task. That is why we will now look at an implementation of a transformer model and report on the results.

3 Model

For the task of detailed PIO extraction we have trained a transformer model using spaCy 3.1's NER system on the EBM-NLP data.
The EBM-NLP dataset contains a total of 4641 abstracts with detailed annotations for all the sub-categories. A short sample annotation can be found in figure 2. 4457 of the abstracts were annotated by selected crowd-workers and 184 abstracts were annotated by medical professionals. Due to the complexity of the annotation task, the annotations of the medical professionals are used as a test set for more reliable results. This is also in line with the way the models to which we compare the transformer-based model were evaluated. As a validation set, the first 10% of the training set was set aside.

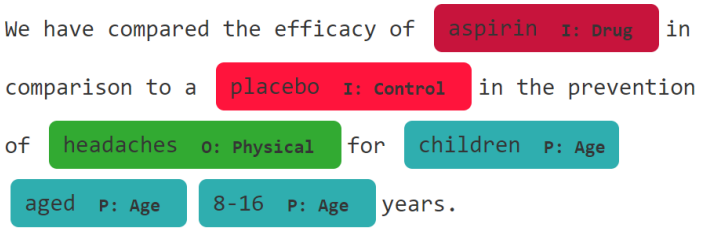


Figure 2: Sample span of text annotated with detailed labels.

Results

The accumulated results of the model for the PIO classes can be found in table 1. We compare our results to the other models on the leaderboard (`https://ebm-nlp.herokuapp.com/`) for this task. We also provide the results for a baseline model where we simply always guess the most frequent assigned entity from the training set (*O: Physical*). The individual scores for each different entity on the transformer model can be found in table 2. The

code, data, step by step instructions and information about the used hyperparameters can be found on the following GitHub page: (`https://github.com/Lukevanl/Thesis_PIO_Extraction/tree/master`).

Combined	**Precision**	**Recall**	**F1-score**
Most Common Entity	0.06	0.27	0.09
LogReg	0.3	0.47	0.36
LSTM-CRF	**0.64**	0.38	0.46
Transformer	0.56	**0.50**	**0.53**

Table 1: Weighted precision, recall and f1-score of transformer model in comparison to the leaderboard and a baseline model all evaluated on the same test set.

Entity	**Precision**	**Recall**	**F1-score**
P: Age	64.15	66.58	65.34
P: Condition	80.66	24.44	37.51
P: Sample size	77.39	67.79	72.27
P: Sex	25.53	75.00	38.10
I: Control	65.75	43.44	52.32
I: Drug	70.67	65.20	67.83
I: Educational	47.01	38.46	42.31
I: Other	0.00	0.00	0.00
I: Physical	21.35	51.63	30.21
I: Psychological	0.00	0.00	0.00
I: Surgical	21.58	41.04	28.29
O: Adverse effects	53.47	29.17	37.75
O: Mental	67.20	25.97	37.46
O: Mortality	67.53	68.42	67.97
O: Other	54.57	36.35	43.63
O: Pain	71.03	59.38	64.68
O: Physical	56.01	63.30	59.43

Table 2: Precision, recall and f1-score of transformer model on each separate entity evaluated on the test set

Looking at the results, we see that the model outperforms the existing Logistic Regression and Bidirectional LSTM-CRF models with a difference of 0.17 and 0.07 on the f1-score respectively. We also see that there seems to be a correlation between the amount of labels for an entity and the f1-score (figure 3). There are some exceptions to this rule when the entity is easier to assign. For example, the entities *P: Sample Size* and *P: Age* have a high f1-score despite a low entity count. This can be explained by the simplicity of assigning these two entities since they are both often numbers. The correlation of elements with higher

entity counts having higher f1-scores can be caused by class imbalance and/or by data saturation not having been achieved.

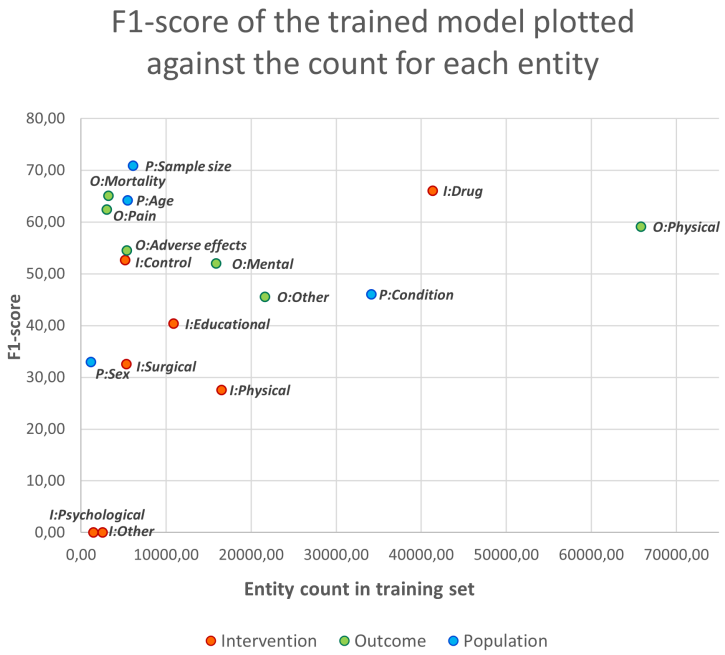


Figure 3: Scatter plot of the F1-score on the test set plotted against the count for each entity in the training set.

4 Data saturation

In the previous section we showed that there seemed to be a correlation between the f1-score and the count of an entity in the trained model (figure 3). To see whether this correlation was caused by data saturation not having been achieved, we have run the model on 20%, 40%, 60% and 80% of the training data with the same early stopping conditions as the original model. The only modification that was made was to the `eval_frequency` hyperparameter which determines after how many iterations the model gets evaluated on the development set. This value was changed with respect to the percentage of the training set to compensate for the decreasing size. So `eval_frequency` is 200 for 100% of the training set, 160 for 80% etc. The used training splits and the exact hyperparameters can also be found in the config file on the GitHub page.

The weighted precision, recall and f1-score of the model on the test set for the different training sizes can be found in figure 4. The individual f1-scores for each entity for the different training set sizes can be found in figure 5. The

entities are sorted on the entity counts in the full training set.

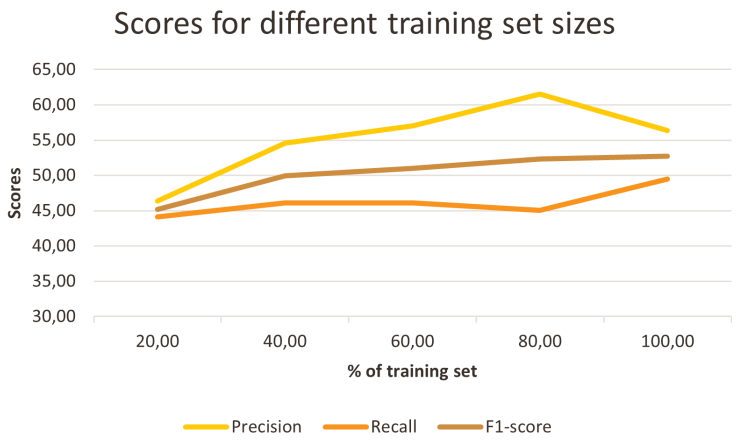


Figure 4: Precision, Recall and F1-score on the test set for models trained on different sizes for the training set.

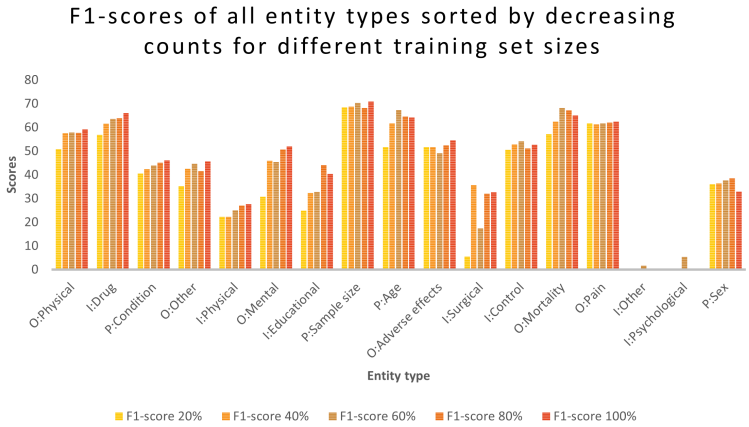


Figure 5: F1-scores for each entity evaluated on the test set trained on different training set sizes sorted in decreasing order on the count of each entity.

These figures show us that the model still benefits from more data. In figure 4 you see an upwards trend for the combined metrics when the training size increases and in figure 5 you can see that in most elements with low and high entity counts, more data increases the f1-scores. This holds for the entities that are easier to assign like *P: Sample Size* and *P: Age* but also for the more complex entities like many of the Interventions. In conclusion, the model could benefit substantially from more labeled data.

To determine how new RCT studies should be sampled and annotated, we first take a closer look at the type of errors the model makes by performing an error analysis. We do this by first comparing the output of the model to the used ground truth. Afterwards, we point out problems with the ground truth and create a ranking of different type of errors and perform a detailed evaluation of the model by hand to get a better idea of the type and severity of the errors that are being made.

5 Error analysis

To know what type of errors the mode makes, we first create the confusion matrix (figure 6) on the development set. Because of the high class imbalance caused by the *None* class, we take the logarithm of each number of the confusion matrix to generate the heatmap for the confusion matrix (figure 7).

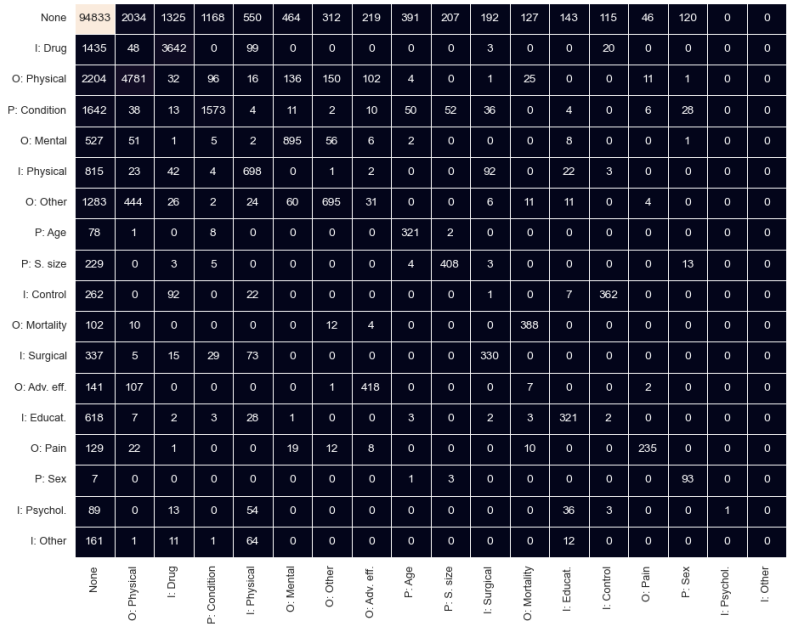


Figure 6: Heatmap of the detailed PIO entities for the development set. The entities are sorted by decreasing entity counts in the development set.

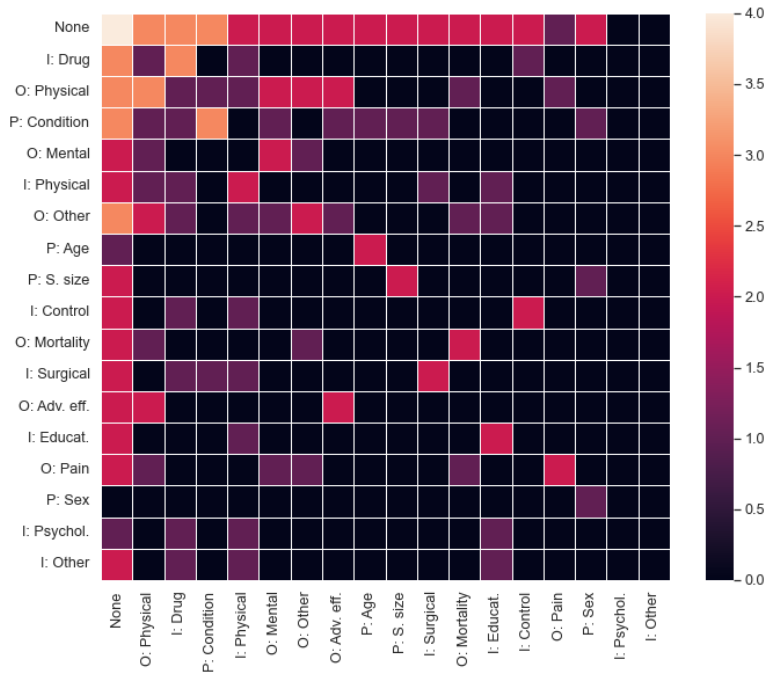
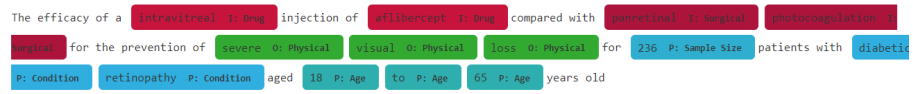


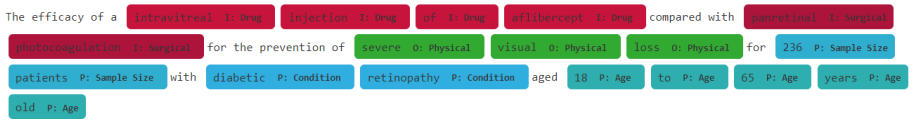
Figure 7: Heatmap of the detailed PIO entities where the log is taken for each count in the development set. The entities are sorted by decreasing entity counts in the development set.

In the confusion matrices we see that very few mistakes are made between the different Population, Intervention and Outcome classes. Instead, the most common mistakes that are made, are false positives (FP) and false negatives (FN) on the *None* class. (TODO: Change for new cm). Of the 19353 total errors 17013 ($\approx 88\%$) are made on the *None* class (6239 FN and 10774 FP).

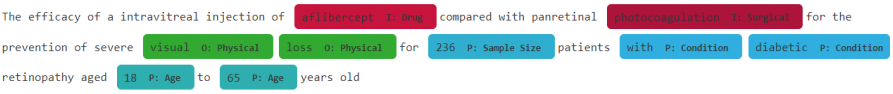
After manually inspecting the output of the model we noticed that evaluating on the ground truth does not always give a good indication about the actual usefulness of the output of the model. Consider the (made up) span annotated with what we consider to be the ground truth in figure 8a. The same span could be annotated in many different ways while losing little to no information. For example, in figure 8b and 8c we annotate with maximal spans and minimal spans respectively. For some applications, maximal or minimal annotations might even be considered better. However, you can see that the f1-score is considerable lower for annotations b and c.



(a) Ground truth: $Precision = 100, Recall = 100, F1 - score = 100$



(b) Maximal spans: $Precision = 72, Recall = 100, F1 - score = 84$



(c) Minimal spans: $Precision = 100, Recall = 69, F1 - score = 82$

Figure 8: Span of text annotated with the ground truth (a), maximum spans (b) and minimum spans (c) together with the precision recall and f1-score of the annotations evaluated against the ground truth.

To get a better overview of the type of mistakes the model makes, we first subdivide the type of errors into 9 distinct clauses which are ranked from most to least critical:

1. Annotation completely missing causing the loss of critical information
2. Annotation was too short causing the loss of critical information
3. Entity assigned where none should have been
4. Entity assigned where another entity should have been of a different PIO element
5. Annotation was too long causing the inclusion of irrelevant tokens
6. Annotation completely missing causing the loss of less critical information
7. Entity assigned where another entity should have been of the same PIO element
8. Annotation was too short causing the loss of less critical information
9. Annotation was too long causing the inclusion of less important tokens
10. Repetition of entity missed

In general we consider false positives (FN) to be more critical than false negatives (FP) because for many tasks it is often better to include non-important information than to miss important information from a clinical perspective. We also make a distinction between information that is considered critical like the drug that was given to the patients and less critical information like the way it was applied (e.g. intravenously). We give repetitions a low rank if it was

assigned the correct entity at some stage but was missed in another. We do this, because very little information is gained if it is annotated a second or third time. If all the occurrences of a certain P, I or O entity were missed they are all assigned a type 1 error.

Now we evaluate the output of the model by hand using the ranked types of errors described above. The evaluation process is separated into two rounds:

1. Take the 10 abstracts from the development set that have a minimum of 100 words, with the highest ratio of FNs and evaluate them by hand.
2. Use the gained insights of round one and a discussion with a medical professional to sample and evaluate 10 more abstracts.

Round 1

For round 1, we have decided to sample based on the ratio of FNs in the abstracts because this is a type of error that is common and we also consider this type of error to be more critical than FPs. At first this only returned very short abstracts. Therefore, we removed abstracts with fewer than a 100 words (the average length of an abstract is about 271 words). During the evaluation process we keep track of the following information:

- The number of errors of each type in each abstract for every PIO element.
- The subject of each abstract: Most sampled RCT studies in the EBM-NLP dataset were either about cardiovascular diseases, cancer or autism. We keep track of the subject to make sure the sample is representative.
- The main intervention type of each abstract

Round 2

6 Conclusion

Re-iterate motivation and most important points and suggest future work.

References

[1] P. Fontelo and F. Liu, "A review of recent publication trends from top publishing countries," *Systematic Reviews 2018 7:1*, vol. 7, pp. 1–9, 9 2018.

[2] E. Lehman, J. DeYoung, R. Barzilay, and B. C. Wallace, "Inferring which medical treatments work from reports of clinical trials," 4 2019.

[3] J. DeYoung, E. Lehman, B. Nye, I. J. Marshall, and B. C. Wallace, "Evidence inference 2.0: More data, better models," 5 2020.

[4] I. J. Marshall, J. Kuiper, E. Banner, and B. C. Wallace, "Automating biomedical evidence synthesis: RobotReviewer," *Proceedings of the Conference of the Association for Computational Linguistics (ACL)*, vol. 2017, pp. 7–12, July 2017.

[5] I. J. Marshall, B. Nye, J. Kuiper, A. Noel-Storr, R. Marshall, R. Maclean, F. Soboczenski, A. Nenkova, J. Thomas, and B. C. Wallace, "Trialstreamer: A living, automatically updated database of clinical trial reports," *Journal of the American Medical Informatics Association*, vol. 27, pp. 1903–1912, 12 2020.

[6] D. Jin and P. Szolovits, "Pico element detection in medical text via long short-term memory neural networks," pp. 67–75, 2018.

[7] L. Schmidt, J. Weeds, and J. P. T. Higgins, "Data mining in clinical trial text: Transformers for classification and question answering tasks,"

[8] B. C. Wallace, J. Kuiper, A. Sharma, M. Zhu, and I. J. Marshall, "Extracting pico sentences from clinical trial reports using supervised distant supervision hhs public access keywords evidence-based medicine; distant supervision; data extraction; text mining; natural language processing," 2016.

[9] B. Nye, J. J. Li, R. Patel, Y. Yang, I. J. Marshall, A. Nenkova, and B. C. Wallace, "A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature," *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, vol. 1, pp. 197–207, 6 2018.

[10] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 20, no. 2, pp. 215–232, 1958.

[11] J. Lafferty, A. Mccallum, F. C. N. Pereira, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," pp. 282–289, 2001.

[12] S. Hochreiter and J. Schmidhuber., "Long short-term memory," *Neural Computation*, 11 1997.

[13] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," 2015.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.