Show US the Data: Database Mention Extraction Using (Sci)BERT

Group 20
Aashutosh Ganesh (s1055287)
Thomas Rood (s1005156)
Pascal Schröder (s1062138)
Radboud University
Nijmegen, The Netherlands

1 INTRODUCTION

A term often coincident with machine learning, deep learning and artificial intelligence is "dataset". When examining the vast expanse of modern machine learning papers, a dataset mention is a necessity as it contributes the authenticity of the paper and the reproducibility of their findings. It serves also to benchmark various strides and breakthroughs in computational models. Datasets are the bedrock for good research and there is a pressing need for us to examine the impact of certain datasets on research.

The works of Mathiak and Boland [12] highlighted the challenges in simple string matching to find datasets in citations. The authors also highlight a pressing issue in research papers where the impact of the dataset isn't as well known as the impact of papers. The "Show US the Data" challenge organisers pose a similar problem: Can Machine Learning and Natural Language Processing methods be employed to retrieve all the dataset mentions in an academic paper?

In this work, we focus on neural approaches to solve the task of retrieving dataset mentions. To this end, we formulate the problem as a Named Entity Recognition (NER) task and employ popular neural models such as BERT [6] and SciBERT [5] to detect dataset mentions as named entities.

2 METHODS

2.1 Metrics for Named Entity Recognition

When viewed as a classification problem, the data for most NLP problems, including NER, would resemble a hugely unbalanced dataset. For instance in NER, the no-entity tag usually makes up more than 99% of all entity tags. To evaluate model performance on such imbalanced data, it is usually advised against the usage of accuracy as evaluation metric, as it can lead to deceptively high scores. Therefore, NLP has adopted the use of F1 Scores.

However, even F1 scores may not be the best to evaluate NER models [4]. This is due to the fact that ordinary F1 scores work on a single token level, meaning that it is computed using the difference between true and predicted token for each separate token. This enables the metric to catch one of three errors:

I The true and predicted tag align (True Positive and True Negative)
II The model tags an entity when there is none (False Positive)
III The model misses an entity (False Negative)

This can be considered problematic, since these errors do not capture partial matches and scenarios in which the model correctly predicts the extent of the entity, but assigns the wrong tag to it. In fact, these are highly relevant to understand the performance of NER models, and may be captured as one of three additional errors:

IV The model assigns the wrong entity type
V The model gets the boundaries of the surface string wrong
VI The model gets the boundaries and entity type wrong

Since in our NER problem we only had a single entity that we wanted the model to find, errors IV and VI did not matter to us. However, we wanted to use an evaluation metric that is able to capture errors of type V, as well as gives us specific counts on the entities our model may miss, dream up, or correctly match. To this end, we used an open-source NER-Evaluation library implementing the above mentioned error metrics [3].

2.2 BERT and SciBERT

Since the introduction of deep learning models to NLP, virtually all previously used methods have been taken over. The field is moving fast, and new models appear every year. While not being the absolute state-of-the-art anymore, the industry standard at the point of writing is the Bidirectional Encoder Representations from Transformers (BERT) model, developed in 2018 [6]. Its widespread adoption made it an ideal candidate as the baseline model for our project. BERT is available as a pre-trained network, which can be fine-tuned for most NLP tasks, including NER.

While we initially experimented with BERT, we soon switched to an alternate version of it, SciBERT [5], which features the same model architecture, but has been pre-trained on scientific papers obtained from the Semantic Scholar corpus [1]. As our competition dataset featured scientific papers as input as well, this should make SciBERT the better model for our task.

Both BERT and SciBERT come in two sizes: a smaller version featuring 12 layers with 110M parameters, and a larger version with 340M parameters in 24 layers. Due to the already large amount of parameters in the smaller model and the limited computational capacity available to us, we opted for the smaller model. This is a reasonable decision for experimentation, since the tripling in parameters leads to only a marginal increase in performance on benchmark datasets, which for this project did not justify the additional time required for training the larger models.

Further, both models come in a 'cased' and 'uncased' version. The difference here is that the tokenisation done on the raw text inputs to transform them into suitable numbers that can be fed into a model, is either case-sensitive or case-agnostic. During our initial data analysis, we found the dataset labels exhibited unique capitalisation properties that could help the model discern them from other parts of the texts (see Appendix A.1). Therefore, we chose to use the cased version of all models.

When used for NER, the BERT architecture, including its SciBERT derivative, comes with the pre-trained model as a 'base model', on the last layer's embeddings of which a single layer linear classifier is stacked. Intuitively, this means that the base model extracts useful embeddings from the input data based on its pre-training knowledge, while the linear classifier transforms these embeddings into an actual entity type prediction.

The established training regimes either employ a full-finetuning of the base model together with the classifier, or freeze the weights of the base model and only train the classifier. We noted, however, that the SciBERT paper states that the full-finetuning regime usually outperforms the frozen regime, even when the frozen model is aided with a more complex BiLSTM instead of a single-layer linear classifier [5]. Thus, we opted for the full-finetuning approach in our experiments. We did, however, investigate an additional hybrid-approach for the SciBERT model, in which we only re-trained the last three layers of the base model. We reasoned that given the small amount of dataset names present in the train set (see section 2.2), the chance of overfitting on these names when using the full capacity of the BERT model was considerable. We proposed this configuration as a trade-off between being able to finetune more than just the final linear classifier and minimising the chance of overfitting.

As recommended in both the SciBERT and BERT papers, we employed a linearly decaying learning rate schedule, albeit without warmup [5, 6]. We ran an initial experiment with SciBERT using a learning rate of $2e{-5}$ and the Adam optimiser without weight decay, as was done in the original paper [5]. In all other experiments, we set the learning rate to $3e{-5}$ and used an Adam optimiser with a weight decay factor of 0.01 to improve regularisation of the weight matrices, as introduced in [11]. In all experiments, we set $\epsilon = 1e{-8}$ for the optimiser, and further clipped the gradient norm before each parameter update to avoid exploding gradients, with a maximum norm of 1.0.

2.3 Data Augmentation

In our data analysis we observed that many texts had the same dataset label. This meant that despite the fact that the dataset contained over 14,000 different texts, we only observed 130 unique dataset names in all of their labels.[1] This implies that the model is provided with ample examples of contexts in which dataset mentions might occur, but few different dataset names. As mentioned before, this would likely result in overfitting on the few dataset names that are in the training data.

The most prominent solution to this problem would be to add more data from a different source. However, finding a public dataset of scientific literature that has been similarly annotated with the dataset names mentioned within proved difficult. Since manually annotating new data would not be feasible given the scope of the project, we resorted to a type of data augmentation we will refer to as mention replacement. Introduced by Raiman and Miller [14] under the name of 'Type Swaps', this method was used to train a question answering system, where entities of a certain type (person, location, etc.) were swapped with entities of the same type from a knowledge bank. Because the positions of the entities in the text are known, no additional labelling is required, while additional entity mentions are introduced.

2.3.1 Implementation. We introduced mention replacement as a way to increase the variety of dataset names provided to the model in the following way. We extracted a knowledge base of 377 additional dataset names from the 'Awesome Public Datasets' repository by extracting the dataset names and cleaning them by removing hyperlinks and explanations [2].

During training, every sentence containing a dataset mention was randomly augmented prior to being fed to the model. With 50% chance, the original mention would be replaced by a random dataset name from the knowledge base. As the new name could contain more or less words than the original, the annotation labels, attention masks and padding where adapted automatically to correspond to the new sentence. If the new dataset name would cause the sentence to exceed the maximum length of the model, the original sentence would be fed to the model instead.

We note that by applying this augmentation, we are committing to the assumption that the specifics of the dataset name itself is independent of the context. In other words, the context only signals that there is a dataset mention, and does not hold any relationship to the name itself. This assumption might break down when dataset names from certain scientific fields are introduced in texts from a different field, as the dataset names often clearly relate to the topic of the field. However, the gain in data variety might outweigh the performance loss caused by this violated assumption.

2.4 Sampling Strategies

Due to the nature of the challenge and the annotations given in the corpus, the dataset mentions were quite sparse. Out of approximately 550K sentences, there were only 47K sentences with dataset mentions. As a result of this sparsity, the model performance may suffer. To deal with this problem, we employed weighted sampling with replacement, or over-sampling positive samples from the dataset. Over-sampling is the practice of showing the network duplicate dataset mentions to deal with class imbalance [13]. In our experiments we experimented with a weighted sampler, which prioritises dataset mentions with a probability based on the proportionality of positive and negative mentions in the dataset. In our experiment the probability from a set of samples that a dataset mention is sampled was set to 0.8.

2.5 Experiments

Experiments were conducted on the models, also in combination with aforementioned proposed optimisations implemented. The models were trained for 5-10 epochs based on train and validation loss convergence, after which the model checkpoint from the epoch with the best validation performance was submitted to be assigned a test score. For our best model, we submitted each model checkpoint for each epoch to be assigned a test score, in order to

[1]This number is optimistic, since the uniqueness here is based on exact matches. As an example, the 'unique' list of mentions still contains the following three mentions: 'cas covid 19 antiviral candidate compounds *data*', '(...) *data set*', and '(...) *dataset*', which are arguably too similar to be treated as unique dataset labels.

be able to compare the validation metrics and resulting test score. Finally, these test scores were compared, also taking into account the baseline score of string matching dataset names from the test set to compute the predictions (see [7]).

3 RESULTS

The results of our different model experiments are displayed in Table 1. The overall best performance was achieved by the SciBERT model with a learning rate of 3e−5, which obtained a submission score of 0.439. This score was reached at epoch 3, while model was continued to be trained for 10 epochs in total. We note that this model achieved a better score than its counterpart with a lower learning rate of 2e−5 and no weight decay, even though the latter configuration was recommended by the SciBERT authors [5]. We further recognise that the best model, which employs full finetuning of all base model weights, achieved a better score than its equivalent with only the last three layers being finetuned. The latter scored at 0.424 after 5 epochs of training. The SciBERT model with a learning rate of 3e−5 achieved better scores than the BERT model when trained for the same amount of epochs, respectively yielding a score of 0.438 and 0.432.

Model	# Epochs	Score
SciBERT + Data Aug	7	0.376
SciBERT + Oversampling	3	0.402
SciBERT + Data Aug	4	0.408
SciBERT 3 Layer Finetuning	5	0.424
SciBERT, LR = 2e−5, no weight decay	5	0.434
BERT	5	0.432
SciBERT	5	0.438
SciBERT	3	**0.439**
Literal matching baseline [7]	N/A	0.535

Table 1: Model results. Submission score was calculated as the micro $F_{0.5}$ score between predicted texts and ground truth texts [9]. All neural models were fully finetuned (i.e. no frozen weights) unless specified otherwise. LR is learning rate.

The SciBERT model using oversampling scored considerably worse than the model without it, yielding a score of 0.403.

For the SciBERT-based model trained using data augmentation, we observed a small decrease in performance, with the model trained for four epochs which had the best validation results, attaining a score of 0.408. This contradicts our expectation, given that the augmentation would allow the model to train on a wider variety of dataset names.

Finally, included the literal matching baseline result from Dixit [7]. This baseline simply applies string searching of all the datasets observed in the train set on the test set to retrieve the dataset mentions. We observe that this baseline results, scoring 0.535, clearly outperforms even our best model.

3.1 Best Model Analysis

We investigated the performance of the best overall model in more depth. Figure 1 shows the progression of the submission score at

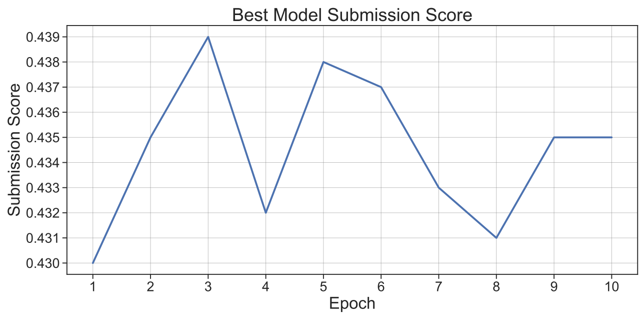

Figure 1: Final submission scores at each epoch of the best, fully finetuned SciBERT model.

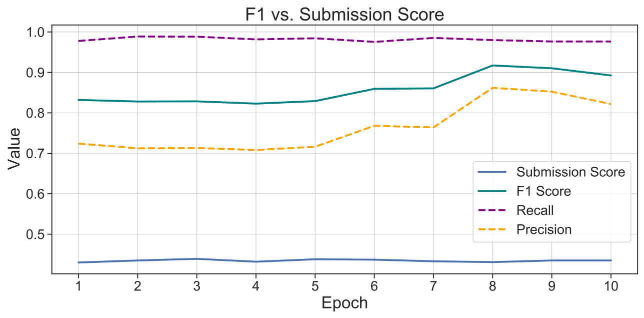

Figure 2: Precision, Recall, and F1 Scores for the best, fully finetuned SciBERT model plotted against the final submission score per epoch.

each of the 10 epochs. Note that after the best score was achieved at epoch 3, every following score was lower. However, the range of the score values seems to stay the same, and no convergent or divergent behaviour can be observed.

Figure 2 shows the same submission score over 10 epochs, plotted against the F1 score, precision and recall collected on the validation set at each epoch. It can be seen that the precision is notably increasing after 5 epochs, pushing the overall F1 up with it. However, there is no apparent correlation between the F1 score on the validation set, and the final submission score.

As described in section 2.1, we investigated different types of errors for this model, measured on the validation set at each epoch. The results can be seen in Figure 3. Please note the different y-axis extents for each of the error types. Overall, the model makes very few type V errors, meaning that when it finds in entity, it usually manages to correctly determine its boundaries. For all epochs, it only wrongly assigns between 6 and 18 boundaries. It makes more type III errors: between 137 (in epoch 2) and 319 (in epoch 6) out of 13,212 entities were missed, i.e. between 1.04% and 2.4%. However, most errors were made of type II. Here, the model predicts around 5000 entities in the first 5 epochs that are not actually present in the data (minimum 4919 in epoch 1 and maximum 5341 in epoch 4). A steep decline in this type of error can be noted for the remaining 5 epochs, with the lowest error count of 2,065 being achieved at epoch 8.

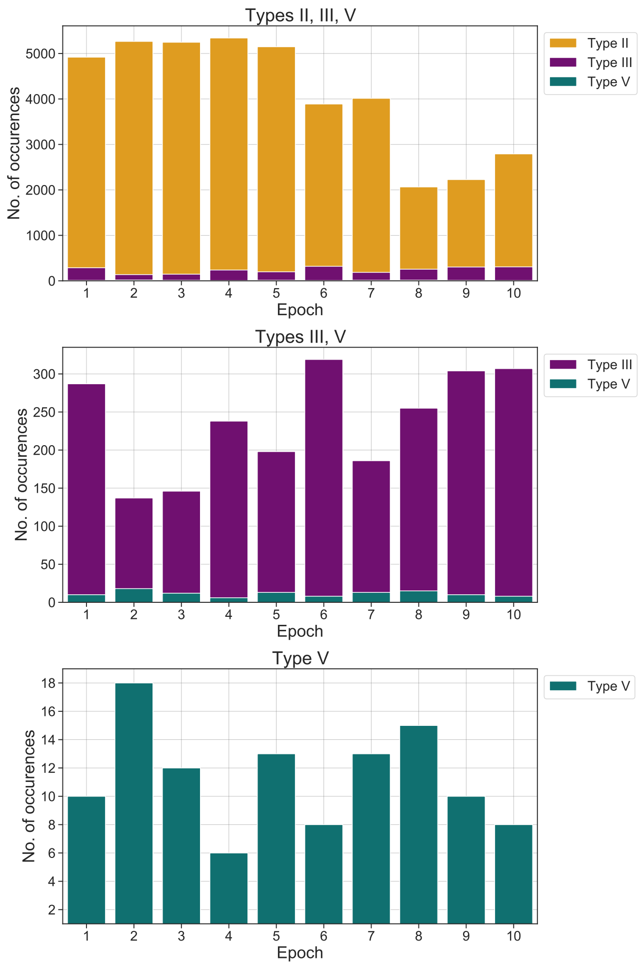

Figure 3: Type II, III, and V errors for the SciBERT model at each of 10 epochs. Note the different y-axis extents for the different error types.

Note that the amount of made up entities (type II error) directly influences the precision score, and the amount of entities missed (type III error) the recall score, respectively. This explains the sudden improvement in precision and F1 after 5 epochs as seen in Figure 2.

4 DISCUSSION

We observed that our best model was achieved with a different configuration than recommended by the original SciBERT authors, who didn't employ weight decay and recommended a learning rate of 2e−5 rather than 3e−5. However, this was not surprising: For one, weight decay is a technique that, due to its regularising capabilities, can bring an improved performance to large models in general. Secondly, the SciBERT authors noted that their choice of learning rate works best across most datasets, implying that for specific datasets a different learning rate may be found to perform better.

Additionally, we found that the SciBERT model with only the last 3 layers being finetuned achieved a considerably worse score (0.424) than the SciBERT models as well as the BERT model that all use full finetuning (each scoring above 0.43, excluding oversampling and augmentation models). This makes sense from the perspective that a fully-finetuned model has considerably more capability to tune itself specifically to the task at hand. However, it could also be reasoned that the SciBERT pre-training task, and thus its embeddings in earlier layers, would show sufficient similarity to our task that such a depreciation in score should not appear. We thus concluded that to really investigate this effect, one should fit several models with different numbers of layers being finetuned, and record performance at each epoch.

For the model with oversampling, we found a considerable decrease in submission score with respect to the model without oversampling. This was surprising, given that the performance on the validation set was adequate. The analysis of the failure of oversampling would therefore require an analysis of the performance on the test set, which was not available to us. Thus the exact reason for its failure remains to be investigated.

Data Augmentation. For the model trained with augmented data, we noted a decrease in performance. We discuss multiple possible causes to this performance decrease. First, we found that the way in which the data was labelled by our preprocessing pipeline caused issues for augmentation. An example original sentence would be '(...) ADNI data.', in which only the word 'ADNI' would be tagged as dataset since the text's label entry would read 'ADNI'. The mention would be swapped out with a dataset name from the knowledge base, for example, 'MIT Cancer Genomics Data'. This would result in the sentence '(...) MIT Cancer Genomics Data data.'. This resulted in sentences that would otherwise never occur in scientific writing, and training on such unrealistic data might have contributed to the degraded model performance.

Metrics. As presented in our results for the best model, we observed that the F1 scores did not reflect the final submission score. This was surprising to us, as we expected to see at least some kind of positive correlation, given the direct relation between the tagging performance of the model, and the final score.

However, it has to be noted that the most glaring mistake our best model made is to predict many more entities than are actually present. While it misses only around 1.5% of all entities in the validation set, it recognises several thousands of entities that do not actually exist, compared to 13,212 real ones. This explains the lacking final $F_{0.5}$ score, as this also punishes spurious predictions. We note that this may be an important shortcoming of these models in general, and should be subject to future investigations.

One problem with the validation data used to calculate the metrics is that there was overlap between the training and validation dataset labels. Because the dataset contained many more texts than unique dataset labels and a single text could contain multiple dataset labels, making a unique validation split with unseen labels proved non-trivial. This means that the validation metrics did not give a good insight in generalisation performance.

Generalisation Performance. Although establishing generalisation performance on the validation set alone was hard, we clearly observed a lower score than that of the literal matching baseline, which indicates generalisation might not be the biggest issue, given that simply remembering the training set would theoretically yield at least the baseline score. The question then is where the problem lies exactly.

We note that this might be due to the complexity of the problem. In code from other contestants, we observe how even models that focus more on context-awareness, such as Masked Language Models, have a poor test performance [8]. These notebooks often have a relatively high score, but literal matching aided with external datasets are the main cause of this. We want to emphasise that although the literal matching baseline performed better on this particular test set, it is a method that only works when there is overlap between the dataset mentions in the train and test set. When such a method would be applied in the field without feeding it new dataset labels, performance would decrease as new dataset names are used in the literature.

Model Choice. Furthermore, we note that BERT is not state-of-the-art anymore, and neither is SciBERT. For instance, the LUKE language model [16] achieved an F1 score of 94.3% on the CoNLL dataset [15], while BERT lies at 92.8%. However, we found that due to their widespread adoption, solid documentation, and reference performance metrics, BERT-based models provided a better starting point for us to get into NLP. It has to be further noted that the RoBERTa paper [10] showed that the original BERT model was severely undertrained, and that an optimised training procedure can cause BERT to achieve significant performance improvements. Therefore, we reason that BERT, or SciBERT for that matter, may be found to achieve better results than we could report, if researchers set out to optimise hyperparameter values and training conditions for the specific dataset at hand. Unfortunately, such investigations were out of the scope of this project.

5 CONCLUSION

In conclusion, we proposed a number of interesting optimisations to the SciBERT model in an attempt to improve upon the base model performance. The results show that none of the optimisations yield an improvement to the score. However, we observed how there is a unaccounted discrepancy between the test score and the validation results, which requires further investigation. In addition, because the test scores were all below baseline, it might be possible that optimisations like data augmentation do improve generalisation performance, but yield a lower score because a model that is tuned more to the train set might be able to attain a score closer the literal matching baseline. In addition, better state-of-the-art models could be employed in future work to investigate the test score gain.

6 AUTHOR CONTRIBUTIONS

Ganesh	Rood	Schröder
Balanced sampling	EDA	Input Pipeline
Submission Pipeline	Submission Pipeline	Adv. Metrics
Result analysis	Data Cleaning	SciBERT Analysis
-	Data Augmentation	-
Report	Report	Report

7 EVALUATION OF THE PROCESS

In the second project of this course, we planned to do less, but more complex experiments with the model than in the first project of the course. We noted how, given the unfamiliarity with and complexity of NER, it took us (and other groups) a lot of time and effort to set up a working baseline. Despite this, we were able to conduct some interesting experiments yielding valuable results. The collaboration went smoother than the first project, having learned from our first experience with Kaggle, we organised a better code base and version control to keep everything organised.

8 EVALUATION OF THE SUPERVISION

We felt like there was less support from supervision during the second project than during the first project. The fact that it took a long time to get the baseline working likely contributes to this feeling, as this meant that the first meetings were mostly discussing implementations progress and difficulties, while during the experimentation phase (after the flash talks), there were little meetings to discuss experiment setups and results.

REFERENCES

[1] Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, Rodney Kinney, Sebastian Kohlmeier, Kyle Lo, Tyler Murray, Hsu-Han Ooi, Matthew Peters, Joanna Power, Sam Skjonsberg, Lucy Wang, Chris Wilhelm, Zheng Yuan, Madeleine van Zuylen, and Oren Etzioni. 2018. Construction of the Literature Graph in Semantic Scholar. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*. Association for Computational Linguistics, New Orleans - Louisiana, 84–91. https://doi.org/10.18653/v1/N18-3011
[2] AwesomeData and 157 contributors. 2021. *Awesome Public Datasets*. https://github.com/awesomedata/awesome-public-datasets/tree/20130478980e72e99e3feeae85c855bcba95e1ae (Accessed on 2021-06-09).
[3] David S. Basista, Julien Rossi, and ivyleavedtoadflax. 2021. *Named Entity Evaluation as in SemEval 2013 task 9.1.* https://github.com/davidsbatista/NER-Evaluation (Accessed on 2021-06-09).
[4] David Batista. 2018. *Named-Entity evaluation metrics based on entity-level.* http://www.davidsbatista.net/blog/2018/05/09/Named_Entity_Evaluation/ (Accessed on 2021-06-10).
[5] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3615–3620. https://doi.org/10.18653/v1/D19-1371
[6] 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)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423
[7] Prashans Dixit. 2021. *Coleridge Initiative-EDA & Baseline Model*. https://www.kaggle.com/prashansdixit/coleridge-initiative-eda-baseline-model#5.-Baseline-model-and-Submission%F0%9F%93%9 (Accessed on 2021-06-09).
[8] Chien-Hsiang Hung. 2021. *additional_datasets + MLMv4*. https://www.kaggle.com/chienhsianghung/additional-datasets-mlmv4 (Accessed on 2021-06-09).
[9] Coleridge Initiative. 2021. *Show US the Data: Evaluation*. https://www.kaggle.com/c/coleridgeinitiative-show-us-the-data/overview/evaluation (Accessed on 2021-06-09).
[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).
[11] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).
[12] Brigitte Mathiak and Katarina Boland. 2015. Challenges in matching dataset citation strings to datasets in social science. *D-Lib Magazine* 21, 1/2 (2015), 23–28.
[13] Cristian Padurariu and Mihaela Elena Breaban. 2019. Dealing with Data Imbalance in Text Classification. *Procedia Computer Science* 159 (2019), 736–745.

https://doi.org/10.1016/j.procs.2019.09.229 Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Conference KES2019.

[14] Jonathan Raiman and John Miller. 2017. Globally Normalized Reader. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 1059–1069. https://doi.org/10.18653/v1/D17-1111

[15] Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 142–147. https://www.aclweb.org/anthology/W03-0419

[16] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 6442–6454. https://doi.org/10.18653/v1/2020.emnlp-main.523

A APPENDIX

A.1 Exploratory Data Analysis

We performed an exploratory data analysis, mainly focusing on the features of the dataset names, as well as some general statistics of the dataset. The first thing we noted when inspecting the data was how many dataset names consist of words that are all starting with a capital letter. This is of course the result of the dataset name being a title, and we speculated that this information could be used by the model when extracting the dataset mentions.

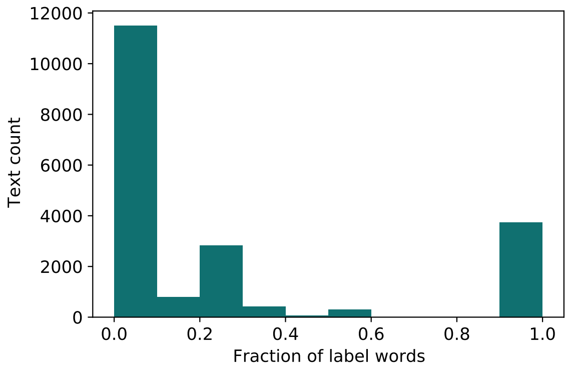

Figure 4: Fraction of dataset label words starting with a capital letter, formatted as a histogram showing the number of texts per fraction.

These initial observations showed to hold ground in the statistics of the data. As shown in Figure 4, on average over four out of five words in the dataset title would contain a leading capital. We can clearly see how this distribution is skewed to the left, with the most common observation of all words starting with a leading capital.

Additionally, we found that there are also quite some titles that contain fully capitalised words (see Figure 5). This is to be expected, since dataset names are often accompanied with some abbreviations or acronym to denote them shortly (e.g. ADNI for the Alzheimer's Disease Neuroimaging Initiative).

This led to us choosing a cased pre-trained model as our basis, which means that the model was pre-trained on data that was not lowercased. The intuition would be that if capitalisation properties of the dataset names are a discerning feature, a model that can actually detect these differences would be preferred.

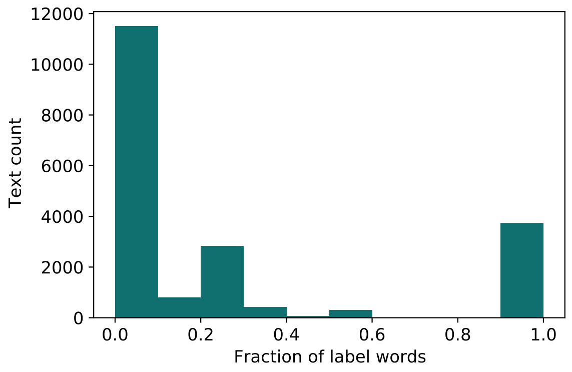

Figure 5: Fraction of dataset label words that are fully capitalised, formatted as a histogram showing the number of texts per fraction.

In addition to this conclusion, we found additional properties of the data which were important to take into account while preparing the data for training. First off, we found that literal searching of a reported dataset label in the corresponding text would not always result in finding any dataset mention. When doing this search case insensitive, more mentions are found, but we found that still 3.2% of the texts will end up with 0 mentions of the dataset label in the text (see Figure 6). This is important, since in our proposed pipeline, we use literal search to find the dataset mentions in order to tag them word for word with entity tags needed for the (SciBERT) model. With this pipeline, these texts would not be annotated and would wrongly punish the model for predicting a dataset label, even if it was correct. Therefore, we opted to remove these texts from our training procedure, as we did not see an easy and reliable substitute method for automatically tagging the dataset mentions, and it only concerned a small part of the training set.

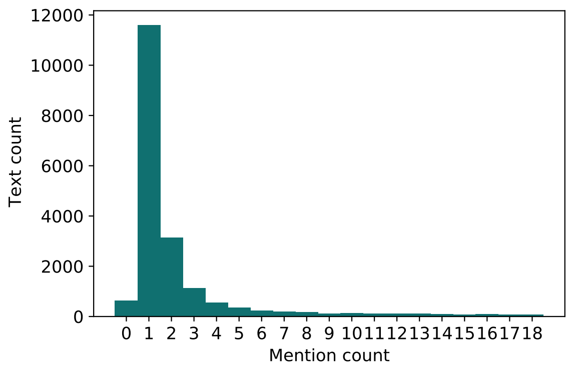

Figure 6: Number of times the dataset label is mentioned in the text (as determined by case-insensitive matching).

Finally, we investigated the length of the texts in the training set. We found that the word count of the texts is distribution with 99.2% of its mass below 70,000 words, and a long tail to larger values (see Figure 7). However, the largest text in the dataset contains an

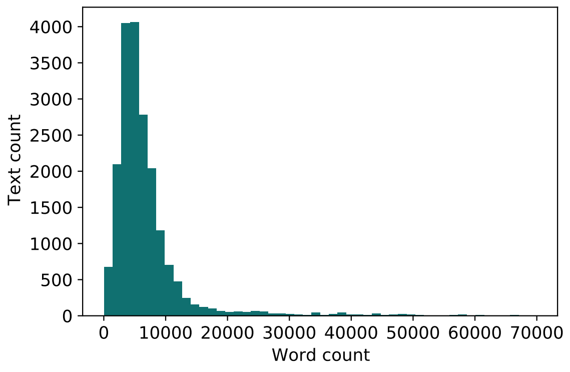

Figure 7: Word count in dataset texts. Note that 0.8% of the data lies beyond 70,000.

impressive 1.8 million words. When inspecting these large texts we found that they are not scientific articles, but rather complete books. Since the format of these books seemed to significantly differ from the rest of the dataset, and would take up a disproportionate amount of time in training, whilst being less clean, we chose to remove the 0.8% of texts that had a word count above 70,000.

A.2 Code

All training/submission notebooks and supplementary code is publicly available at https://github.com/verrannt/show-us-the-data.