Counts@IITK at SemEval-2021 Task 8: SciBERT Based Entity And Semantic Relation Extraction For Scientific Data

Akash Gangwar* **Sabhay Jain*** **Shubham Sourav*** **Ashutosh Modi**
Indian Institute of Technology Kanpur (IIT Kanpur)
{akashgnr, sabhayj, ssourav}@iitk.ac.in
ashutoshm@cse.iitk.ac.in

Abstract

This paper presents the system for SemEval 2021 Task 8 (MeasEval). MeasEval is a novel span extraction, classification, and relation extraction task focused on finding quantities, attributes of these quantities, and additional information, including the related measured entities, properties, and measurement contexts. Our submitted system, which placed fifth (team rank) on the leaderboard, consisted of SciBERT with [CLS] token embedding and CRF layer on top. We were also placed first in Quantity (tied) and Unit subtasks, second in MeasuredEntity, Modifier and Qualifies subtasks, and third in Qualifier subtask.

1 Introduction

SemEval 2021 Task 8 (Harper et al. 2021) is a task for extracting entities and semantic relations between them from a corpus of scientific articles coming from different domains. Instead of just identifying quantities, the task gives more weightage to parsing and extracting important semantic relations among the extracted entities. This is challenging because texts are ambiguous, and inconsistent, and extraction relies heavily on implicit knowledge. The results of this task can also be used for extractive scientific data summarization.

Given a scientific text, the task is to identify the span of quantities, units, and other attributes of those quantities and related measured entities, properties, and qualifiers, if any. The organizers have divided the task into five subtasks and submissions will be evaluated against all five sub-tasks[1].

1. **Quantity Extraction**: For each paragraph of text, identify all Quantity spans.

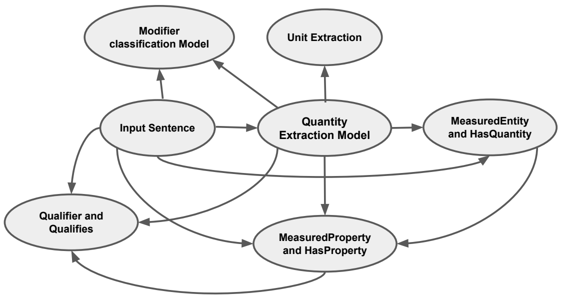
Modifier classification Model
Unit Extraction
Input Sentence
Quantity Extraction Model
MeasuredEntity and HasQuantity
Qualifier and Qualifies
MeasuredProperty and HasProperty

Figure 1: Overview of our proposed approach

2. **Unit Detection and Modifier Classification**: For each identified Quantity, identify the Unit of measurement and classify additional value Modifiers (count, range, approximate, mean, etc.) that apply to the Quantity.
3. **MeasuredEntity and MeasuredProperty Extraction**: For each identified Quantity, identify the MeasuredEntity and MeasuredProperty associated with it.
4. **Qualifier Extraction**: Identify and mark the span of any Qualifier that is needed to record additional related context.
5. **HasQuantity, HasProperty and Qualifies Extraction**: Identify relationships between Quantity, MeasureEntity, MeasuredProperty, and Qualifier.

We consider subtask 1 as an entity extraction task, and subtask 3, 4, and 5 are viewed as relation extraction tasks. After extracting the quantities, other attributes (MeasuredEntity, Property, and Qualifier) related to those quantities need to be predicted. The directed graph in Figure 1 gives an overview of our proposed approach. The set of incoming edges to each node represents the input to the trained model (represented by node), and the label at each node represents the prediction made by the model. The task data is extracted from CC-BY ScienceDirect Articles and made available by

* Authors equally contributed to this work.
[1] https://competitions.codalab.org/competitions/25770

the Elsevier Labs via the OA-STM-Corpus[2]. This motivated the use of SciBERT (Beltagy et al. 2019) model for various subtasks. “SciBERT leverages unsupervised pretraining on a large multi-domain corpus of scientific publications to improve performance on downstream scientific NLP tasks”.

Our final submitted system consisted of SciBERT with [CLS] token embedding and CRF layer on top, and it achieved an overall F1-overlap score of 0.432. We were ranked fifth on the global leaderboard. The top performance on the leaderboard achieved an overall F1-overlap of 0.519. The implementation of our system is made available via Github[3].

The rest of this paper is arranged as follows. Section 2 introduces the previous work in this field and describes the organizers' dataset. Section 3 explains our overall approach. Section 4 contains the experimental setup for training the model. We conclude with the analysis of our model performance in section 5 and concluding remarks in section 6.

2 Background

2.1 Related work

Magge et al. 2018 attempted to recognize the Clinical Entities using a LSTM CRF based architecture. The authors used the word and character level embedding obtained from word2vec (Mikolov et al. 2013). For relation extraction between these entities, authors build a binary classifier using random forest classifier. This approach has higher time complexity as it checks for all possible relationships that could exist and classifies them. The more recent work in entity extraction is by Lee et al. 2019, where they fine-tuned the BERT model using the Bio-Medical data, and have shown SOTA performance. Some other works in entity extraction includes Taher et al. 2020, where they fine-tuned BERT followed by a fully connected layer and a CRF layer.

The work by Wu and He 2019 on Relation Extraction uses BERT to identify the different types of relations between pair of entities in the given text. The system does not automatically recognize the entities between which relation exists, rather entities of interest need to be manually specified.

[2]https://github.com/elsevierlabs/OA-STM-Corpus
[3]https://github.com/akashgnr31/Counts-And-Measurement

2.2 Task setup

The scientific articles in the training and test corpus are from the following sub-domains: Astronomy, Engineering, Medicine, Materials Science, Biology, Chemistry, Agriculture, Earth Science, and Computer Science. These articles were manually annotated. The inter-annotator agreements was calculated using Krippendorff's Alpha IAA score (Table 1).

Class	IAA Score
Quantity	0.943
MeasuredEntity	0.640
MeasuredProperty	0.545
Qualifier	0.333
Units	0.866

Table 1: IAA scores of various classes

The training dataset comprised of 298 paragraphs containing 1164 quantities, 1148 measured entities, 742 measured properties, and 309 qualifiers. The evaluation set included 135 paragraphs.

3 System overview

3.1 Pre Processing

Since we are using the SciBERT model, a maximum of 512 tokens can be passed as input to the model. Therefore, we used SciSpaCy (Neumann et al. 2019) to split the paragraph into sentences, and these sentences were passed as input to the SciBERT model.

3.2 Subtask 1 (Quantity Extraction)

Input sentences were tokenized using a SciBERT tokenizer from HuggingFace (Wolf et al. 2020) implementation. The Quantity span were transformed into BIO / IOB format (Ramshaw and Marcus 1995) and used as the true-labels for training the model.

The tokenized sentence is passed through SciBERT. Tanh activation function is applied over the final hidden state of SciBERT i.e.

$$H_i^{'} = W_1[tanh(H_i)] + b_1 \quad i = 0, 1, ..., len$$

Here H_i is the hidden units corresponding to token i and len is the maximum length of the tokenized sentence. Similarly, [CLS] token is processed.

$$H_{cls}^{'} = W_0[tanh(H_0)] + b_0$$

Finally, we get the final representation for the sentence by concatenating H'_{cls} and H'_i and this is used for prediction via the softmax.

$$H''_i = W_2[concat(H'_i, H'_{cls})] + b_2 \quad i = 0, 1, ..., len$$
$$H'' = \left[H''_0, H''_1, ..., H''_{len}\right]^T$$
$$p = softmax(H'', \ dim = -1)$$

Matrices W_0 and W_1 have same dimension, i.e., $W_0 \in R^{d\times d}, W_1 \in R^{d\times d}, W_2 \in R^{t\times 2d}$, where d is the hidden state size from BERT and t represent the number of tags, i.e., t = 3 in our case as we are using BIO encoding..

CRF (Conditional Random Field) (Lafferty et al. 2001) is a probabilistic model that makes it possible to extract structural dependencies among the BIO tags. The tag probability vector for all the tokens, i.e., p, is passed through the CRF layer to generate the most probable output sequence.

We trained the model using CRF loss and Adam optimizer. The overall architecture of the model is shown in Figure 2. The tuned hyper-parameters are reported in appendix A.1.

3.3 Subtask 2 (Unit Detection)

The Quantity phrases are tokenized using Spacy (Honnibal et al. 2020) character-based tokenizer. The true-label for training is formatted as a binary vector marking one at the indices for characters in the unit's span in the Quantity phrases.

We trained a Character-based Bi-LSTM (Hochreiter and Schmidhuber 1997) model with trainable word embeddings using BCE (Binary Cross Entropy) loss and Adam optimizer. The model architecture and tuned hyper-parameters are reported in appendix A.2

3.4 Subtask 2 (Modifier Classification)

We formulated this subtask as a multi-label classification problem with 12 labels (HasTolerance, IsApproximate, IsCount, IsList, IsMean, IsMeanHasSD, IsMeanHasTolerance, IsMeanIsRange, IsMedian, IsRange, IsRangeHasTolerance, None). To enable the BERT module to capture the location of a quantity, we insert the special symbol "$" at the beginning and end of the Quantity span. If there are multiple Quantities in a sentence, multiple copies of the same sentence are generated with "$" at different positions. Suppose H_i to H_j are the final hidden state vector for the Quantity span. Then, the average operation is applied to get the vector representation of the Quantity. The averaged output is passed through a fully connected layer followed by softmax activation.

$$H'_q = W\left[tanh\left(\frac{1}{j-i+1}\sum_{k=i}^{j} H_k\right)\right] + b$$
$$p_q = sigmoid(H'_q)$$

Matrix W has dimension $R^{l\times d}$, where l represnts the number of classification label, i.e., l = 12 in our case and d is the hidden state size from BERT.

The above model was trained using BCE (Binary Cross Entropy) and Adam optimizer. The threshold value for prediction was determined using cross-validation. The model architecture and tuned hyper-parameters are reported in appendix A.3.

3.5 Subtask 3 and 5 (MeasuredEntity and HasQuantity Extraction)

As done in the previous subtask to capture the location, we insert the special symbol "$" at the beginning and end of the quantity span. The modified sentences are tokenized using a SciBERT tokenizer. The span of the MeasuredEntity related to Quantity enclosed in the "$" symbol is transformed into BIO / IOB format and used as the true-label for training the model.

The formatted data is used to train a model similar to the Quantity Extraction (SciBERT + CRF Model). The above model extracts the MeasuredEntity associated with the Quantity enclosed in "$". Thus, it predicts the MeasuredEntity as well as the HasQuantity relationship of the predicted MeasuredEntity.

3.6 Subtask 3 and 5 (MeasuredProperty and HasProperty Extraction)

To extract MeasuredProperty and HasProperty relationship, we used a similar approach as used for MeasuredEntity and HasQuantity. We enclosed the Quantity span in "$" symbol and the MeasuredEntity span in "#" symbol. The modified sentences are passed through the SciBERT tokenizer. The span of MeasuredProperty related to MeasuredEntity, Quantity pair is transformed into BIO / IOB format and used as the true-label for training the model.

The formatted data is used to train a model similar to the Quantity Extraction (SciBERT + CRF Model). The model trained is used to extract MeasuredProperty linked with the MeasuredEntity, Quantity pair. If the above model predicts

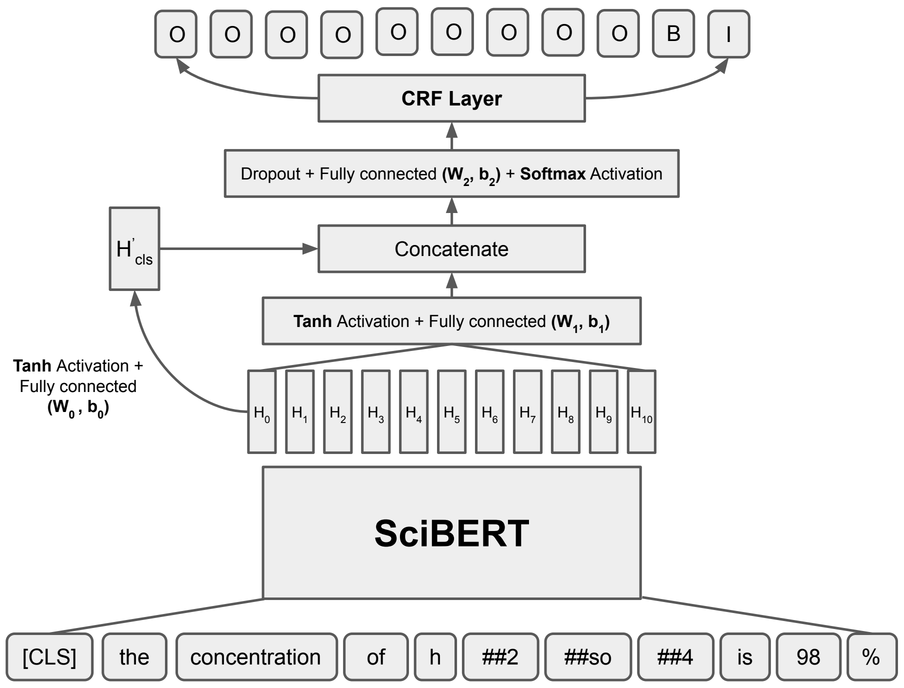
O O O O O O O O O B I
CRF Layer
Dropout + Fully connected (W2, b2) + Softmax Activation
H'cls
Concatenate
Tanh Activation + Fully connected (W1, b1)
Tanh Activation + Fully connected (W0, b0)
H0 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10
SciBERT
[CLS] the concentration of h ##2 ##so ##4 is 98 %

Figure 2: SciBERT with [CLS] token embedding and CRF layer on top (SciBERT + CRF Model)

Measured-Property's span, then the HasQuantity relation is updated to MeasuredProperty, and the HasProperty relation is added to MeasuredEntity.

3.7 Subtask 4 and 5 (Qualifier and Qualifies Extraction)

To extract Qualifier and Qualifies's span, two separate models similar to Quantity Extraction (SciBERT + CRF Model) were trained. While training the first model, we insert "$" at the beginning and end of the Quantity span because we assumed that Qualifier Qualifies Quantity. During the second model training, we enclosed the MeasuredProperty span in "$" because of the assumption that Qualifier Qualifies MeasuredProperty.

3.8 Post Processing

Once the predictions from all the models are available, we need to transform the predicted BIO/ IOB format into entity span format. We initially map each token's span in the tokenized sentence and use it to determine the predicted entity's span. While finding the span of the MeasuredEntity, MeasuredProperty, or Qualifier, if our model predicts multiple entities, then we predict the one which is closest to the Quantity span. After that, we convert the sentence span of each entity extracted to the paragraph span.

4 Experimental Setup

The dataset is split into two parts - train set and dev set in a ratio of 90:10. The models were trained on the train set and were validated on the dev set. The environment and packages used for training and pre-processing are listed in appendix B.

4.1 Evaluation Metrics

The official metrics used by the SemEval organizer are F1-measure, F1-overlap, and Exact Match. Exact Match is a binary value of 0 or 1, while F1-measure is a token level overlap ratio of submission to true spans, where tokenization is done using simple white space delimiters. F1-overlap is a SQuAD (Rajpurkar et al., 2016) style Overlap score based on F1-measure, which penalizes the negative submissions more strictly. The final evaluation is based on a global F1-overlap score averaged across all subtasks.

5 Results

5.1 Model Variants Used

We tried various models like BERT-Base, BERT-Medium (Devlin et al., 2018), SciBERT, and BioBERT (Lee et al. 2019). We could not try BERT-Large due to computational limitations. The results for the top two models are shown in Table 2.

We also experimented with Bi-LSTM layers on top of BERT, but the model was overfitting due

Model	Data Set	Quantity	Unit	Modifier	MeasuredEntity	MeasuredProperty
SciBERT	eval	0.861	0.804	0.614	0.406	0.245
SciBERT	dev	0.887	0.744	0.696	0.322	0.216
BERT-Med.	eval	0.791	0.675	0.379	0.302	0.163

Table 2.A

Model	Data Set	Qualifier	HasQuantity	HasProperty	Qualifies	Overall F1 overlap
SciBERT	eval	0.077	0.311	0.183	0.064	**0.432**
SciBERT	dev	0.083	0.270	0.137	0.083	0.410
BERT-Med.	eval	0.0	0.193	0.114	0.0	0.330

Table 2.B

Table 2: Table 2.A represents the F1-overlap score for subtask 1, 2, 3, and Table 2.B represents the F1-overlap score for subtask 4, 5 and overall F1-overlap

Metric	SciBERT + CRF	Base line
Precision	0.703	-
Recall	0.560	-
F-Measure	0.623	-
F1-overlap	0.432	0.239
Exact Match	0.371	0.211

Table 3: Overall Results on evaluation set

to its high complexity. Consequently, it was not included in the final model.

5.2 Results on evaluation set

The results achieved on the evaluation set for each subtask are shown in Table 2, and the overall results are shown in Table 3. Figure 3 represents the results achieved in the various subdomains.

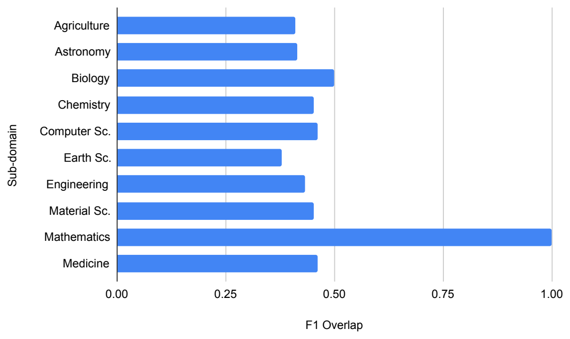

Figure 3: F1-Overlap scores of various sub-domains

The difference between the Exact Match score and F1-overlap score shows that the spans predicted by our model were precisely the same as gold data whenever they matched.

We achieved an overall fifth rank (among 19 participating teams) in the competition. We were also placed first in Quantity (tied) and Unit subtasks, second in MeasuredEntity, Modifier and Qualifies subtasks, and third in Qualifier subtask.

5.3 Error Analysis

The relation extraction subtask was challenging because associating entities with the quantities they are related to is context-dependent and based on one's understanding. This is also evident from the IAA scores reported for the train data that even humans can achieve deficient performance.

Some of the aspects where our model did not work well are:

1. Our model looks for relations only within a sentence, which may cause problems when a relation exists outside the same sentence.
2. There is loss in reconstructing the TSV files from entities because the neighboring data may/maynot be part of the same entity group
3. Our model didn't work well on MeasuredProperty and Qualifiers as it did on other subtasks, which is evident as we achieved only 0.53 and 0.35 F1-overlap on training data for these two subtasks.

6 Conclusion

This paper proposed SciBERT + CRF Model (SciBERT with [CLS] token embedding and CRF layer on top) for span extraction, classification, and semantic relation extraction. Our model shows significant improvement in performance over the baseline model and works equally well across all the scientific sub-domains. In the future, we plan to explore various other pre-trained contextual models for our approach.

References

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. SciBERT: Pretrained Contextualized Embeddings for Scientific Text. *CoRR*, abs/1903.10676.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, abs/1810.04805.

Corey Harper, Jessica Cox, Curt Kohler, Antony Scerri, Ron Daniel Jr., and Paul Groth. 2021. SemEval 2021 task 8: MeasEval – extracting counts and measurements and their related contexts. In *Proceedings of the Fifteenth Workshop on Semantic Evaluation (SemEval-2021)*, Bangkok, Thailand (online). Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Arjun Magge, Matthew Scotch, and Graciela Gonzalez-Hernandez. 2018. Clinical NER and Relation Extraction using Bi-Char-LSTMs and Random Forest Classifiers. volume 90 of *Proceedings of Machine Learning Research*, pages 25–30. PMLR.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text.

Lance Ramshaw and Mitch Marcus. 1995. Text Chunking using Transformation-Based Learning. In *Third Workshop on Very Large Corpora*.

Ehsan Taher, Seyed Abbas Hoseini, and Mehrnoush Shamsfard. 2020. Beheshti-NER: Persian Named Entity Recognition Using BERT. *CoRR*, abs/2003.08875.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shanchan Wu and Yifan He. 2019. Enriching Pre-trained Language Model with Entity Information for Relation Classification.

Appendix

A Model Training

A.1 SciBERT + CRF

In this section we provide hyper-parameter values (Table 4) we used for training our final model to facilitate reproduciblity of our results.

Hyper-parameters	Value
Hidden State Dimension (d)	768
Number of tags (t)	3
Dropout	0.1
Batch Size	24
Max Length (len)	255
Learning Rate	10^{-5}

Table 4: Hyper-parameters

A.2 Unit Detection (Character-based Bi-LSTM)

In this section we provide model architecture (Figure 4) and hyper-parameter values (Table 5) we used for training our final unit extraction model to facilitate reproduciblity of our results.

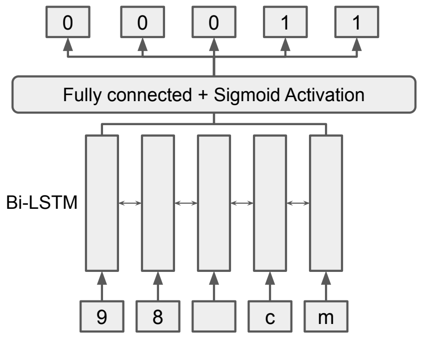

Figure 4: Character-based Bi-LSTM

Hyper-parameters	Value
Hidden State Dimension of Bi-LSTM	32
Number of Bi-LSTM layers	1
Batch Size	38
Max Length (len)	64
Learning Rate	10^{-4}

Table 5: Hyper-parameters

A.3 Modifier Classification (SciBERT with embedding averaging)

In this section we provide model architecture (Figure 5) and hyper-parameter values (Table 6) we used for training our modifier classification final model to facilitate reproduciblity of our results.

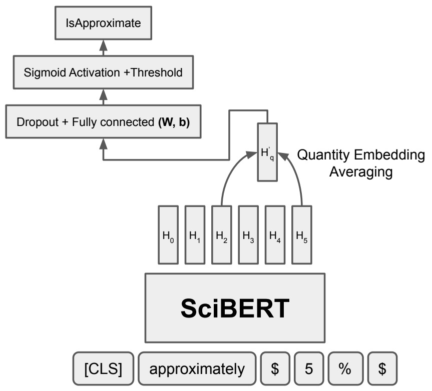
IsApproximate
Sigmoid Activation +Threshold
Dropout + Fully connected (W, b)
H'q
Quantity Embedding Averaging
H0 H1 H2 H3 H4 H5
SciBERT
[CLS]
approximately
$
5
%
$

Figure 5: SciBERT with embedding averaging

Hyper-parameters	Value
Hidden State Dimension (d)	768
Number of labels (l)	12
Dropout	0.1
Batch Size	24
Max Length (len)	255
Learning Rate	10^{-5}
Threshold	0.5

Table 6: Hyper-parameters

B Tools/Libraries used

We used Google Colab Nvidia T4 GPU (16GB) for training purpose. Python packages (alongwith version) used for pre-processing and training are tabulated below:

Package	Version
transformers	4.3.2
torchcrf	0.7.2
torch	1.7.0
scikit-learn	0.22.2
en_core_sci_sm	0.3.0
Stanza	1.2
spaCy	2.3.5
NLTK	3.2.5
pandas	1.1.5
NumPy	1.19.5

Table 7: Python Packages