# KGP at SemEval-2021 Task 8: Leveraging Multi-Staged Language Models for Extracting Measurements, their Attributes and Relations

**Neel Karia***, **Ayush Kaushal***, **Faraaz Rahman Mallick***
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
{karianeel, ayushkaushal, faraazrm}@iitkgp.ac.in

## Abstract

SemEval-2021 Task 8: MeasEval aims at improving the machine understanding of measurements in scientific texts through a set of entity and semantic relation extraction sub-tasks on identifying quantity spans along with various attributes and relationships. This paper describes our system, consisting of a three-stage pipeline, that leverages pre-trained language models to extract the quantity spans in the text, followed by intelligent templates to identify units and modifiers. Finally, it identifies the quantity attributes and their relations using language models boosted with a feature re-using hierarchical architecture and multi-task learning. Our submission significantly outperforms the baseline, with the best model from the post-evaluation phase delivering more than 100% increase on F1 (Overall) from the baseline.

## 1 Introduction

Most scientific experiments are accompanied by relevant measurements, which help researchers to quantify their observations and qualitative arguments. Measurements also play a pivotal role in summarizing large experiments, and provide a brief idea of the results obtained. It is customary for scientists to present their research in the form of scientific papers. Nowadays, with thousands of papers being published digitally every year, it is extremely difficult to go through every single paper in order to get the desired data. The most popular electronic open-access repository of e-prints, arXiv, currently has 1,867,929 articles[1]. The sheer vastness of this number suggests just how important it is for us to automate the task of extracting measurement-related information from research papers (Singh et al., 2016).

A thorough understanding of the measurements not only requires the numerals, but also the context in which the quantities occur. Moreover, the entities and the properties measured along with the qualifiers that condition the measurements are crucial for understanding the measurement. MeasEval (Harper et al., 2021) is a semantic relation extraction task focused on obtaining 9 different entities pertaining to counts, measurements and qualifying attributes of these quantities in a collection of excerpts from research papers in English. Figure 1 shows an example of a quantity along with its attributes and relations from this dataset.

We propose a three-stage pipeline to address this task. The first stage uses a pre-trained BERT model (Devlin et al., 2019) to detect quantity spans from sentences. Receiving the detected spans as inputs, the second stage obtains the units and modifiers using extracted units and modifier keywords. Finally, the third stage receives the quantity spans from the first stage and uses another pre-trained language model over each quantity-span-conditioned sentence to obtain quantity-span-aware contextualized representations for each sub-token in the sentence. These representations are then used to detect the measured entity corresponding to each quantity (if any). The predictions from the measured entity task are then fused with the individual representations for each sub-token. These representations are used to detect the measured property and the qualifiers in a multi-task learning setting (Ruder, 2017).

Our submission surpassed the baseline by a significant margin and ranked $3^{rd}$ for the Unit task. Our current best model delivers $516.7\%, 436.8\%$, and $296.4\%$ F1 (Overlap) (Mei and Radev, 1979) gains for Measured Entity, Measured Property and Qualifier tasks respectively, over the baseline.

*Equal contribution.
[1]as of $9^{th}$ April, 2021

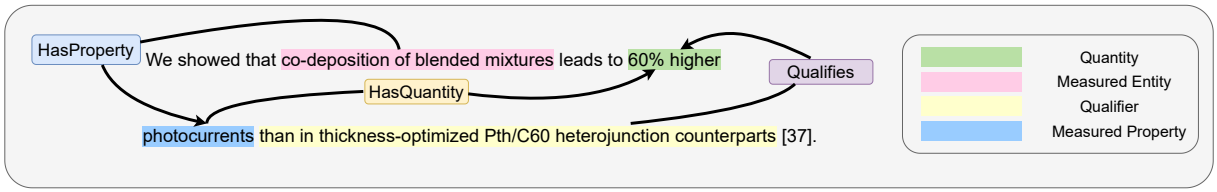


Figure 1: Visualization of Annotated Dataset

## 2 Related Works

Understanding and extracting information from scientific documents has been receiving increasing interest (Tsai et al., 2006; Nadeau and Sekine, 2007). Extracting units of measurement from scientific documents was previously studied via regular expressions and supervised classifiers (Berrahou et al., 2013; Sevenster et al., 2015).

In the orthogonal directional, there has been rapid progress in understanding natural language using deep pre-trained language models (Peters et al., 2018; Devlin et al., 2019; Liu et al., 2020; Yang et al., 2019), which has lead to a general improvement across multiple tasks. The sequence labelling (Lample et al., 2016; Panchendrarajan and Amaresan, 2018) and span prediction (Luo et al., 2020; Pang et al., 2019) tasks for natural language have also received great interest recently. We build upon these systems.

## 3 Problem Statement

We are given a set of **documents** $D = \{(d_i)_{i=1}^{n}\}$. Every document $d_i \in D$, consists of various **Quantity** (Q) spans $Q_i = \{(q_i^j)_{j=1}^{m}\}$. Every $q_i^j \in Q_i$, can have a **Unit** of measurement (e.g. $cm$, $ml$) associated with it. Also every $q_i^j \in Q_i$ is associated to some (or no) **Modifiers** (Mod) which provide information about the type of Q (e.g. whether it is a range of values, whether it denotes the Median of a set of values, etc.)[2]. For every $q_i^j \in Q_i$, there can exist a corresponding **Measured Entity** (ME) $e_i^j$. Some Qs do not have any ME, e.g. in '3413 women', the measurement is 3413 and *'women'* is the 'unit' of 3413 and **not** its ME (according to "S0006322312001096-1177.tsv"). Similarly in 'three occasions', the measurement is *'three'* and *'occasions'* is its 'unit' and not its ME (according to "S0165587612003680-1078.tsv"). If a $q_i^j$ has a corresponding ME $e_i^j$, it can also have an associated **Measured Property** (MP) $p_i^j$. Finally, the Qs, MEs and MPs can have a number of **Qualifiers** (Qual) $qual_i^j$ providing additional information about them.

The problem also introduces three relations, namely **Qualifies** (QS), **HasProperty** (HP), and **HasQuantity** (HQ). These relations are defined between Qs, MEs, MPs and Quals as binary classification functions ($f(x, y) \rightarrow (0, 1)$):

- $QS(x, a) = 1, \iff$ the Qual, $a$, *qualifies* the element $x$, where $x$ is a Q, ME or MP.
- $HP(p, e) = 1, \iff$ the MP, $p$, is *associated* with the ME, $e$.
- $HQ(y, q) = 1, \iff$ the Q, $q$, is *related* to element $y$, where $y$ is an ME or MP.

The problem statement consists of 5 sub-tasks. We deal with identifying all Q spans in the documents in sub-task 1, followed by detecting the Units and Mods for each identified Q in sub-task 2. In sub-tasks 3 and 4, we identify the ME, MP, and Qual spans, corresponding to the extracted Qs. Finally in sub-task 5, we identify the relationships HQ, HP, and QS between the detected Q, ME, MP, and Qual spans. Figure 1 shows the annotation procedure to be followed (Stenetorp et al., 2012).

## 4 System Overview

We model all the previously described sub-tasks as supervised learning problems. Firstly, we perform a minimal pre-processing of sentence segmentation and number normalization on the documents. Then, Stage 1 handles sub-task 1 and the Stage 2 handles sub-tasks 2 respectively, and the remaining ones are handled by Stage 3 of our pipeline.

Before proceeding to describe our approach, we describe the baseline model, provided by the task organizers. The baseline treats the detection of Q, ME, MP and Qual spans all as sequence labeling problems. It uses the spaCy Entity Tagger model (Honnibal et al., 2020) to extract all these four spans. The Units for these Qs are obtained by

[2] https://github.com/harperco/MeasEval/tree/main/annotationGuidelines

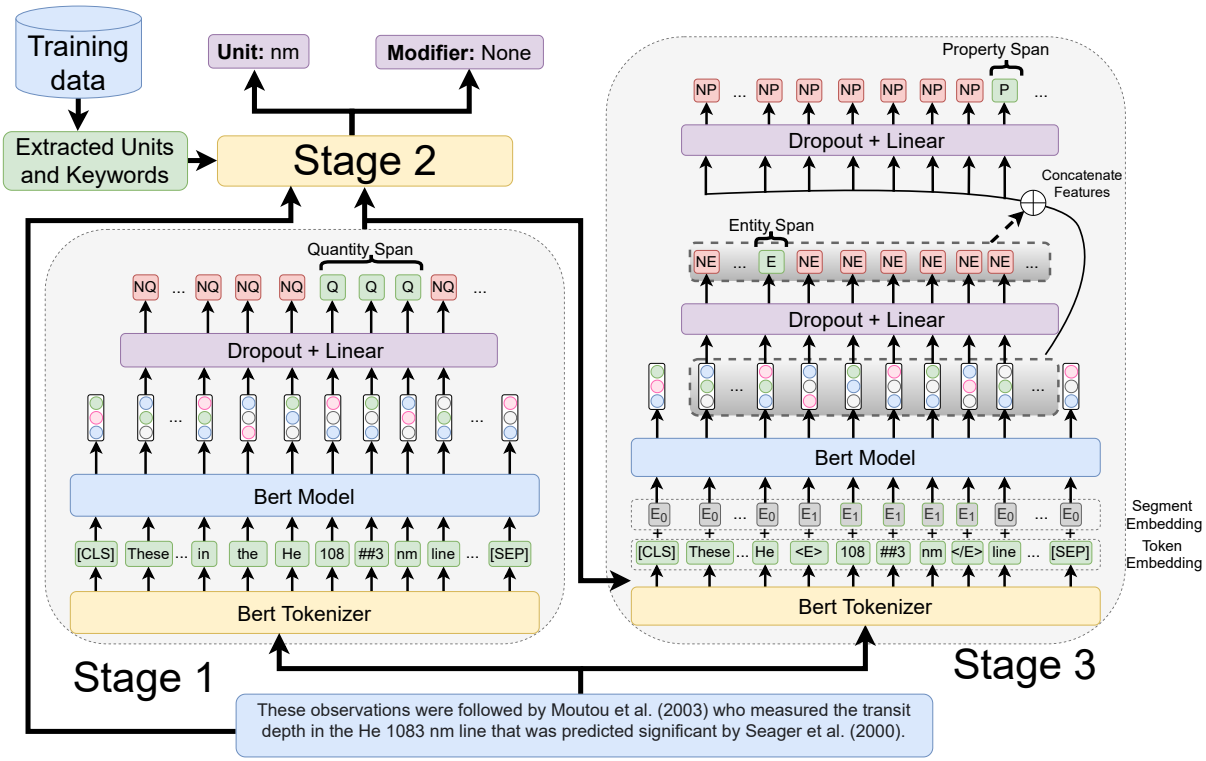


Figure 2: Overview of our Pipeline

matching the largest Units in these predicted spans with those from the train dataset.

### 4.1 Stage 1

Similar to the baseline, we treat the Q spans learning problem as a sequence labelling problem. This is an intuitive step as it can detect multiple spans within the same text segment while being significantly cheaper in terms of the computation cost. Specifically, for a given sentence $s$, the input to our model is $[CLS]$ $s$ $[SEP]$. It is sub-word tokenized (Wu et al., 2016) to get the one-hot sub-token sequence $w_0, w_1 \dots w_n$. These sub-tokens are then fed to BERT to obtain the contextualized representations $x_0, x_1 \dots x_n$. as follows.

First the word vectors are obtained using the Embedding $E$ and Positional-Embedding $E_{pos}$:

$$x_j^{(0)} = w_j E + E_j^{pos} \quad (1)$$

Then these vectors are passed through $L$ layers of transformer encoder (Vaswani et al., 2017) to obtain the contextualized representation. Each transformer encoder layer $l$ receives the output vector sequence $\{(x_j^{(l-1)})_{j=0}^n\} = x_0^{(l-1)}, x_1^{(l-1)} \dots x_n^{(l-1)}$ from the previous layer $l-1$ and computes the output representation $\{(x_j^{(l)})_{j=0}^n\}$ as follows:

$$\{(z_j^{(l)})_{j=0}^n\} = LN(MSA(\{(x_j^{(l-1)})_{j=0}^n\}) + \{(x_j^{(l-1)})_{j=0}^n\}) \quad (2)$$

$$\{(x_j^{(l)})_{j=0}^n\} = LN(\ \{(z_j^{(l)})_{j=0}^n\} + \{((W_2^l)^T f((W_1^l)^T z_j^{(l)} + b_1^l) + b_2^l)_{j=0}^n\}) \quad (3)$$

Here $MSA$ is Multi-headed Self Attention and $LN$ denotes Layer Norm. $W_2^l, W_1^l, b_1^l, b_2^l$ are trainable parameters and $f$ is the activation function.

The final contextualized representations $\{(x_j)_{j=0}^n\} = \{(x_j^{(L)})_{j=0}^n\}$ are the outputs of the $L^{th}$ transformer layer. Finally, these representations $x_j$ (excluding $j=0$, $j=n$ for $[CLS]$, $[SEP]$ tokens) are each classified to a binary label:

$$(y_j^{NQ}, y_j^{Q}) = W_c^T x_j + b_c \quad (4)$$

Here $W_c$ and $b_c$ are learnable parameters and $(y_j^{NQ}, y_j^{Q})$ are the logits. This formulation of our problem can also be treated as the popular BIO tagging scheme excluding the 'B' beginning tag. This is then used to greedily match the largest contiguous span of sub-tokens with positive labels.

### 4.2 Stage 2

This stage receives the Q span predictions from Stage 1 as input and uses a method similar to the baseline, to obtain the Units. We extracted the set of Units occurring in the annotated Qs, from the documents in the dataset. However, in scientific documents, often combinations of units are present (e.g. $Kgms^{-2}$ is a combination of '$Kg$', '$m$' and '$s$'). Our future work includes extending our approach to be exhaustive to handle such complex combinations of units.

To obtain the keywords for modifiers, given a Q span, we extracted the set of tokens occurring inside the span as well as in the neighboring window of 10 characters, on either side of the actual span. We discarded stopwords, punctuation marks and numbers. Then, we calculated the rate of co-occurrence between the remaining set of tokens and the Mods in the train dataset. This helped us to obtain keywords acting as significant cues for the respective Mod classes. Examples include "approximately" for IsApproximate, "greater than" for IsRange, etc. Another challenge with the sub-task is the presence of similar sets of keywords corresponding to multiple Mod types. For example, the Mods 'IsMean' and 'IsMeanHasTolerance' are very similar with the slight difference that keywords corresponding to the Mod 'IsMeanHasTolerance' contain the additional symbol, '±'. We adopted a hierarchical approach in order to detect such minute differences and correctly identify the type of Mod for every Q span, e.g. IsMeanHasTolerance is True when IsMean and HasTolerance are both true. We started by detecting a general Mod class, and gradually used extra cues to classify the span into more specific Mod classes such as {IsMeanHasSD, IsMeanHasTolerance, IsRangeHasTolerance, IsList}.

### 4.3 Stage 3

The input to this stage is the sentence-quantity tuple $\langle s, q \rangle$ and our objective is to detect the spans for ME, MP and Qual. There could be multiple Qs in a single sentence. We treat detecting ME, MP, and Qual as three sequence labeling sub-tasks in a multi-task learning setting.

We create a modified sentence $s'$ where the Q span $q$ inside the sentence is enclosed within a special start marker ⟨E⟩, and a special end marker ⟨/E⟩ (Baldini Soares et al., 2019; Kaushal and Vaidhya, 2020; Zong et al., 2020). We additionally have a special segment embedding for the Quantity ($q$) portion of the quantity-context encoded sentence $s'$, different from the remainder of the sentence. We input $s'$ and corresponding segment embeddings to BERT and obtain quantity-aware contextualized vectors $\{v_1, v_2 ... v_n\}$ for each of the $n$ sub-tokens in $s'$. We then obtain the ME task logits $e_i$, for each sub-token vector $v_i$:

$$e_i = W_e^T v_i + b_e \tag{5}$$

Here $W_e^T$ and $b_e$ are learnable parameters. Now, as per the annotation rules of the task, a Q will have an associated MP only if an ME related to the given Q exists. Hence, for predicting the MP, we extract features from the ME task logits and concatenate them with each sub-token vector $v_i$ as follows:

$$r = [max_{i=1}^{n} e_i; mean_{i=1}^{n} e_i] \tag{6}$$

$$p_i = W_p^T [v_i; r] + b_p \tag{7}$$

Here $W_p^T$ and $b_p$ are learnable parameters, $max$ and $mean$ are element-wise operations and $p_i$ is the logit of the $i^{th}$ sub-token for the MP sub-task. Here ; denotes concatenation. Similarly, we obtain the logits $qu_i$ corresponding to the Qual task, for every sub-token vector $v_i$ of the sentence $s$, as follows:

$$qu_i = W_{qu}^T [v_i; r] + b_{qu} \tag{8}$$

Here $W_{qu}^T$ and $b_{qu}$ are learnable parameters. The model is trained with the following combined multi-task learning objective:

$$Loss(s, q, (y_i^e)_{i=1}^n, (y_i^p)_{i=1}^n, (y_i^{qu})_{i=1}^n) = \sum_{j=1}^{n} (\mathcal{L}(e_j, y_j^e) + \mathcal{L}(p_j, y_j^p) + \mathcal{L}(qu_j, y_j^{qu})) \tag{9}$$

Here $(y_i^e)_{i=1}^n, (y_i^p)_{i=1}^n, (y_i^{qu})_{i=1}^n$ are ground truths for each sub-token for the ME, MP and Qual sub-tasks respectively; $\mathcal{L}$ is the softmax cross-entropy loss (Dunne and Campbell, 1997).

Similar to Stage 1, we greedily match the longest contiguous positive labeled spans for each of the three sub-tasks and obtain the ME span $e$, MP span $p$ and Qual span $qu$ corresponding to the input Q span $q$ for the sentence $s$. Here $(q, e, p, qu)$ forms an annotation set which is then post processed to generate the relations HP, HQ and QS on this annotation set as per their definitions in §3.

| Model | Precision | Recall | F1 |
|---|---|---|---|
| BERT-base | 0.872 | **0.972** | 0.919 |
| BERT-large | 0.874 | 0.933 | 0.902 |
| RoBERTa-BioMed | 0.890 | 0.951 | 0.919 |
| SciBERT | **0.920** | 0.889 | 0.904 |
| BioBERT | 0.904 | 0.946 | **0.924** |

Table 1: Stage 1 Results

| Model | $F1_{ME}$ | $F1_{MP}$ | $F1_{Qual}$ |
|---|---|---|---|
| BERT Individual | 0.499 | 0.386 | 0.137 |
| BERT ME, MP | 0.515 | **0.467** | N/A |
| BERT MP, Qual | N/A | 0.433 | 0.166 |
| BERT ME, Qual | 0.420 | N/A | **0.191** |
| BERT ME, MP, Qual | **0.517** | 0.465 | **0.191** |
| BERT X | 0.459 | 0.330 | 0.125 |
| BERT Y | 0.510 | 0.428 | 0.143 |

Table 2: Multi-Task Results. $F1_{ME}$, $F1_{MP}$, and $F_{Qual}$ are the F1 measures on the ME, MP, and Qual tasks respectively.

### 4.4 Domain Specific BERT

Domain specific language model weights lead to a significant performance boost (Müller et al., 2020; Nguyen et al., 2020; Lee et al., 2019; Vaidhya and Kaushal, 2020; Beltagy et al., 2019). SciBERT (Beltagy et al., 2019), BioBERT (Lee et al., 2019), and RoBERTa-BioMed (Liu et al., 2020; Gururangan et al., 2020) performed relatively well as they are pre-trained on scientific documents in domains relevant to our task.

## 5 Experiments and Discussion

All experiments were performed using PyTorch (Paszke et al., 2019) and HuggingFace's transformers (Wolf et al., 2019). Optimization was done using Adam (Kingma and Ba, 2014). We include the complete set of experimental parameters in §D.

### 5.1 Development Phase

After dividing the 5 sub-tasks into 3 stages, we worked on each stage individually. We trained the models exclusively on the train dataset and used the trial dataset for validation and hyperparameter tuning. We used the F1, Precision and Recall metrics for each token in the sequence labeling sub-tasks, for evaluating individual components over the validation set during the development phase.

| Model | $F1_{ME}$ | $F1_{MP}$ | $F1_{Qual}$ |
|---|---|---|---|
| BERT-base | 0.517 | 0.465 | 0.191 |
| BERT-large | 0.573 | 0.446 | **0.317** |
| RoBERTa-BioMed | **0.577** | 0.473 | 0.232 |
| SciBERT | 0.556 | 0.486 | 0.188 |
| BioBERT | 0.575 | **0.501** | 0.297 |

Table 3: Stage 3 Results

Table 1 shows the performances of various BERT models in Stage 1. We observe that BioBERT delivers the best F1 score, followed by BERT-base and RoBERTa-BioMed. Much to our surprise, BERT-Large and SciBERT performed worse than BERT-base despite their large size (Li et al., 2020) and domain specificity.

In order to understand the role of each component of our model in Stage 3, we perform various ablation studies as shown in Table 2. First, we experiment with various combinations of multi-task learning with the three tasks - ME, MP and Qual. We observe that multi-task learning can lead to significant gains on all three tasks. Only the multi-task combination of ME and Qual led to performance reduction. Multi-task training all three tasks together nearly gives the best performance on all three metrics. We attribute this gain in performance to the inter-related natures of the three sub-tasks.

Secondly, we study the importance of segmentation and concatenation of features. We create BERT X, which doesn't add separate segment embeddings for the Q span, and BERT Y which does not concatenate the ME logit features for predicting MP and Qual spans. From Table 2, we observe that BERT X has a significant reduction in performance for all the three sub-tasks upon excluding the segment embeddings, as the model input doesn't have a clear demarcation between the Q span portion and non-Q span portion of the sentences. We also observe a reduction in performance for MP and Qual for BERT Y, showing the importance of fusing the logits of ME for the former two sub-tasks.

Similar to Stage 1, we experiment with various BERT models as shown in Table 3. Here we observe that RoBERTa-BioMed, BioBERT and BERT-large perform the best for ME, MP and Qual respectively. BERT-Base performs the worst for all of them. All the models except BioBERT have significantly lower $F1_{Qual}$ than BERT-Large. Each model produces an $F1_{ME}$ score greater than $0.5$.

### 5.2 Post-Evaluation Phase

The evaluation was done using the official script[3]. The classification and relation extraction sub-tasks were both evaluated by a binary match score and the span identification tasks by a SQuAD style (Ra-

[3] https://github.com/harperco/MeasEval/blob/main/eval

| Model | Q | ME | MP | Qual | Unit | Mod | HQ | HP | QS | Overall |
|---|---|---|---|---|---|---|---|---|---|---|
| **Evaluation Phase** | | | | | | | | | | |
| Baseline | 0.815 | 0.066 | 0.068 | 0.028 | 0.531 | 0.000 | 0.081 | 0.010 | **0.014** | 0.225 |
| **Submission** | 0.787 | 0.113 | 0.012 | 0.005 | 0.748 | 0.309 | 0.076 | 0.006 | 0.000 | 0.278 |
| **Post-Evaluation Phase** | | | | | | | | | | |
| BERT-base | 0.828 | 0.338 | 0.277 | 0.072 | 0.765 | **0.465** | 0.310 | 0.174 | 0.000 | 0.402 |
| BERT-large | 0.705 | 0.343 | 0.296 | 0.081 | 0.755 | 0.442 | 0.325 | 0.207 | 0.000 | 0.392 |
| RoBERTa-BioMed | 0.812 | 0.384 | 0.365 | 0.104 | 0.804 | 0.434 | 0.383 | 0.238 | 0.005 | 0.440 |
| SciBERT | 0.809 | 0.382 | 0.324 | 0.072 | **0.811** | 0.435 | 0.354 | 0.230 | 0.000 | 0.433 |
| BioBERT | **0.844** | **0.407** | **0.365** | **0.111** | 0.796 | **0.465** | **0.400** | **0.269** | 0.000 | **0.456** |

Table 4: Test Set Performance

jpurkar et al., 2016) overlap score. The leaderboard ranking was based on a global F1 score averaged across all sub-tasks.

For our official submission, we selected BioBERT as it achieved the best F1 score in Stage 1 and near-best performance for the tasks in Stage 3. Minor discrepancies in the submission format involving the annot-id reference, quotes, whitespace-sensitivity and utf-8 encoding, not detected by the evaluation script were fixed in the post-evaluation phase. Table 4 shows the final performance of our models. After proper conversion to the desired format during the post-evaluation phase, we also evaluated various other BERT models along with our best model, BioBERT. BioBERT delivers the best performance of 0.456 F1 (Overall) followed by RoBERTa-BioMed and SciBERT. BioBERT also performs best on 7 of the 9 individual tasks.

### 5.3 Future Work

Stage 3 of our pipeline operates at a sentence-level, so for a given Q span, it does not capture the ME, MP, and Qual spans occurring across sentences. However, our approach can be easily extended to consider the nearby sentences or even the entire document (at the cost of computation speed).

The identification of exact word boundaries for the span identification tasks is crucial. Treating these tasks as sequence labeling problems and greedily matching for spans can lead to a few problems. For example, if a sub-token occurring within a long span is mislabeled, then the span is split into two components. In the future, we can explore leveraging contrastive learning (Chen et al., 2020) to improve the predictions for exact word boundary match. We can have transition based labeling layers such as Conditional Random Fields (CRFs) (Wallach, 2004) over the more popular BIO/BIOES sequence tagging schemes (Yang et al., 2018).

Lastly, while the multi-staged approach is fairly interpretable at the intermediate outputs of Q spans, it also leads to a few issues. The predictions for MP, ME and Qual spans in Stage 3 are heavily dependent on the Q spans from Stage 1, and there does not exist any mechanism to rectify errors in Stage 1 later, in our approach. There is also an exposure bias (Schmidt, 2019; Galloway et al., 2019) as the model is trained on the ground truth, while tested on the predicted Q spans. Moreover, we believe that having common weights between the BERT models of Stage 1 and Stage 3 will not only make our approach faster and lighter, but also more performant through multi-task learning.

## 6 Conclusion

In this paper, we present our system details for the SemEval 2021 Task-8: MeasEval which is aimed at extracting entity and semantic relations pertaining to counts and measurements. We use a multi-staged approach where we first identify the quantity spans using BERT, then the units and modifiers for these predicted quantity spans by intelligent templates that leverage extracted units and modifier keywords. Finally we input the quantity-aware sentences to another BERT model to predict ME, MP, and Qual in a multi-task learning settings with feature re-use. Our submission achieved the second runner up position on the leaderboard for the Unit-identification sub-task and it showed the highest improvement in the post-evaluation phase, with an F1 (Overall) score only 0.063 lower than the highest score across both the phases.

## Acknowledgments

We would like to thank the MeasEval organizers, especially Corey Harper, for their constant support and for clearing our doubts. We would like to thank Aadarsh Sahoo for the insightful discussions. We would also like to thank the Department of Computer Science and Engineering, IIT Kharagpur, for providing us with the computing resources.

## References


Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: Pretrained language model for scientific text. In *EMNLP*.

Soumia Lilia Berrahou, Patrice Buche, Juliette Dibie-Barthelemy, and Mathieu Roche. 2013. How to extract unit of measure in scientific documents? In *KDIR: Knowledge Discovery and Information Retrieval*, pages 454–459. Springer.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

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.

Rob A Dunne and Norm A Campbell. 1997. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, volume 181, page 185. Citeseer.

Angus Galloway, Anna Golubeva, Thomas Tanay, Medhat Moussa, and Graham W Taylor. 2019. Batch normalization is a cause of adversarial vulnerability. *arXiv preprint arXiv:1905.02161*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of ACL*.

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.

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

Ayush Kaushal and Tejas Vaidhya. 2020. Winners at W-NUT 2020 shared task-3: Leveraging event specific and chunk span information for extracting COVID entities from tweets. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 522–529, Online. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

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*, 36(4):1234–1240.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez. 2020. Train big, then compress: Rethinking model size for efficient training and inference of transformers. In *International Conference on Machine Learning*, pages 5958–5968. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. RoBERTa: A robustly optimized BERT pretraining approach.

Huaishao Luo, Yu Shi, Ming Gong, Linjun Shou, and Tianrui Li. 2020. MaP: A matrix-based prediction approach to improve span extraction in machine reading comprehension. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 687–695, Suzhou, China. Association for Computational Linguistics.

Qiaozhu Mei and Dragomir Radev. 1979. Information retrieval. In *The Oxford Handbook of Computational Linguistics 2nd edition*. Oxford Press.

Martin Müller, Marcel Salathé, and Per E Kummervold. 2020. Covid-twitter-bert: A natural language processing model to analyse covid-19 content on twitter. *arXiv preprint arXiv:2005.07503*.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. Bertweet: A pre-trained language model for english tweets. *arXiv preprint arXiv:2005.10200*.

Rrubaa Panchendrarajan and Aravindh Amaresan. 2018. Bidirectional LSTM-CRF for named entity recognition. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong. Association for Computational Linguistics.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Lixin Su, and Xueqi Cheng. 2019. Has-qa: Hierarchical answer spans model for open-domain question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6875–6882.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary 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. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Florian Schmidt. 2019. Generalization in generation: A closer look at exposure bias. *EMNLP-IJCNLP 2019*, page 157.

M Sevenster, J Buurman, P Liu, JF Peters, and PJ Chang. 2015. Natural language processing techniques for extracting and categorizing finding measurements in narrative radiology reports. *Applied clinical informatics*, 6(3):600.

Mayank Singh, Barnopriyo Barua, Priyank Palod, Manvi Garg, Sidhartha Satapathy, Samuel Bushi, Kumar Ayush, Krishna Sai Rohith, Tulasi Gamidi, Pawan Goyal, and Animesh Mukherjee. 2016. OCR++: A robust framework for information extraction from scholarly articles. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3390–3400, Osaka, Japan. The COLING 2016 Organizing Committee.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107.

Richard Tzong-Han Tsai, Shih-Hung Wu, Wen-Chi Chou, Yu-Chun Lin, Ding He, Jieh Hsiang, Ting-Yi Sung, and Wen-Lian Hsu. 2006. Various criteria in the evaluation of biomedical named entity recognition. *BMC bioinformatics*, 7(1):1–8.

Tejas Vaidhya and Ayush Kaushal. 2020. IITKGP at W-NUT 2020 shared task-1: Domain specific BERT representation for named entity recognition of lab protocol. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 268–272, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Hanna M Wallach. 2004. Conditional random fields: An introduction. *Technical Reports (CIS)*, page 22.

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. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Jie Yang, Shuailong Liang, and Yue Zhang. 2018. Design challenges and misconceptions in neural sequence labeling. *arXiv preprint arXiv:1806.04470*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32.

Shi Zong, Ashutosh Baheti, Wei Xu, and Alan Ritter. 2020. Extracting covid-19 events from twitter.

| Dataset | Total Documents | Q | ME | MP | Qual | Avg. Q | Avg. ME | Avg. MP | Avg. Qual |
|---|---|---|---|---|---|---|---|---|---|
| Train | 233 | 883 | 875 | 563 | 210 | 3.790 | 3.755 | 2.416 | 0.901 |
| Trial | 65 | 281 | 273 | 179 | 99 | 4.323 | 4.200 | 2.754 | 1.523 |
| Eval | 130 | 499 | 499 | 330 | 162 | 3.838 | 3.838 | 2.538 | 1.246 |

Table 5: Span Statistics. Here *Avg.* signifies the average number of spans present per document.

| Dataset | Total Documents | HQ | HP | QS | Avg. HQ | Avg. HP | Avg. QS |
|---|---|---|---|---|---|---|---|
| Train | 233 | 878 | 560 | 210 | 3.768 | 2.403 | 0.901 |
| Trial | 65 | 275 | 177 | 99 | 4.231 | 2.723 | 1.523 |
| Eval | 130 | 499 | 330 | 162 | 3.838 | 2.538 | 1.246 |

Table 6: Relation Statistics. Here *Avg.* signifies the average number of relations present per document.

| Dependency | Version | Usage |
|---|---|---|
| PyTorch | 1.4 | NN Layers & Autograd |
| Transformers | 4.2 | BERT Models |
| Scikit-learn | 0.23 | Metrics |
| SciPy | 1.5 | Metrics |
| NLTK | 0.5 | Sentence Tokenization |
| Pandas | 1.2 | Loading Files |
| Pandasql | 0.7 | Querying DataFrames |
| Vladiate | 0.0.23 | Validating Results |
| NumPy | 1.18 | Numerical computation |

Table 7: Packages Used

| Hyperparameter | Value |
|---|---|
| Learning Rate | $3e-5$ |
| Stage 1 Epochs | 5 |
| Stage 3 Epochs | 10 |
| Batch Size | 16 |
| Dropout Final | 0.1 |
| BERT Dropout | 0.1 |
| Adam: $(\beta, \epsilon)$ | $((0.9, 0.999), 1e-8)$ |
| Weight Decay | 0 |
| BERT Configuration | Default |
| BERT Embeddings | Trainable |

Table 8: Best Hyperparameters

## A Appendices

Following is the overview of the appendix.

- §B – We provide implementation details: codebases, trained models and detail dependencies.
- §C – We provide details of the dataset in shared task, its statistics and annotation set for the task.
- §D – We detail the experimental settings and hyperparameters.

## B Code and Dependencies

We will make our code public [4] with instructions to replicate our systems. We also release our pre-trained model for our submissions [5].

All experiments were performed using PyTorch (Paszke et al., 2019) and HuggingFace's transformers (Wolf et al., 2019) libraries. The optimization was done using Adam optimize (Kingma and Ba, 2014). We used git for reproducibility setup. In Table 7 we list all the dependencies used in our codebase. We include a step-by-step guide to setup and run the codebase in our README file present within the code also with details to set up our environment.

## C Dataset Details

We experiment on the dataset provided by the task organizers, consisting of gold annotations (Harper et al., 2021) for the set of scientific documents in English which are released here[6]. These scientific documents are a subset of the Elsevier Labs OA-STM-Corpus available publicly[7].

**Basic Annotation Set:** The basic annotation set consists of 4 types of spans and 3 types of relations between them. The span types are Quantity (counts and measurements), Measured Entity (the item whose measurement/count is provided by the Quantity spans), Measured Property (the property of the Measured Entity, whose measurement is provided by the Quantity spans) and Qualifier (special circumstances which affect a particular measurement). These spans are related using three types of Relations - HasQuantity (relates a Measured Entity or a Measured Property to a Quantity), HasProperty (relates a Measured Entity to a Measured Property) and Qualifies (relates a Qualifier to any Measured Entity, Measured Property, or Quantity).

[4] https://github.com/Ayushk4/SE-T8

[5] https://github.com/Ayushk4/SE-T8/releases

[6] https://github.com/harperco/MeasEval

[7] https://github.com/elsevierlabs/OA-STM-Corpus

| Model | Huggingface's Model API |
|---|---|
| BERT-base | bert-base-cased |
| BERT-large | bert-large-cased |
| RoBERTa-BioMed | allenai/biomed_roberta_base |
| SciBERT | allenai/scibert_scivocab_cased |
| BioBERT | dmis-lab/biobert-v1.1 |

Table 9: BERT Versions

| Hyperparameter | Set of Values |
|---|---|
| Learning Rate | $\{3e-4, 3e-5, 3e-6, 3e-7\}$ |
| Number of Epochs | $\{5, 10, 15, 20\}$ |
| Batch Size | $\{4, 8, 16, 24\}$ |

Table 10: Sets of Hyperparameters

**Statistics:** The complete dataset is divided into three parts: train, trial and eval. We train on the train set. Trial is used for validating and Eval is the held-out test dataset on which the final performance of the models are evaluated. In Table 5, we list the dataset statistics for the spans of each type. In Table 6, we list the dataset statistics related to the various relations - (HP, HQ, QS).

## D Experimental Settings

**Preprocessing:** We sentence tokenize every document using the **NLTK** sentence tokenizer. we observed that phrases such as "Fig. 1", "Table. 2" and "et al. ", along with a few others, caused sentences to be tokenized at wrong intervals (due to the presence of "."). We detected and re-joined the instances for such phrases.

**Normalization:** We normalized the dataset by replacing all numerals by the same digit - 0. The helped our model identify the Q spans better. We observed that without normalization, the F1 (Overlap) Score for Q spans decreased considerably (from 0.844 to 0.790).

**Training and Hyperparameters:** The model take $\approx$ 20 seconds per epoch on Tesla P100. The number of parameters are same as BERT. Table 9 lists the HuggingFace model names corresponding to the BERT models we used. We validated our models using F1 metrics for Stage 1 and Stage 3 over the trial dataset. In Table 10 we share the sets of hyperparameters that we explored whereas in Table 8 we mention the **best** set of hyperparameters that we obtained.