Scientific Software Citation Intent Classification using Large Language Models

Ana-Maria Istrate[1][0000−0002−7953−5168], Joshua Fisher[2][0000−0003−2599−4630], Xinyu Yang[3][0000−0001−8368−7773], Kara Moraw[4,5][0009−0005−2202−7006], Kai Li[6][0000−0002−7264−365X], Donghui Li[1][0000−0003−3335−4537], and Martin Klein[7][0000−0003−0130−2097]

[1] Chan Zuckerberg Initiative, Redwood City, CA, 94063, USA
[2] Elsevier, Research Collaborations Unit
[3] Department of Information Science, Cornell University, USA
[4] Software Sustainability Institute, UK
[5] EPCC, University of Edinburgh, UK
[6] School of Information Sciences, University of Tennessee, Knoxville, USA
[7] Los Alamos National Laboratory, Los Alamos, NM, USA

Abstract. Software has emerged as a crucial tool in the current research ecosystem, frequently referenced in academic papers for its application in studies or the introduction of new software systems. Despite its prevalence, there remains a significant gap in understanding how software is cited within the scientific literature. In this study, we offer a conceptual framework for studying software citation intent and explore the use of large language models, such as BERT-based models, GPT-3.5, and GPT-4 for this task. We compile a representative software-mention dataset by merging two existing gold standard software mentions datasets and annotating them to a common citation intent scheme. This new dataset makes it possible to analyze software citation intent at the sentence level. We observe that in a fine-tuning setting, large language models can generally achieve an accuracy of over 80% on software citation intent classification on unseen, challenging data. Our research paves the way for future empirical investigations into the realm of research software, establishing a foundational framework for exploring this under-examined area.

Keywords: Research software · Citation intent · Large Language Models

1 Introduction

Research software is a critical instrument in contemporary scientific environments, as it offers the computational capacity to expand human capacities to observe and investigate phenomena and acquire new knowledge from the increasing amount of data [19]. As a result, software gradually takes more important roles in data-driven science [40] and is regarded as a "first-class research object" by scientists in a growing list of research domains [6]. During the past decade, there has been significant progress in the development of research infrastructure

to support the publishing, using, and crediting research software [39, 2], which in turn, supports empirical investigations into the impacts of research software and their roles in scientific research [27, 34]. All these efforts are believed to contribute to the construction of a more fair and transparent scientific system [14].

Despite this progress, one major gap in existing research is that there is a lack of a more granular understanding of the links between scientific publications and research software, i.e., how software is cited in scientific publications. This question is central to citation context analysis developed from the field of scientometrics, which examines the different types of contexts in citation sentences, such as sentiment, function, or level of importance of individual citations [43]. This approach is able to reveal more granular reasons behind citations and impact and hence contributes to a deeper understanding of how credit and knowledge flow between publications [9]. On the same page, when this method is applied to research software, we can also understand not only how many times a software object is cited in publications, but the reasons why it is cited. This knowledge is critical for the construction of a new research infrastructure for research software and evaluation of research software and its developers.

A large number of citation context classification schemes have previously been proposed, focusing specifically on citations between scientific publications (Scite [31]). However, we are arguing that these schemes cannot be used to sufficiently understand why software is cited in scientific publications, since there may be distinct reasons why a software package is cited in a paper. In light of this topic, very few schemes have been proposed for citations of research software in scientific publications, with the exceptions of SoftCite [12] and SoMeSci [37]. We believe it is vital to revitalize existing efforts by (1) developing a new classification system for citation intents of research software that builds on existing efforts, and (2) applying and testing the system on new software mentions datasets.

In this paper, we are presenting our preliminary results, including (1) a new classification system for software citation intents in full-text scientific publications and (2) performance assessment of machine learning algorithms, in particular large language models, for classifying the citation intent of software-mention sentences. We compile a new dataset that can be used for software citation intent analysis by aggregating and annotating the SoftCite [12] and SoMeSci [37] datasets. We evaluate the performance of the machine learning models on a subset from a recent large-scale software name mention dataset published by the Chan Zuckerberg Initiative (CZI) [22]. More specifically, we are examining the intent of informal citations to the software. Borrowing previous research [35], formal and informal citation approaches to mentioning software in scientific publications are defined as those mentions with and without an official citation to the software respectively. This project, to our knowledge, is the first effort to identify and classify contexts of software mentions (or informal software citations) from full-text publications. It is our hope that our results will improve existing research infrastructure for empirical studies on research software.

2 Related Work

2.1 Research software studies

Software has become a cornerstone of contemporary scientific systems, due to the large quantity of data available to researchers and the computational resources required to analyze such data [10]. Given its elevated importance, it is important to treat research software as a "first-class research object" just like research articles, which requires the support from infrastructure to publish, peer-review, reuse, and cite software entities [6]. Among these requirements, giving and tracing citations to software is central to the assignment of reward to software development activities and promoting researchers' motivation to develop and publish software [13].

Software citation is a highly challenging issue in the scholarly communication system because various empirical studies have found that software is inconsistently cited, if cited at all, in scientific publications [20, 28]. In addition, when a software is cited, there is often a complicated relationship between citations and software, which makes it very hard to trace how specific software is cited [26]. These findings have inspired recent efforts to develop software citation principles, especially aligned with the FAIR Principles [39, 2].

Despite these progresses to develop a more robust software citation infrastructure, it is commonly accepted that information citations to software, or software mentions, are critical for investigating the links between scientific publications and software [38]. This approach is reflected in recent efforts to publish large-scale software name datasets extracted from full-text publications, especially the CZI dataset that covers close to 4 million open-source scientific publications [22], as well as some similar datasets [12]. These fresh open datasets will undoubtedly promote new empirical research on this critical topic to promote the openness of science, including the present research.

2.2 Citation Intent Classification

Citations have long been regarded as a gold standard to measure one's impacts within the scientific system, as they represent one's intellectual debts to other authors [24, 29]. Based on this normative theory, it is possible to construct a systematic evaluation system by collecting the citations between all documents, which is the idea behind the Scientific Citation Index (SCI) as well as many other scientific evaluation systems [16]. Despite the proven effectiveness (at least to some text) of using citations to evaluate one's scientific impacts [4, 17], however, one concern with just focusing on the citation count is that the citation itself can bear multiple semantic meanings. For example, Bruno Latour, the famous sociologist of science, made the following argument:

> "[Sources] may be cited without being read, that is perfunctorily; or to support a claim which is exactly the opposite of what its author intended;

> or for technical details so minute that they escaped their author's attention; or because of intentions attributed to the authors but not explicitly stated in the text." [25]

Such challenges to classic citation analysis method gave rise to a new line of research that focuses on the symbolic meanings of citations in the full-text publications, often called *citation context and content analysis*[8]. In their review of this topic, Zhang et al. identified a few important aspects of the symbol that have been analyzed, such as sentiment, function, or level of importance of individual citations [43].

In addition, a few important classification systems have been proposed to classify regular citations (especially those citing research articles) during the past few decades, each with their own categories and considerations. [23, 30, 31, 41]. However, Cohan et al. [7] argue that these classification systems are usually too fine-grained to allow a meaningful application to software citations. Having many fine-grained categories successfully captures rare contexts but hinders a meaningful analysis of the citation impact. More recent efforts, such as SoftCite [12] and SoMeSci [37], are trying to directly address this research problem, by developing citation context categories dedicated to software entities. However, both of these efforts are based on and only tested using limited publication samples. Moreover, the citation context categories are not the same between the two datasets. As a result, we believe there is still a large gap in this field that can be addressed by our efforts to apply advanced data science methods to classify software citation sentences to a common scheme.

3 Methods

We first defined a set of citation intent classes and created mappings for any existing informal software citation datasets with intention annotations. We then used the combined dataset to fine-tune multiple language models.

3.1 Citation Intent Classes

We reviewed multiple schemes for citation context and intention that have been proposed for both regular research publications and software. The respective schemes are listed in Table 1. Both the ACL-ARC and SciCite were proposed for the intent classification of research article citations. For their work on the ACL Anthology Reference Corpus (ARC) [5], Jurgens et al. [23] propose six categories for citation function, unifying several previously proposed schemes. Their scheme focuses on how authors align their work to cited publications and maintains a higher granularity for classifying indirect mentions like background information or contrasting related works than our scheme does. We found that these kinds of mentions currently don't occur with a high enough frequency to warrant further splitting these classes. For SciCite, Cohan et al. [7] similarly argue that previous datasets for citation intent often use too fine-grained schemes. They propose

instead to use only three categories, focusing on direct use and comparison of results, and regarding everything else as background information providing more context.

However, these schemes cannot be directly transferred to software citations. Unlike research article citations, software can be cited as research software that has been created as part of the publication. Both the SoftCite [12] and SoMeSci [37] provide annotations for the intent of software citations and propose software-specific schemes. Apart from software creation and usage, they both consider categories related to software publication, namely *sharing* and *deposition*.

Table 1: Citation Intent Schemes.

Source	**Target**	**Categories**
ACL-ARC [23]	publications	Background, Motivation, Uses, Extension, Comparison or Contrast, Future
SciCite [7]	publications	Background information, Method, Result comparison
SoftCite [12]	software	Created, Used, Shared
SoMeSci [37]	software	Usage, Mention, Creation, Deposition

For the development of our citation intent scheme, we established the following two guiding principles:

1. The scheme should be able to distinguish the most common and relevant types of citation intent.
2. The scheme should exhibit high inter-annotator agreement so that it can be consistently applied by a human.

The first principle ensures that the resulting scheme is not too fine-grained, as this would hinder a meaningful analysis in future empirical research. The second principle allows multiple annotators to label a potentially large corpus consistently. Based on the principles and the previously described schemes, we propose the following three categories in our scheme for informal research software citation intent:

- **Paper <describes_creation_of> Software:** the paper describes or acknowledges the creation of a research software entity.
- **Paper <describes_usage_of> Software:** the paper describes the use of research software in any part of the research procedure, for any purpose.
- **Paper <describes_related_software> Software:** the paper describes the research software for any other reasons beyond the first two categories. Note that throughout the paper, we refer to this category by using "related" and "mention" interchangeably.

The most relevant distinction in our scheme is that between a sentence in a publication describing the creation of a piece of software as opposed to one citing

the usage of software. We make sure not to confound the annotation process or analysis with any rare labels by instead encapsulating any other mention in the third category.

Similar to existing efforts, this scheme only considers *functional* intents, i.e., functional reasons for mentioning the software in publications, instead of other aspects of the intent, such as sentiment and importance [43]. In contrast to the schemes proposed for SoMeSci and SoftCite, we specifically did not include a category for *sharing* or *deposition*. We believe that distinguishing this citation intent from creation and usage is not strongly relevant to the evaluation of impact of software being mentioned in publications, especially considering the case where sharing software is often strongly related to creating the software in the first place.

An important attribute of our scheme is that it is designed to be applied on the sentence level: the evaluation is made based on each sentence where a software entity is mentioned. Hence, a paper-software pair can have multiple citation intents if a software entity is mentioned multiple times in the paper. For creating our datasets, we decided that each sentence could only be classified into one category. In the case where multiple categories were applicable, we chose the category carrying more weight in evaluating impact, where *Paper* < *describes_creation_of* > *Software* has a larger weight than *Paper* < *describes_usage_of* > *Software*, which in turn has a larger weight than *Paper* < *describes_related* > *Software*. Note that since we are considering only one citation intent per sentence, we are making the assumption that even if multiple software are mentioned in the same sentence, they are mentioned with the same intent. In some rare corner cases, such as multiple software being mentioned in the same sentence with multiple intents, this assumption will not hold true. Hence, our findings are not applicable in these cases.

3.2 Data

We wanted to re-use existing datasets as much as possible and build on top of previous work, rather than create new datasets and define new gold standards. This is why we chose to build on the SoftCite [12] and SoMeSci [37] datasets by merging them into one representative dataset that can be used for analyzing software citation intent. The datasets, for the most part, consist of single sentences that contain a software mention (*informal* citation, by means of verbal reference to software, whereas a *formal* citation would be by verbal means in conjunction with an included URI, or an official citation, such as a literature reference) and their corresponding labels, which vary between the datasets. Consolidating these similar yet still slightly different datasets was outside the scope of this work. However, given our decision regarding citation intent classes outlined above, we had to make a few adjustments to the existing labels in the provided datasets. We did this through manual curation. Table 2 shows the mappings between our proposed scheme and software citation intent schemes used in the SoMeSci and SoftCite datasets. From the SoftCite dataset, we were able to transfer labels **Used** and **Created** directly to our **Usage** and **Creation** classes and mapped

most of the **Shared** labeled data into **Creation**. After careful consideration and debate between multiple annotators, we moved some records that had multiple labels or no labels at all into our **Mention** category. For the SoMeSci dataset, we transferred the **Usage**, **Mention** and **Creation** labels straight to our own labels. We disregarded entries with a label of **Deposition**.

Table 2: Mappings to other software citation intent schemes.

Ours	**SoftCite**	**SoMeSci**
Paper <describes_creation_of> Software	Created	Creation
Paper <describes_usage_of> Software	Used	Usage
Paper <describes_related> Software	n/a	Mention

As part of data curation, we created a pipeline that downloaded all available full text of papers in the two datasets (SoftCite [12] and SoMeSci [37]) via the PMC API [42] in order to augment the existing data at the sentence-level with an expanded citation context of three sentences surrounding the citation: *leading*, *citing* and *trailing sentence*. After all pre-processing, we ended up with a single dataset. The final dataset consists of 3188 software citations, each labeled as **Creation**, **Usage**, or **Mention**, along with the sentence in which the software mention occurs (<*citing sentence*>) and the citation context. The citation context consists of:

– <*leading sentence*><*citing sentence*><*trailing sentence*>

Some examples from the combined training dataset can be found in Table 3. In addition, we augmented the dataset with 1,000 sentences that contained no citation contexts by sampling randomly from the set of sentences that were not tagged with a software mention. This data can be used as negative training examples. The distribution of labels and the number of words in the contexts in the training dataset used can be seen in Figure 1.

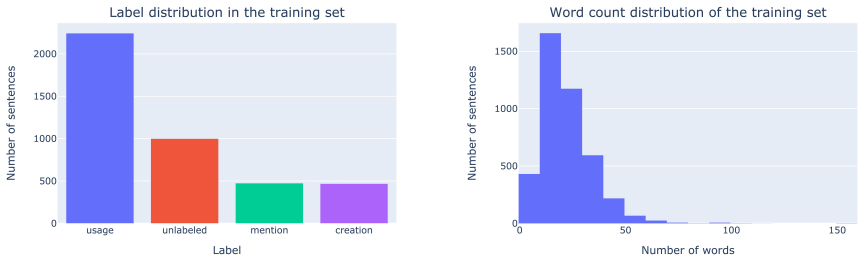

Fig. 1: Training Dataset Data Distribution

Table 3: Examples from the training data

Intent type	Sentence	Full Context
creation	We have developed coXpress as a means of identifying groups of genes that are differentially co-expressed.	**We have developed coXpress as a means of identifying groups of genes that are differentially co-expressed.** The utility of coXpress is demonstrated using two publicly available microarray datasets.
usage	The resulting trajectories were analyzed using the CPPTRAJ module in AMBERTOOLS 16 [55,56].	To study the dynamic behavior of the proposed ligand-protein complex, we have performed in a short production run of 10 ns using AMBER 14. **The resulting trajectories were analyzed using the CPPTRAJ module in AMBERTOOLS 16 [55,56].**
mention	M-Track provides a valuable and user-friendly interface to streamline the analysis of spontaneous grooming in biomedical research studies	**M-Track provides a valuable and user-friendly interface to streamline the analysis of spontaneous grooming in biomedical research studies.**
unlabeled	Developing a new drug is an evidence-based exercise.	**Developing a new drug is an evidence-based exercise.**

We used the combined dataset of 4,188 sentences to train the language models. We split the dataset 80/20 for training and testing in order to facilitate a reasonable comparison between models. We evaluated the models on the test held-out portion of the data that the model has not seen during training. Moreover, we had an additional dataset of 210 samples curated by Chan Zuckerberg Initiative. This dataset is a subset of the CZI Software Mentions Dataset [21] and was manually curated by CZI annotators by reviewing sentences that contain mentions of software names; the dataset was initially curated using a more granular intent classification which was subsequently mapped to the intent classification described above (**creation**, **used**, **mention**). Note that since the original CZI Software Mentions Dataset was not annotated with intent classification classes, this has been done manually by CZI bio-curators after the initial dataset has been published. Because of the effort required and the size of the initial dataset, we were only able to use a subset for evaluation. One annotator initially classified the sentences, and an additional annotator resolved any conflicts at an ulterior time.

3.3 Training Models

We explored finetuning several BERT [11] models, as well as GPT-3.5 [32] and GPT-4 [33] in various training settings.

BERT models We studied four different BERT models, namely BERT [11], DistilBERT [36], SciBERT [3] and PubMedBERT [18]. BERT [11] was pre-trained on BookCorpus [44] and English Wikipedia [15]. It is well suited for fine-tuning downstream tasks that use a full sentence, e.g. for text classification. DistilBERT [36] is a smaller and thus a faster BERT model. It was pre-trained on the same corpus using knowledge distillation with BERT as its teacher. SciBERT was pre-trained on a corpus of scientific texts from Semantic Scholar [1]. It was found to outperform BERT on tasks and datasets in the scientific domain. PubMedBERT [18] was in turn trained on biomedical papers, specifically abstracts from PubMed and full-text articles from PubMedCentral [42]. Hence, it is tailored to tasks in the biomedical domain.

We used the model architecture provided by Hugging Face's library, applying fine-tuning to all four models for text classification tasks. This fine-tuning was consistent across models, employing identical parameters: (epochs=10, learning_rate=2e-5, weight_decay=0.01).

GPT-3.5/4 We also investigated GPT-3.5 and GPT-4 in three different learning settings: *zero-shot learning*, *few-shot learning* and *fine-tuning*. In *zero-shot learning*, the model is only given a description of the task before solving the task. Specifically, we used the system message described in Listing 1 to instruct the model. In *few-shot learning*, the model receives a similar instruction followed by a handful of examples for expected interactions between the user and the assistant (i.e. the model). The model is supposed to learn from these few examples how to generalize on new data. For this, we sampled five examples from each class (creation, usage, mention, and none) and provided these to the model. The corresponding prompt is shown in Listing 2.

Fine-tuning can further improve a model's performance. Instead of a handful of examples, a larger training set is provided. We fine-tuned GPT3.5 on the sentence and full context, using the same dataset as for the BERT models. Both models were fine-tuned for a total of 5 epochs using the OpenAI API and the 'gpt-3.5-turbo' model. The OpenAI API does not allow much hyper-parameter tuning besides the number of epochs, so we used the API's default settings. Since the model can give back answers that do not fit into one of the provided classes, we post-process the answers by lowercasing and stripping punctuation marks.

4 Results

We evaluated the models on a 20% test split of the training data (Table 4), as well as on the additional CZI Validation Dataset (Table 5). The evaluation metrics used to assess model performance were precision, recall, F1-score, and overall accuracy, both at the individual label level and for the aggregate performance. In Table 4 we also attach the metrics reported for classification of intent classes by SoftCite [12], as well as SoMeSci [37]. These metrics are extracted from the corresponding papers and are evaluated on different datasets then the ones in this paper. Note that since the classifiers are not evaluated on the same data or even

```
instruction_message = (
        "You are a scientist trying to figure out the citation "
        "intent behind software mentioned in sentences coming "
        "from research articles. "
        "Your four categories are: creation, usage, mention, or none. "
        "The definitions of the classes are: "
        "- creation: software was created by the authors of the paper"
        "- usage: software was used in the paper "
        "- mention: software was mentioned in the paper, but not used, "
        "nor created "
        "- none: none of the previous 3 categories apply"
        "You need to output one category only."
)
# send instruction as system message to shape the assistant's behaviour
zero_shot_messages = [
    {
        "role": "system",
        "content": instruction_message
    }
]
```

Listing 1: **Zero-shot prompt.** The 'instruction_message' sets the assistant's tone, behavior or persona.

trained to predict the same citation intent classes, the results are not necessarily comparable. Nonetheless, we report the metrics where applicable. For example, the SoftCite paper [12] only reports the performance of a trained citation intent classifier for the **used** and **not used** categories. Hence, we show the metrics for the **used** category, mapping it to our **Usage** category. For the SoMeSci [37] dataset, the paper reports metrics for the following classes: **Allusion**, **Usage**, **Creation** and **Deposition**. We map the **Allusion** to our **Mention** category, and the **Usage** and **Creation** classes directly. We don't report the **Deposition** metrics, since we have discarded this category in our own scheme.

4.1 Results of BERT Models

As shown in Table 4, all models achieve high scores across the metrics and categories on the test split. PubMedBERT outperforms the others by a small degree. As seen in Table 5, on the CZI Validation Dataset, model performance drops across the board, DistilBERT, however, outperforming the other BERT models. Considering the moderate sample size of the training dataset, it may imply that a light architecture such as DistilBERT performs better in this dataset, achieving a more balanced result, especially when compared with the original BERT.

For the category-specific results, the classification is related to the availability of the labels in the training and validation dataset - all the models perform

```
# same instruction message as in zero-shot learning
few_shot_messages = [
    {
        "role": "system",
        "content": instruction_message
    }
]

# append examples as messages
for example in few_shot_examples["usage"]:
    # example for user prompt (sentence or context)
    few_shot_messages += [{"role": "user", "content" : example}]
    # example for assistant output (category)
    few_shot_messages += [{"role": "assistant", "content" : "usage"}]
# same for "creation", "mention" and "none"
...
```

Listing 2: **Few-shot prompts.** When interacting with the OpenAI API, a query can have one of three different roles: (1) **system** - where the behavior, tone, or persona, of the GPT assistant is being defined. We set the behavior of the GPT assistant as in the 'instruction_message' presented in Listing1. The other roles assumed can be (2) **user** - which contains the queries presented by the user to the assistant, and (3) **assistant** - which gives back the GPT assistant's response.

best in the **Usage** label, which is the most frequent in both the training and validation set, and perform the worst on the **Creation** label. Notably though, DistilBERT is the only model to achieve non-zero scores in the **Creation** category across all three metrics, highlighting its unique capability to identify and classify this particularly challenging category. For the **Mention** category, the performance drops for all BERT models, with PubMedBERT outperforming the other BERT models. The same trend can be observed for the test split in Table 4. By definition, this category encompasses more varied instances than the other two, which might be why the models struggle to consistently recognise it. In the **Usage** category, SciBERT and PubMedBERT perform the highest.

4.2 Results of GPT-3.5/GPT-4

In general, **fine-tuning** the model at the sentence level seems to achieve the best results for the GPT-3.5 model on both the test split and the CZI validation dataset. The GPT3.5 model fine-tuned on the sentence level achieves the highest performance on the challenging CZI validation set (P = 0.571, R = 0.531, F1 = 0.545, Accuracy = 0.881) surpassing all BERT models as well. Fine-tuning GPT-3.5 on the entire context (containing the leading, citing, and trailing sentence) leads to a decrease in performance. This tells us that feeding the entire context around the sentence is actually not helping the model learn more information

Table 4: **Evaluation of Different Models on the Test Split.** We assessed the overall Precision (P), Recall (R), F1-score (F1), and Accuracy (Acc) across the entire validation dataset. Additionally, we analyzed the Precision, Recall, and F1-score for each label within every model. For comparison, we attach the metrics reported for classification of intent classes by SoftCite [12], as well as SoMeSci [37]. These metrics are extracted from the corresponding papers and are evaluated on different datasets than the ones in this paper.

Model	Training Methodology	Overall				Creation			Mention			Usage			Unlabeled		
		P	R	F1	Acc	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BERT	Fine-tuning	0.866	0.859	0.862	0.903	**0.90**	**0.87**	**0.88**	0.71	0.69	0.70	0.93	0.94	0.94	0.92	0.94	0.93
DistilBERT	Fine-tuning	0.823	0.826	0.824	0.884	0.86	0.81	0.84	0.59	0.62	0.61	0.95	0.93	0.94	0.90	0.95	0.92
SciBERT	Fine-tuning	0.85	0.846	0.846	0.906	0.89	0.77	0.82	0.63	0.68	0.65	0.96	0.95	0.95	0.93	0.99	0.96
PubMedBERT	Fine-tuning	**0.867**	**0.891**	**0.88**	**0.919**	0.88	0.88	0.88	**0.69**	**0.78**	**0.73**	**0.97**	**0.93**	**0.95**	0.94	0.97	0.96
GPT-3.5	Zero-Shot	0.7	0.608	0.627	0.717	0.77	0.46	0.57	0.24	0.47	0.32	0.84	0.86	0.85	0.96	0.65	0.77
	Few-Shot (sentence)	0.617	0.556	0.54	0.612	0.81	0.46	0.59	0.28	0.24	0.26	0.93	0.57	0.71	0.45	0.95	0.61
	Few-Shot (context)	0.546	0.512	0.463	0.5	0.72	0.57	0.64	0.21	0.15	0.17	0.88	0.35	0.5	0.38	0.98	0.55
	Fine-tuning (sentence)	0.839	0.867	0.851	0.9	0.8	0.91	0.85	0.65	0.69	0.67	**0.97**	**0.93**	**0.95**	0.94	0.93	0.94
	Fine-tuning (context)	0.766	0.808	0.783	0.857	0.66	0.88	0.76	0.49	0.52	0.51	0.96	0.88	0.92	**0.95**	**0.95**	**0.95**
GPT-4	Zero-Shot	0.684	0.662	0.664	0.815	0.73	0.70	0.71	0.26	0.11	0.15	0.83	0.96	0.89	0.92	0.88	0.90
	Few-Shot (sentence)	0.746	0.736	0.738	0.839	0.74	0.62	0.67	0.46	0.44	0.45	0.93	0.91	0.92	0.86	0.97	0.91
	Few-Shot (context)	0.716	0.73	0.716	0.832	0.68	0.82	0.74	0.43	0.26	0.33	0.92	0.91	0.92	0.83	0.93	0.87
SoftCite	BidGRU x 10	-	-	-	-	-	-	-	-	-	-	0.965	0.992	0.979	-	-	-
	SciBERT	-	-	-	-	-	-	-	-	-	-	0.956	0.995	0.975	-	-	-
SoMeSci	Bi-LSTM-CRF (Test)	-	-	-	-	0.87	0.51	0.64	0.68	0.18	0.29	0.77	0.84	0.8	-	-	-
	Bi-LSTM-CRF (Devel)	-	-	-	-	0.97	0.5	0.66	0.68	0.28	0.4	0.78	0.8	0.79	-	-	-

about the intent on citing the software in the sentence itself. The same observation holds for both GPT3.5/4 few-shot models. None of the **few-shot** and **zero-shot** approaches for both GPT-3.5 and the GPT-4 models achieved close to the performance of the fine-tuned models, which means that despite these models' generalizable power, they still don't hold enough information to be able to classify software citation intent without additional training data. We observe in general that GPT-4 models tend to outperform GPT-3.5 models both in zero and few-shot contexts. Notably, however, both GPT-3.5 and GPT-4 few-shot models generally tend to do worse than the zero-shot models counterparts. We haven't investigated in detail why that might happen, but it is an interesting observation to note that for this task, learning from a few examples is detrimental, whereas learning from a lot (i.e. finetuning) is helpful.

Inspecting per-class performance, we observe that, similarly to BERT models, GPT models tend to do very well on predicting **Usage** and **Unlabeled** labels, achieving precision, recall and F1 score >0.9 for both test splits and CZI Validation Dataset and struggle the most with predicting the **Mention** class. This makes sense given that any software will be, after all, mentioned in the paper. While we instruct the model to predict this label if none of the **Usage** or **Creation** might apply, it is still ambiguous. Some examples of model mistakes on the CZI Validation dataset can be seen in Table 6.

5 Data and Code Availability Statement

Training scripts are available on our GitHub repository: `https://github.com/karacolada/SoftwareImpactHackathon2023_SoftwareCitationIntent`.

Table 5: **Evaluation of Different Models on the CZI Validation Dataset.** We assessed the overall Precision (P), Recall (R), F1-score (F1), and Accuracy (Acc) across the entire validation dataset. Additionally, we analyzed the Precision, Recall, and F1-score for each label within every model.

Model	Training Methodology	Overall				Creation			Mention			Usage		
		P	R	F1	Acc	P	R	F1	P	R	F1	P	R	F1
BERT	Fine-tuning	0.323	0.368	0.335	0.771	0.00	0.00	0.00	0.15	0.29	0.20	0.94	0.85	0.90
DistilBERT	Fine-tuning	0.481	0.412	0.443	0.801	0.71	0.50	0.59	0.29	0.26	0.27	0.94	0.88	0.91
SciBERT	Fine-tuning	0.302	0.308	0.306	0.80	0.00	0.00	0.00	0.27	0.35	0.30	0.95	0.90	0.92
PubMedBERT	Fine-tuning	0.319	0.392	0.342	0.81	0.00	0.00	0.00	0.28	0.39	0.33	0.94	0.91	0.93
GPT-3.5	Zero-Shot	0.464	0.511	0.478	0.8	0.55	0.6	0.57	0.35	0.61	0.44	0.96	0.84	0.89
	Few-Shot (sentence)	0.525	0.291	0.373	0.457	0.5	0.3	0.37	0.6	0.39	0.47	1.00	0.47	0.64
	Few-Shot (context)	0.454	0.195	0.269	0.338	0.5	0.2	0.29	0.33	0.22	0.26	0.98	0.36	0.53
	Fine-tuning (sentence)	**0.571**	**0.531**	**0.545**	**0.881**	0.71	0.5	0.59	**0.59**	**0.7**	**0.64**	**0.98**	**0.93**	**0.95**
	Fine-tuning (context)	0.553	0.503	0.509	0.819	0.83	0.5	0.62	0.41	0.65	0.5	0.97	0.86	0.91
GPT-4	Zero-Shot	0.473	0.544	0.495	0.8	0.58	0.70	0.64	0.35	0.65	0.45	0.96	0.82	0.89
	Few-Shot (sentence)	0.473	0.385	0.421	0.614	**0.7**	**0.7**	**0.7**	0.21	0.17	0.19	0.98	0.67	0.79
	Few-Shot (context)	0.399	0.206	0.269	0.471	0.5	0.2	0.29	0.11	0.09	0.1	0.99	0.54	0.7

Table 6: Examples of mistakes the GPT3.5 model fine-tuned at the sentence level makes on the CZI Validation dataset

Sentence	**True Label**	**Predicted Label**
Very recently, one research group applied Mask R-CNN on cervix segmentation tasks, the obtained (Dice, IoU) score is (0.8711, 0.765) on "Kaggle Dataset" as reported in [26]	mention	usage
Identifying major effect QTL underlying single or multiple traits in various populations determines the successful use of QTL through MAS [8]	mention	none
In Figure 11, p miss of SVM-SMP is nearly equal to 0, which is much better than SVM-LA	usage	none
FVA can be set up in COBRA toolbox using the function fluxVariability()	usage	mention

BERT finetuning scripts can be found under **BERT_finetuning** and all the GPT scripys, including zero-shot, few-shot and fine-tuning under **GPT_models**. The merged dataset, together with training, validation and test splits, as well as GPT-3.5 formatted data and the CZI Validation Dataset can also be found under the **data** folder, together with extra documentation and a README file.

6 Discussion

The determination of software citation intent requires a system to not only identify the entity but understand the semantic relationships provided by the context around the entity. In this study, we focused on the latter part of said system, by investigating which model types can effectively learn the intent of authors by their reporting of software. Prior work done by the SoftCite [12] and SoMeSci [37]

Fig. 2: **GPT-3.5 fine-tuned** Comparison of True and Predicted Labels Distributions. Y axis represents the counts, and X axis the label categories.
pred_gpt3.5 true

groups offered valuable datasets that were combined and normalized to a simple scheme. This corpus allowed for the fine-tuning and experimentation of various flavors of BERT, GPT-3.5 and GPT-4 models. Our intuition was that these models would be able to accurately characterize these intents. One interesting finding was that including the full context in classification seemed to hurt model performance. This insensitivity towards extra contextual clues indicates that intent can typically be determined in close proximity to the mention of the software entity. Further text analysis may elucidate exactly what type of language is characteristic of each intent class. A quick word frequency count in sentences that are of the "creation" class identify "software", "available", "http", "developed", and "source" as the most common. A more proper analysis for all intent types to identify word patterns could improve the classification system.

To test the full breadth of capabilities of the GPT models, we employed various experiments including zero-shot, few-shot and fine-tuned approaches. None of the **few-shot** and **zero-shot** approaches for both GPT-3.5 and the GPT-4 models achieved performance comparable to the fine-tuned models, which means that software citation intent classification is not a task that these models can do out of the box, without additional training. Adding example cases to the prompts in a few-shot setting yielded a decrease in both precision and recall over the evaluation sets compared to the zero-shot setting. Beyond this preliminary

work, future experiments will have to test different versions of prompts to further probe this unexpected behavior, as prompt engineering was not an extensive part of this work. Fine-tuning the GPT-3.5 model generated results comparable to the BERT models. We did not experiment with fine-tuning a GPT-4 model because this process is closed to the general public at the moment of paper publication, but we would expect that a GPT-4 fine-tuned model would achieve even higher performance. The easiest category to predict was **Usage**, followed by **Creation**. The **Mention** category was the hardest for models to learn to predict well, which makes sense given that software falling in the other classes can automatically fall under this category as well. While trained on different intent categories and data, our best models surpass the metrics reported by SoftCite and SoMeSci on the **Creation** and **Mention** categories and are comparable for the **Usage** category.

Fine-tuned models generally exhibited high performance on the test set. However, despite our best efforts to identify and resolve systematic differences in the CZI validation set from our test set, models were not able to achieve similar performance. The only competitive performance can be observed using the fine-tuned GPT-3.5 model. Given this is a challenging dataset coming from a different distribution than the training data, this speaks to the power of the GPT family of models to generalize and find nuance in ambiguous text, compared to BERT models. Further experiments necessitate a proper quality and error analysis of the validation set.

7 Conclusion

In conclusion, this preliminary work presents a new system for the classification of software citation intent in scholarly research and insights into the use of large language models for classifying scientific software citation intent. Building on prior work in this research space, we offer an aggregated and normalized corpus that can be used to train and evaluate the performance of machine learning models tasked on the classification of mentions, usage, and creation of software in text. We present, to the best of our knowledge, the first study on using large language models on predicting software citation intent. The identification of these entities contributes to the link between research software and scientific publications. We believe this work establishes a foundational framework for exploring the under-examined area of studying scientific software citation intent.

8 Acknowledgements

The authors would like to acknowledge the Chan Zuckerberg Initiative for hosting the Mapping the Impact of Research Software in Science hackathon that led to the development of this work. We would also like to thank Michaela Torkar for help with curating the CZI validation dataset and valuable expert feedback. Kara Moraw was supported by the UK Research Councils through grant EP/S021779/1.

References

1. Ammar, W., Groeneveld, D., Bhagavatula, C., Beltagy, I., Crawford, M., Downey, D., Dunkelberger, J., Elgohary, A., Feldman, S., Ha, V., et al.: Construction of the literature graph in semantic scholar. arXiv preprint arXiv:1805.02262 (2018)
2. Barker, M., Chue Hong, N.P., Katz, D.S., Lamprecht, A.L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L.J., Gruenpeter, M., Martinez, P.A., et al.: Introducing the fair principles for research software. Scientific Data **9**(1), 622 (2022)
3. Beltagy, I., Lo, K., Cohan, A.: SciBERT: A pretrained language model for scientific text. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 3615–3620. Association for Computational Linguistics, Hong Kong, China (Nov 2019). https://doi.org/10.18653/v1/D19-1371, `https://aclanthology.org/D19-1371`
4. Bensman, S.J.: Garfield and the impact factor: The creation, utilization, and validation of a citation measure. Annual Review of Information Science and Technology (ARIST) **42** (2008)
5. Bird, S., Dale, R., Dorr, B.J., Gibson, B.R., Joseph, M.T., Kan, M.Y., Lee, D., Powley, B., Radev, D.R., Tan, Y.F., et al.: The acl anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In: LREC (2008)
6. Chassanoff, A., Altman, M.: Curation as "interoperability with the future": Preserving scholarly research software in academic libraries. Journal of the Association for Information Science and Technology **71**(3), 325–337 (2020)
7. Cohan, A., Ammar, W., Van Zuylen, M., Cady, F.: Structural scaffolds for citation intent classification in scientific publications. arXiv preprint arXiv:1904.01608 (2019)
8. Cronin, B.: The need for a theory of citing. Journal of documentation **37**(1), 16–24 (1981)
9. Cronin, B.: The citation process. The role and significance of citations in scientific communication **103** (1984)
10. Crouch, S., Hong, N.C., Hettrick, S., Jackson, M., Pawlik, A., Sufi, S., Carr, L., De Roure, D., Goble, C.A., Parsons, M.: The software sustainability institute: changing research software attitudes and practices. Computing in Science & Engineering **15**(6), 74–80 (2014)
11. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding (2019)
12. Du, C., Cohoon, J., Lopez, P., Howison, J.: Softcite dataset: A dataset of software mentions in biomedical and economic research publications. Journal of the Association for Information Science and Technology **72**(7), 870–884 (2021)
13. Du, C., Cohoon, J., Priem, J., Piwowar, H., Meyer, C., Howison, J.: Citeas: better software through sociotechnical change for better software citation. In: Companion Publication of the 2021 Conference on Computer Supported Cooperative Work and Social Computing. pp. 218–221 (2021)
14. Easterbrook, S.M.: Open code for open science? Nature Geoscience **7**(11), 779–781 (2014)
15. Foundation, W.: Wikimedia downloads, `https://dumps.wikimedia.org`

16. Garfield, E.: "science citation index"—a new dimension in indexing: This unique approach underlies versatile bibliographic systems for communicating and evaluating information. Science **144**(3619), 649–654 (1964)
17. Garfield, E.: Is citation analysis a legitimate evaluation tool? Scientometrics **1**, 359–375 (1979)
18. Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., Poon, H.: Domain-specific language model pretraining for biomedical natural language processing. ACM Transactions on Computing for Healthcare **3**(1), 1–23 (Oct 2021). https://doi.org/10.1145/3458754, http://dx.doi.org/10.1145/3458754
19. Horsfall, D., Cool, J., Hettrick, S., Pisco, A.O., Hong, N.C., Haniffa, M.: Research software engineering accelerates the translation of biomedical research for health. Nature Medicine pp. 1–4 (2023)
20. Howison, J., Bullard, J.: Software in the scientific literature: Problems with seeing, finding, and using software mentioned in the biology literature. Journal of the Association for Information Science and Technology **67**(9), 2137–2155 (2016)
21. Istrate, A.M., Li, D., Fisher, J., Li, K., Klein, M., Yang, X., Moraw, K.: SoftwareImpactHackathon2023: Software Citation Intent (10 2023), https://github.com/karacolada/SoftwareImpactHackathon2023_SoftwareCitationIntent
22. Istrate, A.M., Li, D., Taraborelli, D., Torkar, M., Veytsman, B., Williams, I.: A large dataset of software mentions in the biomedical literature. arXiv preprint arXiv:2209.00693 (2022)
23. Jurgens, D., Kumar, S., Hoover, R., McFarland, D., Jurafsky, D.: Measuring the evolution of a scientific field through citation frames. Transactions of the Association for Computational Linguistics **6**, 391–406 (2018)
24. Kaplan, N.: The norms of citation behavior: Prolegomena to the footnote. American documentation **16**(3), 179–184 (1965)
25. Latour, B.: Science in action: How to follow scientists and engineers through society. Harvard university press (1987)
26. Li, K., Chen, P.Y., Yan, E.: Challenges of measuring software impact through citations: An examination of the lme4 r package. Journal of Informetrics **13**(1), 449–461 (2019)
27. Li, K., Yan, E.: Co-mention network of r packages: Scientific impact and clustering structure. Journal of Informetrics **12**(1), 87–100 (2018)
28. Li, K., Yan, E., Feng, Y.: How is r cited in research outputs? structure, impacts, and citation standard. Journal of Informetrics **11**(4), 989–1002 (2017)
29. Merton, R.K.: The sociology of science: Theoretical and empirical investigations. University of Chicago press (1973)
30. Moravcsik, M.J.: Citation context classification of a citation classic concerning citation context classification. Social Studies of Science **18**(3), 515–521 (1988)
31. Nicholson, J.M., Mordaunt, M., Lopez, P., Uppala, A., Rosati, D., Rodrigues, N.P., Grabitz, P., Rife, S.C.: Scite: A smart citation index that displays the context of citations and classifies their intent using deep learning. Quantitative Science Studies **2**(3), 882–898 (2021)
32. OpenAI: Models - openai api, https://platform.openai.com/docs/models/gpt-3-5-turbo
33. OpenAI: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
34. Pan, X., Yan, E., Cui, M., Hua, W.: Examining the usage, citation, and diffusion patterns of bibliometric mapping software: A comparative study of three tools. Journal of informetrics **12**(2), 481–493 (2018)

35. Park, H., You, S., Wolfram, D.: Informal data citation for data sharing and reuse is more common than formal data citation in biomedical fields. Journal of the Association for Information Science and Technology **69**(11), 1346–1354 (2018)
36. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter (2020)
37. Schindler, D., Bensmann, F., Dietze, S., Krüger, F.: Somesci-a 5 star open data gold standard knowledge graph of software mentions in scientific articles. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 4574–4583 (2021)
38. Schindler, D., Bensmann, F., Dietze, S., Krüger, F.: The role of software in science: a knowledge graph-based analysis of software mentions in pubmed central. PeerJ Computer Science **8**, e835 (2022)
39. Smith, A.M., Katz, D.S., Niemeyer, K.E.: Software citation principles. PeerJ Computer Science **2**, e86 (2016)
40. Symons, J., Alvarado, R.: Can we trust big data? applying philosophy of science to software. Big Data & Society **3**(2), 2053951716664747 (2016)
41. Teufel, S., Siddharthan, A., Tidhar, D.: Automatic classification of citation function. In: Proceedings of the 2006 conference on empirical methods in natural language processing. pp. 103–110 (2006)
42. Wheeler, D.L., Barrett, T., Benson, D.A., Bryant, S.H., Canese, K., Chetvernin, V., Church, D.M., DiCuccio, M., Edgar, R., Federhen, S., et al.: Database resources of the national center for biotechnology information. Nucleic acids research **36**(suppl_1), D13–D21 (2007)
43. Zhang, G., Ding, Y., Milojević, S.: Citation content analysis (cca): A framework for syntactic and semantic analysis of citation content. Journal of the American Society for Information Science and Technology **64**(7), 1490–1503 (2013)
44. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler, S.: Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In: The IEEE International Conference on Computer Vision (ICCV) (December 2015)

Appendix

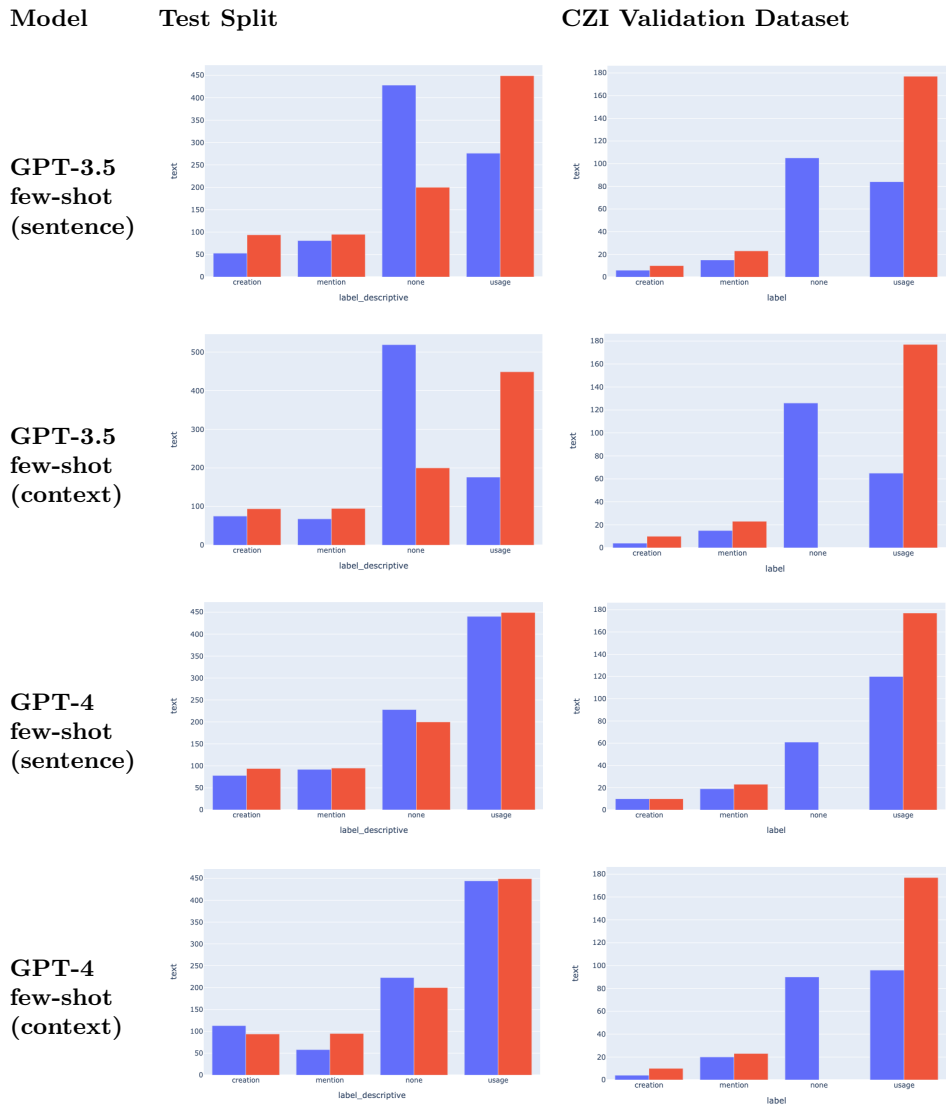

Fig. 3: **GPT-3.5/GPT-4 few-shot** Comparison of True and Predicted Labels Distributions for few-shot GPT-3.5/GPT-4 models, trained both on the sentence and the full context. Y axis represents the counts, and X axis the label categories. pred_gpt3.5/4 true

Fig. 4: **GPT-3.5/GPT-4 zero-shot** Comparison of True and Predicted Labels Distributions for zero-shot GPT-3.5/GPT-4 models. Y axis represents the counts, and X axis the label categories. pred_gpt3.5/4 true

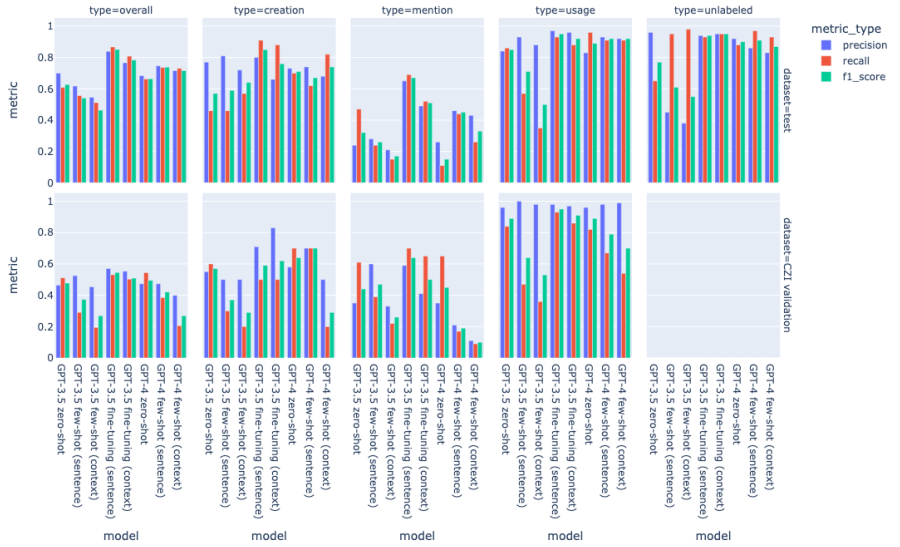

Fig. 5: **Performance of GPT-3.5 and GPT-4 models on the test split and the CZI validation dataset.** Blue bars correspond to Precision, red bars to Recall and green bars to the F1 score. First row corresponds to results on the test split of the training data (0.2 of the training data. Second row corresponds to the CZI validation dataset. The first column shows overall model performance, aggregating the labels in a macro fashion. Each subsequent column represents model performance for that particular intent class. Note that we do not have any 'unlabeled' sentences in the CZI validation dataset, which is why all metrics will be 0.