Eliciting Knowledge from Experts: Automatic Transcript Parsing for Cognitive Task Analysis

Junyi Du[1], He Jiang[1], Jiaming Shen[2], Xiang Ren[1]
[1]Department of Computer Science, University of Southern California
[2]Department of Computer Science, University of Illinois at Urbana-Champaign
{junyidu, jian567, xiangren}@usc.edu, js2@illinois.edu

Abstract

Cognitive task analysis (CTA) is a type of analysis in applied psychology aimed at eliciting and representing the knowledge and thought processes of domain experts. In CTA, often heavy human labor is involved to parse the interview transcript into structured knowledge (e.g., flowchart for different actions). To reduce human efforts and scale the process, automated CTA transcript parsing is desirable. However, this task has unique challenges as (1) it requires the understanding of long-range context information in conversational text; and (2) the amount of labeled data is limited and indirect—i.e., context-aware, noisy, and low-resource. In this paper, we propose a weakly-supervised information extraction framework for automated CTA transcript parsing. We partition the parsing process into a sequence labeling task and a text span-pair relation extraction task, with distant supervision from human-curated protocol files. To model long-range context information for extracting sentence relations, neighbor sentences are involved as a part of input. Different types of models for capturing context dependency are then applied. We manually annotate real-world CTA transcripts to facilitate the evaluation of the parsing tasks[1].

Figure 1: **An example of CTA interview transcript and the human parsed structured text (protocol).** In the protocol, splitting by the highlighted line numbers indicating the sources in transcript, phrases in protocol (called *protocol phrases*) are abstractive description of actions in the transcript. In the transcript, the highlighted numbers are line numbers, and the bolded are *text spans* matched by protocol phrases. The highlighted line numbers are provided by human parsing which provide constraint on mapping protocol phrases back to the transcript, but they are noisy and pointing back to a large scope of sentences, instead of the text span we want to extract.

1 Introduction

Cognitive task analysis (CTA) is a powerful tool for training, instructional design, and development of expert systems (Woods et al., 1989; Clark and Estes, 1996) focusing on yielding the knowledge and thought processes from domain experts (Schraagen et al., 2000). Traditional CTA methods require interviews with domain experts and parsing the interview transcript (*transcript*) into structured text describing processes (*protocol*, shown in Fig. 1). However, parsing transcripts requires heavy human labor, which becomes the major hurdle of scaling up CTA. Therefore, automated approaches to extract structured knowledge from CTA interview transcripts are important for expert systems using massive procedural data.

A natural realization of automated CTA is to apply relation extraction (RE) models to parse interview text. However, the key challenge here is the lack of direct sentence-level supervision data for training RE models because the only available supervision, protocols, are document-level transcripts summaries. Furthermore, the information towards relations between procedural actions spreads all over the transcripts, which bur-

[1]Code is available at: https://github.com/cnrpman/procedural-extraction

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4280–4291
Florence, Italy, July 28 - August 2, 2019.

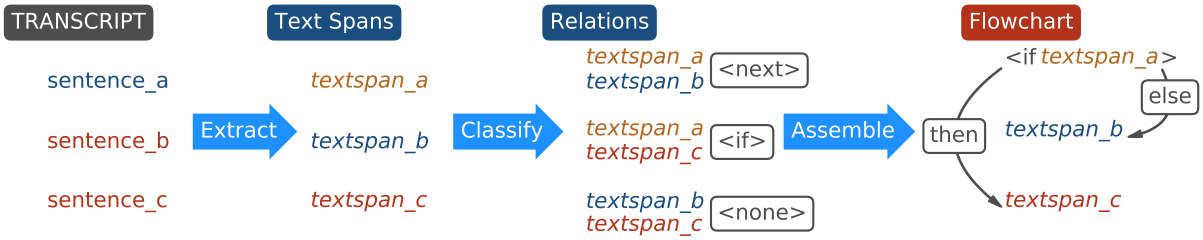

Figure 2: **The framework of Automated CTA Transcripts Parsing.** Text spans are extracted via the sequence labeling model, then the relations between text spans are extracted by the text span-pair relation extraction model (*span-pair RE* model). In the end we assemble the results into structured knowledge (flowchart) for CTA.

dens the RE model to process global information of the text. One previous work (Park and Motahari Nezhad, 2018) studies extracting procedure information on *well-structured text* using `OpenIE` and sentence pair RE models. In this work, however, we focus on *unstructured conversational text* (i.e., CTA interview transcripts) for which `OpenIE` is inapplicable.

To address the above challenges, we develop a novel method to effectively extract and leverage weak(in-direct) supervision signals from protocols. The key observation is that these protocols are structured in the phrase level (c.f. Fig. 1). We split each protocol into a set of *protocol phrases*. Each protocol phrase is associated with a line number that points back to one sentence in the original transcript. Then, we can map these protocol phrases back to *text spans* in transcript sentences and obtain useful supervision signals from three aspects. First, these matched text spans provide direct supervision labels for training text span extraction model. Second, the procedural relations between protocols phrases are transformed into relations between text spans within sentences, which enables us to train RE models. Finally, the local contexts around text spans provide strong signals and can enhance the mention representation in all RE models.

Our approach consists of following steps: (1) parse original protocol into a collection of protocol phrases together with their procedural relations, using a deterministic finite automation (DFA); (2) Match the protocol phrases back to the text spans in transcripts using *fuzzy matching* (Pennington et al., 2014; Devlin et al., 2018); (3) Generate text span extraction dataset and train a sequence labeling model (Finkel et al., 2005; Liu et al., 2017) for text span extraction; (4) Generate text span-pair relation extraction (span-pair RE) dataset and fine-tune pre-trained context-aware span-pair RE model (Devlin et al., 2018). With the trained models, we can automatically extract text spans summarizing actions from transcripts along with the procedural relations among them. Finally, we assemble the results into protocol knowledge, which lays the foundation for CTA.

We explore our approaches from manifold aspects: (i) We experimented different fuzzy matching methods, relation extraction models and sequence labeling models; (ii) We present models for solving context-aware span-pair RE; (iii) We evaluate the approach on real-world data with human annotations, which demonstrates the best fuzzy matching method achieves 47.1% mention level accuracy, best sequence labeling model achieves 38.18% token level accuracy, and best text span-pair relation extraction model achieves 74.4% micro F_1.

2 Related Work

Our work is closely related to procedural extraction, however we focus on conversational text from CTA interviews which is in a low-resource setting and no sentence-by-sentence label is available.

Cognitive task analysis. Cognitive task analysis is a powerful tool for extracting knowledge and thought processes of experts widely used in different domains (Schraagen et al., 2000; Seamster and Redding, 2017). Yet, it is time-consuming and not scalable. Recent years, with the development of natural language processing, techniques are introduced to aid human expertise (Zhong et al., 2015; Roose et al., 2018). Li et al.(2013) used learning agent to discover cognitive model in specific domains. Chaplot et al.(2018) explored modeling cognitive knowledge in well-defined tasks with neural models. However, for the most general setting that extract cognitive processes from interviews, we still need substantial expertise to

interpret the interview transcript.

Procedural extraction. Recent advances in machine reading comprehension, textual entailment (Devlin et al., 2018) and relation extraction (Zhang et al., 2017) shows the contemporary NLP models have the capability of capturing causal relations in some degree. However, it is still an open problem to extract procedural information from text. There were some attempts to extract similar procedural information on well-structured instructional text from how-to community. Park and Motahari Nezhad (2018) treated procedural extraction as a relation extraction problem on sentence pair extracted by pattern matching. They used `OpenIE` for pattern extraction and hierarchical LSTM to classify relation labels of sentence pairs.

Pre-trained language representations. Recent researches showed that language models generically trained on massive corpus is beneficial to various specific NLP tasks (Pennington et al., 2014; Devlin et al., 2018). Language representation has been an active area of research for years. Tons of effective approaches have been developed from feature-based approaches (Ando and Zhang, 2005; Mikolov et al., 2013; Peters et al., 2018) to fine-tuning approaches (Dai and Le, 2015; Alec Radford and Sutskever, 2018; Devlin et al., 2018).

3 Framework

Our automated CTA transcript parsing framework takes interview transcripts as input and outputs structured knowledge consisting of summary phrases. The framework, visualized in Fig. 2, includes two parts: (1) summary text spans extraction and (2) text span-pair relation extraction. The extracted knowledge will then be structured using a flowchart and supports automated CTA.

3.1 Text Spans Extraction

Since CTA interview transcripts are conversational text while structured knowledge are formed of summary phrases describing actions in transcripts (c.f. Fig. 1), we need to first summarize transcript sentences. An intuitive idea is to first leverage off-the-shelf text summarization methods (Shen et al., 2007; Nallapati et al., 2016; Liu et al., 2018). However, CTA is a low-resource task and thus we do not have enough training data for learning seq2seq-based text summarization models. Therefore, in this work, we formulate the summarization of transcript sentences as a sequence labeling task (Liu et al., 2017) and treat the best summarized text span in a transcript sentence as its corresponding summary phrase.

Given a sentence in transcripts, we denote the sentence as $\mathbf{x} = \{x_i\}$ where x_i is the token at position i. The text spans extraction task aims to obtain the prediction $\mathbf{p}_t$ representing the summary text span $\mathbf{t}$ of the transcript sentence $\mathbf{x}$ using a sequence labeling model $\mathbf{p}_t = \mathcal{M}_s(\mathbf{x})$, where $\mathbf{t}$ is a continuous subset of $\mathbf{x}$ labeled by $\mathbf{p}_t = \{p_{ti}\}$ with `IOBES` schema. To train the model, we utilize weakly-supervised sequence labels created in Sec. 4.3.

3.2 Text Span-Pair Relation Extraction

Structural relations between text spans are required to assemble summary text spans into structured knowledge. To extract structural information, following the previous study (Park and Motahari Nezhad, 2018), we formalize text span-pair relation extraction as a sentence pair classification problem. A directed graph $\mathcal{G}_t = (\mathcal{T}, \mathcal{R}_t)$ is used to represent the structured knowledge parsed from a CTA transcript, consisting of nodes for summary text spans in the transcript ($\mathcal{T} = \{\mathbf{t}_i\}$) and edges for procedural information ($\mathcal{R}_t = \{(\mathbf{u}_{ti}, \mathbf{v}_{ti}, r_{ti})\}$ where $\mathbf{u}_{ti}, \mathbf{v}_{ti} \in \mathcal{T}$ are summary text spans and r_{ti} is the procedural relation from text span $\mathbf{u}_{ti}$ to text span $\mathbf{v}_{ti}$). A span-pair RE model $r_{ti} = \mathcal{M}_r(\mathbf{u}_{ti}, \mathbf{v}_{ti}), \forall \mathbf{u}_{ti}, \mathbf{v}_{ti} \in \mathcal{T}$ is then applied to extract relations between all summary text spans $\mathcal{T}$ in the transcript. We train the model using the span-pair RE dataset generated in Sec. 4.4.

To capture the long-range context dependency, we enrich the text span representation $\mathbf{t}$ based on its surrounding contexts and feed the enhance span representation $\mathbf{t}_c$ into the relation extraction model $\mathcal{M}_r$. Examples are shown in Fig. 3.

3.3 Context-aware Models for Text Span-Pair Relation Extraction

We apply state-of-the-art models for natural language entailment (Talman et al., 2018; Devlin et al., 2018) to solve the text span-pair relation extraction task as a sentence pair classification problem. While these models show promising results on the span-pair RE dataset we generated, they do not fully exploit all the information of our dataset. For example, in our dataset, a text span with context information is the combination of matched text span and its surrounding context sentences

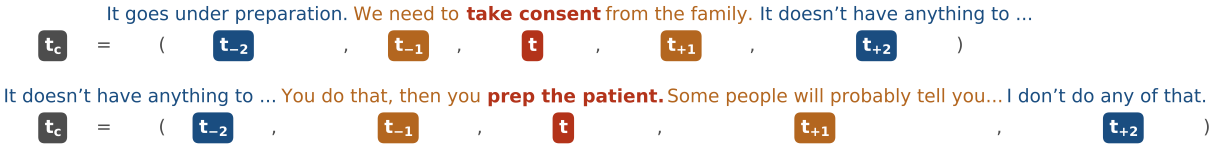

Figure 3: **The construction of text span with context $\mathbf{t}_c$.** The example shows two text spans with context using $K = 2$. Neighbours of text span $\mathbf{t}$ are denoted by $\mathbf{t}_{+i}$ and $\mathbf{t}_{-i}$, $0 < i <= K$

(Fig.3). But in the normal sentence pair classification setting, they are concatenated into a single sequence while its segmentation is ignored. Here, we explored some variants of the neural model, to incorporate the context segmentation and position information with the state-of-the-art model for sentence pair classification.

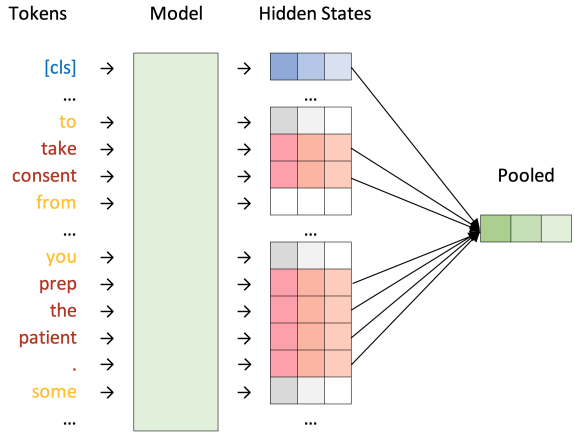

Figure 4: **Visualization of the hidden state masking.** Hidden states for the context sentences are masked before pooling.

Hidden states masking. In this model variant we inject the context segmentation into models by masking out the final layer hidden states for the context sentences and aggregate the remaining hidden states using a pooling function. This structure enables us to incorporate context segmentation information without introducing any new parameters.

$$H_{\mathbf{t}} = \{\mathbf{h}_i | t_i \in \mathbf{t}\}$$
$$\mathbf{h}_{\text{MAX}} = \max(H_{\mathbf{u}_t} \cup H_{\mathbf{v}_t} \cup h_{[cls]} \cup h_{[sep]}) \quad (1)$$
$$\mathbf{h}_{\text{AVG}} = \frac{\sum H_{\mathbf{u}_t} + \sum H_{\mathbf{v}_t} + h_{[cls]} + h_{[sep]}}{|\mathbf{u}_t| + |\mathbf{v}_t| + 2} \quad (2)$$

where $\{\mathbf{h}\}$ are the final layer hidden states, $\mathbf{u}_t$, $\mathbf{v}_t$ are the two tokenized text spans, $t_{[cls]}, t_{[sep]}$ are the $[cls]$ token and $[sep]$ token (for BERT model), $h_{[cls]}, h_{[sep]}$ are the corresponding hidden states, respectively.

Import context position as attention. Inspired by position embedding and position-aware attention (Zeng et al., 2014; Zhang et al., 2017), we define two context position sequences $[c_1, \cdots, c_n]$ and $[c'_1, \cdots, c'_n]$, which correspond to the position of the two text spans, respectively, that is:

$$c_i = \begin{cases} p_i - p_t - 1, & p_i < p_t \\ 1 \text{ or } -1, & p_i = p_t \\ p_i - p_t + 1, & p_i > p_t \end{cases} \quad (3)$$

We use p_i and p_t to denote the position of context and text span in transcript in sentence level. For $i = p_t$, $c_i = 1$ or -1, depends on whether the context is on the left or right of the text span. The two context position sequences are truncated by a fix length for computational complexity, then injected into BERT model using position-aware attention (Zhang et al., 2017).

Import context position as input embedding. Segment embedding is a part of input embedding designed to import sentence-pair segmentation information in BERT model. In this model variant we expand the segment embedding to encode context position sequence above.

4 Dataset Generation

To take advantage of the weak supervision from protocols, we build a pipeline to generate datasets for the CTA parsing framework, showed in Fig. 5.

4.1 Protocol Parsing

We use a deterministic finite automation to parse the protocol into a graph $\mathcal{G}_p = (\mathcal{P}, \mathcal{R}_p)$ describing protocol phrases represented by nodes ($\mathcal{P} = \{\mathbf{p}_i\}$ which denotes all protocol phrases parsed from the protocol) and procedural relations represented by edges ($\mathcal{R}_p = \{(\mathbf{u}_{p_i}, \mathbf{v}_{p_i}, r_{p_i})\}$, where $\mathbf{u}_{p_i}, \mathbf{v}_{p_i} \in \mathcal{P}$ are protocol phrases and r_{p_i} is the procedural relation from phrase $\mathbf{u}_{p_i}$ to phrase $\mathbf{v}_{p_i}$).

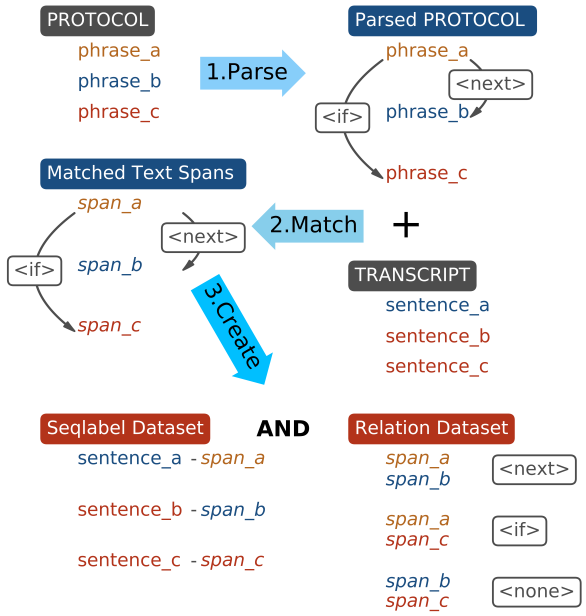

Figure 5: **The dataset generation pipeline.** The protocol is first parsed into a graph with relations between protocol phrases (shown as *phrase*), then match the protocol phrases with the text spans in transcripts (shown as *span*). Finally, sequence labeling dataset and span-pair RE dataset are created according to the matches and the relations.

We consider three types of procedural relations during the parsing: ⟨none⟩ for no procedural relation between protocol phrases, ⟨next⟩ for sequence, and ⟨if⟩ for decision branching.

4.2 Text Spans Matching

To enable the abundant information in protocols, we want to map each phrase in the protocol back to the nearest textual representation in the transcript. We can achieve this by using sentence matching techniques. Following the sequence labeling setting in transcript summarization of our framework, given a protocol phrase $\mathbf{p}$, we want to find the best matching text span $\mathbf{t}$ in the transcript. The scope of search is limited to the source lines $\mathcal{L}_\mathbf{p}$ mentioned in the protocol (Fig. 1). Then we extract all possible text spans $\{\mathbf{t}_i\}$ from these sentences by enumerating all available n-grams and find the best matching text span $\mathbf{t}_{\text{best}}$ for $\mathbf{p}$ that maximizes sentence similarity measure $\mathcal{M}_{sim}(\mathbf{p}, \mathbf{t}_{\text{best}})$. Following is the overall workflow:

$$\begin{aligned}
\mathcal{S} &= \{\text{retrieve_sentence}(\ell) | \ell \in \mathcal{L}_\mathbf{p}\} \\
\mathbf{t}_{\text{best}} &= \arg\max_{\mathbf{t}\in\mathcal{S}} \mathcal{M}_{\text{sim}}(\mathbf{p}, \mathbf{t}) \\
\mathcal{M}_{\text{sim best}} &= \max_{\mathbf{t}\in\mathcal{S}} \mathcal{M}_{\text{sim}}(\mathbf{p}, \mathbf{t}) \\
\text{match} &= \begin{cases} \text{None} & \mathcal{M}_{\text{sim best}} \leq \text{threshold} \\ \mathbf{t}_{\text{best}} & \mathcal{M}_{\text{sim best}} > \text{threshold} \end{cases}
\end{aligned} \tag{4}$$

For the similarity measure $\mathcal{M}_{\text{sim}}$, we adopt sentence embedding from different methods (Pennington et al., 2014; Devlin et al., 2018). The similarity is calculated by the cosine distance between two normalized sentence embedding. An empirical threshold 0.5 is adopted for dropping the protocol phrases without good matched text span. We then match the protocol phrases back to the nearest text span in the transcript.

4.3 Sequence Labeling Dataset

With the matched text spans in the transcript, we are able to assign labels to every token in the transcript, denoting whether the token belongs to a matched text span. We adopt IOBES format (Ramshaw and Marcus, 1999) as the labeling schema for constructing the sequence labeling dataset. The labeled text spans are semantically close to the protocol phrases which are abstractive description of actions, and we can use the labels to train text spans extraction models (Sec. 3.1) in a weakly-supervised manner.

4.4 Text Span-Pair Relation Extraction Dataset

By parsing the protocol we learn the procedural relations between protocol phrases. Thus we can apply them to the matched text spans in transcript to construct the span-pair RE dataset. These relations serve as weak supervision for the span-pair RE model (Sec. 3.2). Corresponding to the relation types parsed from the protocols, the dataset include three types of label: <none>, <next> and <if>.

4.5 Human-Annotated Matching Test Set

Since the datasets for CTA transcript parsing framework are created via matching, we need to evaluate the performance of our matching methods. Thus, for testing purpose, we manually annotated the matched text spans in transcript for 138 protocol phrases as the manual matching test set.

Furthermore, we create two test sets to evaluate the effectiveness of our approach with the manual matching annotations, which are called manual matching sequence labeling test set and manual matching span-pair RE test set. In comparison, we call the test sets generated via text spans matching as generated sequence labeling test set and generated span-pair RE test set.

5 Experiments

In this section we evaluate the effectiveness of our proposed automated CTA transcript parsing framework and the models. Especially, we run three sets of experiments: (1) we evaluate our text spans matching methods with the manual matching test set; (2) we evaluate model performance on the CTA text spans extraction task with the sequence labeling dataset; (3) we evaluate model performance on the CTA span-pair RE task with the RE dataset.

5.1 Text Spans Matching

Implementation. We enumerate all text spans with length $[2, K_t]$ within the sentences in transcripts, where $K_t = 30$ for truncating text spans. For text spans matching, we try two sentence encoding methods to extract sentence embeddings: (1) average pooling on Glove word embeddings of words in sentences and text spans (Pennington et al., 2014); (2) extracting features using pre-trained $\text{BERT}_{\text{BASE}}$ model and sum up the features in the last four layers then average over words in sentences and text spans (Devlin et al., 2018). Then, we normalize the embeddings and find the best matching text spans for each protocol phrase based on cosine similarity. We also provide the exact matching as a baseline, which finds the longest transcript text span matched by a text span in protocol phrase.

Encoding	Tok. Acc.	Tok. F_1	Men. Acc.
Exact	70.30	9.44	2.17
Glove-50d	75.40	43.92	37.68
Glove-300d	**76.97**	**45.12**	42.03
BERT fea.	75.22	37.20	**47.10**

Table 1: **Matching performance on the manual matching testset with different sentence encoding,** in token level accuracy and mention level accuracy. BERT fea. means using features extracted by BERT model. and Exact is the exact matching baseline

Evaluation. We evaluate the performance of our text spans matching methods with the manual matching test set by token level metrics and mention level accuracy, where token level metrics are normalized by sentence lengths. Results in Table 1 show the two methods get acceptable results while the exact matching baseline has a poor performance in comparison. Glove-300d shows better token level accuracy and F_1 score while BERT features have a better mention level accuracy. For cheaper computation, we use Glove-300d as the sentence encoding method of matching for the following sections. Please refer to the appendix for the case study of matching.

5.2 Text Spans Extraction

Models. We conduct the experiments of text spans extraction using off-the-shelf sequence labeling models, including CRF (Finkel et al., 2005), LSTM-CRF (Huang et al., 2015) and LM-LSTM-CRF (Liu et al., 2017). The models are trained on the sequence labeling dataset generated by text spans matching. For comparison, we also implement a hand-crafted rule extraction baseline with TokensRegex.

LSTM-CRF and LM-LSTM-CRF. We use LM-LSTM-CRF[2] to conduct our experiments for both models, with the same setting of 2 layers word level LSTM, word level hidden size $H_w = 300$, SGD with 0.045 learning rate and 0.05 learning rate decay, and 0.3 dropout ratio. The major difference between two models is that LM-LSTM-CRF contains an additional char-level structure optimized via language model loss.

Model	Tok. Acc.	Men. P	Men. R	Men. F_1
Rules	-	12.7	34.8	18.6
CRF	80.7	38.5	37.9	**38.1**
LSTM-CRF	75.9	40.4	31.8	35.6
w/ LM_{16}	74.6	31.8	21.2	25.5
w/ LM_{64}	74.8	33.3	18.2	23.5

Table 2: **Performance of sequence labeling models**, evaluated on manual matching testset. LM-LSTM-CRF is shown as w/ LM, with different character level hidden size.

Evaluation. Results for the text spans extraction models on manual matching test set are presented in Table 2, which shows that CRF achieves the best performance and outperforms the neural

[2]https://github.com/LiyuanLucasLiu/LM-LSTM-CRF

models (LSTM-CRF, LM-LSTM-CRF). The LM-LSTM-CRF which contains character level language model is even worse (shown as w/ LM in table, with different character level hidden size). One reason could be the neural models require a large scale dataset to train, while our dataset does not meet this requirement.

5.3 Text Span-Pair Relation Extraction

Models. For text span-pair relation extraction, we use the pre-trained $\text{BERT}_{\text{BASE}}$ model[3] (Devlin et al., 2018) as our backbone model to address the low-resource issue of our RE dataset generated from the limited CTA data. On this basis, we implement model variants of injecting context information awareness (Sec. 3.3) to utilize the full information in our dataset, which includes: hidden states Masking (Mask_{AVG} and Mask_{MAX}), Context position as Attention (C. Attn.) and Context postion as input Embedding (C. Emb.). For hidden states Masking, the different subscriptions represent different hidden state pooling methods (avg pooling and max pooling) For the two models using context position, we empirically use $E = 30$ as the embedding size and truncate the context position sequence (Sec. 3.3) by ± 10. In addition, we experiment on the hierarchical BiLSTM model (Talman et al., 2018) and Piecewise Convolution Neural Network (Zeng et al., 2015) as the non-pretrained baseline models in comparison. Results are aggregated from 5 runs with different initialization seeds for all experiments.

Sampling portion	Total	<next>	<if>
w/o sampling	138670	693	131
6 : 3 : 1	1310	393	131
4 : 2 : 1	917	262	131
1 : 1 : 1	393	131	131

Table 3: **Size of span-pair RE dataset**, by different sampling portion <none>:<next>:<if>

Label sampling portion. The generated RE dataset has three types of label: <none>, <next> and <if>, with a bias label distribution (Fig. 3, w/o sampling). To leverage this, we do label sampling on the dataset.

Context level. To capture the long-range context information useful to the CTA transcript parsing task, we use text spans with context $\mathbf{t}_c$ (fig. 3) as the input of models. The level of context is controlled by a hyperparameter K (Fig. 3). We experiment our models with different levels of context, while fixing the label sampling portion (Sec. 5.3) to <none> : <next> : <if>$= 4 : 2 : 1$.

[3]https://github.com/huggingface/pytorch-pretrained-BERT

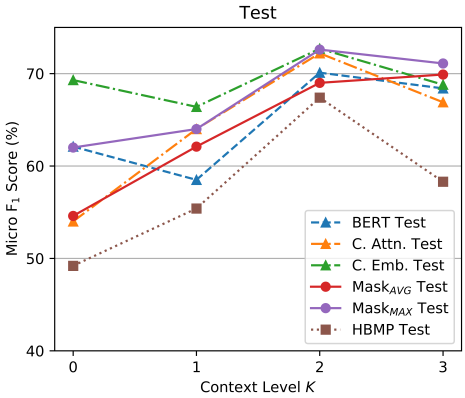

Figure 6: **The micro $\mathbf{F_1}$ score of models on different context level K,** evaluated on generated test set.

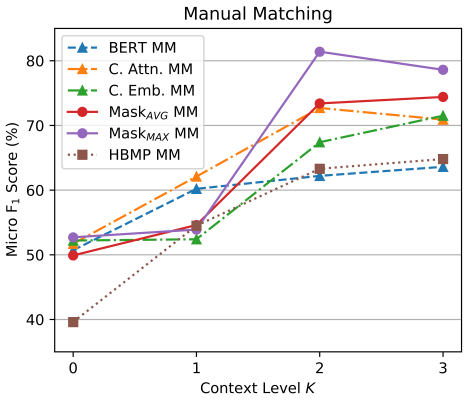

Figure 7: **The micro $\mathbf{F_1}$ score of models on different context level K,** evaluated on manual matching test set.

Evaluation. The results are available in Table 4, which shows the model we proposed can outperform the baselines (BERT, HBMP, PCNN), and the model variant Mask_{MAX} reach best performance among all variants when using context level $K = 2$ and sampling portion $= 4 : 2 : 1$.

Evaluation on context level. The short version table of evaluation results for different context levels are shown in Table 5 and please refer to the appendix for the full version. The results are visualized in Fig. 7 and Fig. 6. The model Mask_{MAX} reached the best micro F_1 score on the manual matching test set with context level $K = 2$ over all models and K, which shows the effectiveness of the span-pair RE and the hidden state masking

Setting	Generated Test Set				Manual Matching Test Set			
	Accuracy	Micro F_1	<next> F_1	<if> F_1	Accuracy	Micro F_1	<next> F_1	<if> F_1
BERT	81.6 ±1.0	70.1 ±1.7	67.9 ±3.2	73.4 ±2.2	77.2 ±2.7	62.2 ±6.1	57.6 ±6.4	72.4 ±10.0
HBMP	76.0	67.4	-	-	72.0	63.3	-	-
PCNN	58	40	-	-	56	43	-	-
C. Attn.	82.5 ±1.5	72.2 ±2.6	**70.9 ±1.8**	74.7 ±4.4	81.2 ±4.7	72.7 ±7.5	68.7 ±9.1	**83.3 ±5.5**
C. Emb.	**82.8 ±1.4**	72.7 ±1.9	70.7 ±2.8	76.3 ±2.5	78.8 ±8.5	67.4 ±8.1	66.2 ±9.2	67.5 ±19.4
Mask_{AVG}	80.5 ±2.7	69.0 ±5.7	63.6 ±7.1	**77.0 ±4.8**	80.4 ±7.1	73.4 ±7.9	71.8 ±9.4	79.2 ±14.5
Mask_{MAX}	82.3 ±1.4	**72.6 ±3.0**	70.7 ±3.2	76.1 ±3.1	**87.6 ±1.5**	**81.4 ±2.4**	**80.8 ±1.9**	83.3 ±6.8

Table 4: **Performance of span-pair RE models,** with sampling portion $4:2:1$ and $K = 2$. Evaluated on generated test set and manual matching test set.

Setting	Generated Test Set				Manual Matching Test Set			
	Accuracy	Micro F_1	<next> F_1	<if> F_1	Accuracy	Micro F_1	<next> F_1	<if> F_1
$\text{BERT}_{\text{K=3}}$	80.2 ±3.2	68.4 ±4.3	64.3 ±6.6	74.6 ±4.3	77.2 ±3.0	63.6 ±5.5	60.9 ±7.2	70.3 ±11.3
$\text{BERT}_{\text{K=2}}$	81.6 ±1.0	70.1 ±1.7	67.9 ±3.2	73.4 ±2.2	77.2 ±2.7	62.2 ±6.1	57.6 ±6.4	72.4 ±10.0
$\text{BERT}_{\text{K=1}}$	73.5 ±2.7	58.5 ±3.0	57.7 ±4.7	60.0 ±4.0	76.4 ±2.3	60.2 ±6.1	54.5 ±7.0	76.1 ±7.2
$\text{BERT}_{\text{K=0}}$	71.9 ±2.6	62.1 ±5.1	58.5 ±6.9	69.0 ±6.9	63.2 ±5.2	50.7 ±6.8	45.3 ±10.2	71.0 ±10.7
$\text{Mask}_{\text{MAX K=3}}$	81.8 ±0.9	71.1 ±1.5	68.6 ±1.9	75.3 ±2.2	85.2 ±4.1	78.6 ±4.8	75.6 ±6.1	**88.9 ±0.0**
$\text{Mask}_{\text{MAX K=2}}$	**82.3 ±1.4**	**72.6 ±3.0**	**70.7 ±3.2**	**76.1 ±3.1**	**87.6 ±1.5**	**81.4 ±2.4**	**80.8 ±1.9**	83.3 ±6.8
$\text{Mask}_{\text{MAX K=1}}$	76.3 ±1.3	64.0 ±1.9	58.4 ±3.4	74.2 ±1.6	69.2 ±1.0	53.9 ±1.5	47.6 ±2.5	75.0 ±0.0
$\text{Mask}_{\text{MAX K=0}}$	71.9 ±2.7	62.0 ±4.6	55.3 ±7.5	73.9 ±2.9	64.4 ±2.7	52.7 ±4.2	47.6 ±5.6	71.7 ±4.1

Table 5: **Performance of span-pair RE models on different context level** K, with sampling portion $4:2:1$. Evaluated on generated test set and manual matching test set.

structure.

Portion	Model	Micro F_1	
		Generated	Manual
6 : 3 : 1	BERT	67.6 ±1.7	69.5 ±4.6
	Mask_{MAX}	68.5 ±2.1	71.1 ±9.1
4 : 2 : 1	BERT	68.4 ±4.3	63.6 ±5.5
	Mask_{MAX}	71.1 ±1.5	**78.6 ±4.8**
1 : 1 : 1	BERT	65.4 ±4.9	62.7 ±3.4
	Mask_{MAX}	70.3 ±2.9	69.8 ±3.5

Table 6: **Performance on text spans relation extraction models on different label sampling settings,** with $K = 3$. *Generated* represent the sampled generated test set follows the sampling portion the model trained on, while *Manual* represents the manual matching test set which is fixed to $6:3:1$.

Evaluation on label sampling. We try 3 sampling settings and find <none>:<next>:<if>$= 4:2:1$ shows the best performance on manual matching test set for most cases (Table 6). Please refer to the appendix for the full results on label sampling.

Discussion. We have some observations when looking through the results on manual matching test set: (1) The model variants injected with context information awareness are more sensitive to the change of context level K, comparing to the vanilla BERT model. These variants are outperforming the vanilla model when provided with more context, but would fall behind if provided with short even no context. (2) Vanilla models without specific context awareness structures (BERT, HBMP, PCNN) also gain improvements from the context on the manual matching test set. (3) A big gap of <next> F_1 score between $K = 1$ and $K = 2$ are observed in most of the models. This is because when $K = 1$ context only provide the sentence enclosing the text span, the $K = 2$ context is providing the last and the next sentence, which is useful for predicting the <next> relation.

The results on generated test set (Fig.6) is also interesting, in which the performance is not stably increased as the K increasing. This may be caused by the propagation of error from fuzzy matching. Since there are some error (noisy) samples in the generated dataset, the models are more likely to capture the noisy patterns from the noisy samples. The larger the context is, the more noisy patterns are contained. Still, changing K from $K = 1$ to $K = 2$ gives noticeable improvement to all models, especially for the <next> F_1 score.

Also, the experiments on label sampling (Table 6, see appendix for the full result) show the performance of models are sensitive to sampling portion. Resampling and reweighting techniques for allevi-

ating label imbalance could be helpful to address such problem in future study.

6 Conclusion

In this paper, we explored automated CTA transcript parsing, which is a challenging task due to the lack of direct supervision data and the requirement of document level understanding. We proposed a weakly supervised framework to utilize the full information in data. We noticed the importance of context in the CTA parsing task and exploited model variants to make use of context information. Our evaluation on manually labeled test set shows the effectiveness of our framework.

7 Acknowledgment

This work has been supported in part by National Science Foundation SMA 18-29268, Schmidt Family Foundation, Amazon Faculty Award, Google Research Award, and JP Morgan AI Research Award. We would like to thank all the collaborators in INK research lab for their constructive feedback on the work. We thank the anonymous reviewers for their valuable feedback.

References

Tim Salimans Alec Radford, Karthik Narasimhan and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. *Technical report, OpenAI*.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(Nov):1817–1853.

Devendra Singh Chaplot, Christopher MacLellan, Ruslan Salakhutdinov, and Kenneth R. Koedinger. 2018. Learning cognitive models using neural networks. *CoRR*, abs/1806.08065.

Richard E Clark and Fred Estes. 1996. Cognitive task analysis for training. *International Journal of Educational Research*, 25(5):403–417.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Nan Li, Eliane Stampfer, William Cohen, and Kenneth Koedinger. 2013. General and efficient cognitive model discovery using a simulated student. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 35.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2018. Generative adversarial network for abstractive text summarization.

Liyuan Liu, Jingbo Shang, Frank F. Xu, Xiang Ren, Huan Gui, Jian Peng, and Jiawei Han. 2017. Empower sequence labeling with task-aware neural language model. *CoRR*, abs/1709.04109.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Hogun Park and Hamid Reza Motahari Nezhad. 2018. Learning procedures from text: Codifying how-to procedures in deep neural networks. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pages 351–358, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Lance A Ramshaw and Mitchell P Marcus. 1999. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Kaitlyn Roose, Elizabeth Veinott, and Shane Mueller. 2018. The tracer method: The dynamic duo combining cognitive task analysis and eye tracking.

Jan Maarten Schraagen, Susan F Chipman, and Valerie L Shalin. 2000. *Cognitive task analysis*. Psychology Press.

Thomas L Seamster and Richard E Redding. 2017. *Applied cognitive task analysis in aviation*. Routledge.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document summarization using conditional random fields. In *IJCAI*, volume 7, pages 2862–2867.

Aarne Talman, Anssi Yli-Jyr, and Jrg Tiedemann. 2018. Natural language inference with hierarchical bilstm max pooling architecture.

David D Woods et al. 1989. Cognitive task analysis: An approach to knowledge acquisition for intelligent system design. In *Studies in Computer Science and Artificial Intelligence*, volume 5, pages 233–264. Elsevier.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45.

Chen Zhong, John Yen, Peng Liu, Rob Erbacher, Renee Etoty, and Christopher Garneau. 2015. An integrated computer-aided cognitive task analysis method for tracing cyber-attack analysis processes. In *Proceedings of the 2015 Symposium and Bootcamp on the Science of Security*, HotSoS '15, pages 9:1–9:11, New York, NY, USA. ACM.

#	Protocol Phrase Matched Text Span	Matching status
1	with your non-dominant hand And now with your non-dominant hand you	over
2	Pass wire pass wire	correct
3	resistance -	noisy
4	remove the wire by bringing it back into the housing remove the wire , bring it back into the little housing	correct
5	Put syringe back on and confirm that there is still blood flow syringe back on and make sure you still have good flow	miss
6	Remove needle remove your needle	correct
7	leave wire in place Leave the wire in place	correct
8	Make a nick in the skin that is wide enough for the catheter make a nick in the skin wide enough for whatever catheter	correct
9	Pass dilator pass dilator	correct
10	Remove dilator out dilator	miss
11	while holding the wire in place leaving the wire in place , always holding onto the wire	correct
12	Put catheter through the wire putting the catheter through the wire	correct
13	Remove wire remove the wire	correct
14	Check and irrigate all ports So we have task one , decide on location	wrong
15	Lock the catheter lock the catheter	correct
16	attach with Luer-lock lock the catheter	wrong
17	Suture in place Task five , insert needle	wrong
18	Verify placement with x-ray Verify placement with	miss
19	Prepare patient prep the patient	correct
20	self -	noisy

Table 7: **Case study for text span matching, using Glove-300d as the sentence encoding method.**

Setting	Generated Test Set				Manual Matching Test Set			
	Accuracy	Micro F_1	<next> F_1	<if> F_1	Accuracy	Micro F_1	<next> F_1	<if> F_1
BERT $_{K=3}$	80.2 ±3.2	68.4 ±4.3	64.3 ±6.6	74.6 ±4.3	77.2 ±3.0	63.6 ±5.5	60.9 ±7.2	70.3 ±11.3
BERT $_{K=2}$	81.6 ±1.0	70.1 ±1.7	67.9 ±3.2	73.4 ±2.2	77.2 ±2.7	62.2 ±6.1	57.6 ±6.4	72.4 ±10.0
BERT $_{K=1}$	73.5 ±2.7	58.5 ±3.0	57.7 ±4.7	60.0 ±4.0	76.4 ±2.3	60.2 ±6.1	54.5 ±7.0	76.1 ±7.2
BERT $_{K=0}$	71.9 ±2.6	62.1 ±5.1	58.5 ±6.9	69.0 ±6.9	63.2 ±5.2	50.7 ±6.8	45.3 ±10.2	71.0 ±10.7
C. Attn. $_{K=3}$	80.2 ±3.7	66.9 ±5.6	66.1 ±6.9	68.0 ±5.4	81.6 ±4.1	70.9 ±8.1	67.9 ±10.1	79.8 ±5.1
C. Attn. $_{K=2}$	82.5 ±1.5	72.2 ±2.6	70.9 ±1.8	74.7 ±4.4	81.2 ±4.7	72.7 ±7.5	68.7 ±9.1	83.3 ±5.5
C. Attn. $_{K=1}$	75.4 ±2.1	64.0 ±3.8	58.8 ±4.6	73.5 ±2.5	74.0 ±2.5	62.1 ±2.5	53.1 ±4.0	86.6 ±8.6
C. Attn. $_{K=0}$	67.3 ±3.1	54.0 ±4.6	43.7 ±7.5	72.5 ±2.3	58.8 ±3.7	51.7 ±4.0	43.6 ±2.9	83.3 ±6.8
C. Emb. $_{K=3}$	79.8 ±2.5	68.8 ±4.0	64.0 ±4.6	76.9 ±3.8	80.4 ±7.1	71.5 ±10.1	68.4 ±9.4	81.2 ±12.6
C. Emb. $_{K=2}$	**82.8 ±1.4**	**72.7 ±1.9**	**70.7 ±2.8**	76.3 ±2.5	78.8 ±8.5	67.4 ±8.1	66.2 ±9.2	67.5 ±19.4
C. Emb. $_{K=1}$	76.5 ±2.3	66.4 ±3.5	62.1 ±3.5	74.4 ±5.0	67.6 ±7.7	52.4 ±10.7	43.3 ±11.7	79.8 ±6.4
C. Emb. $_{K=0}$	77.5 ±1.2	69.3 ±2.9	63.9 ±3.0	**78.4 ±6.0**	67.6 ±6.6	52.2 ±6.6	40.7 ±5.5	83.7 ±7.2
Mask_{AVG} $_{K=3}$	81.4 ±1.3	69.9 ±3.4	67.6 ±2.6	73.6 ±6.3	81.6 ±3.2	74.4 ±7.2	71.0 ±8.1	86.1 ±5.6
Mask_{AVG} $_{K=2}$	80.5 ±2.7	69.0 ±5.7	63.6 ±7.1	77.0 ±4.8	80.4 ±7.1	73.4 ±7.9	71.8 ±9.4	79.2 ±14.5
Mask_{AVG} $_{K=1}$	74.7 ±1.2	62.1 ±2.1	55.9 ±2.4	73.2 ±2.4	71.2 ±3.2	54.6 ±4.4	46.8 ±5.1	77.8 ±5.6
Mask_{AVG} $_{K=0}$	67.1 ±2.2	54.6 ±3.2	45.7 ±5.5	71.1 ±2.0	59.2 ±4.1	49.9 ±3.6	42.0 ±4.8	80.6 ±6.8
Mask_{MAX} $_{K=3}$	81.8 ±0.9	71.1 ±1.5	68.6 ±1.9	75.3 ±2.2	85.2 ±4.1	78.6 ±4.8	75.6 ±6.1	**88.9 ±0.0**
Mask_{MAX} $_{K=2}$	82.3 ±1.4	72.6 ±3.0	70.7 ±3.2	76.1 ±3.1	**87.6 ±1.5**	**81.4 ±2.4**	**80.8 ±1.9**	83.3 ±6.8
Mask_{MAX} $_{K=1}$	76.3 ±1.3	64.0 ±1.9	58.4 ±3.4	74.2 ±1.6	69.2 ±1.0	53.9 ±1.5	47.6 ±2.5	75.0 ±0.0
Mask_{MAX} $_{K=0}$	71.9 ±2.7	62.0 ±4.6	55.3 ±7.5	73.9 ±2.9	64.4 ±2.7	52.7 ±4.2	47.6 ±5.6	71.7 ±4.1
HBMP $_{K=3}$	68.0	58.3	-	-	74.0	64.8	-	-
HBMP $_{K=2}$	76.0	67.4	-	-	72.0	63.3	-	-
HBMP $_{K=1}$	67.0	55.4	-	-	60.0	54.5	-	-
HBMP $_{K=0}$	50.0	49.2	-	-	50.0	39.6	-	-
PCNN $_{K=3}$	47	31	-	-	62	48	-	-
PCNN $_{K=2}$	58	40	-	-	56	43	-	-
PCNN $_{K=1}$	44	28	-	-	50	28	-	-
PCNN $_{K=0}$	44	29	-	-	34	24	-	-

Table 8: **Performance of text spans relation extraction models on different context level** K, with sampling portion $4:2:1$.

Model	Sampled Generated Test Set				Manual Matching Test Set			
	Accuracy	Micro F_1	<next> F_1	<if> F_1	Accuracy	Micro F_1	<next> F_1	<if> F_1
Sampling portion = $6:3:1$ (1.3k samples)								
BERT	79.0 ±1.2	67.6 ±1.7	68.3 ±1.3	65.5 ±3.2	80.0 ±2.5	69.5 ±4.6	72.2 ±1.8	52.0 ±31.1
C. Attn.	75.6 ±2.4	61.4 ±4.0	62.9 ±4.0	57.4 ±6.0	80.4 ±4.3	68.8 ±7.4	66.4 ±8.9	75.2 ±10.4
C. Emb.	77.9 ±1.8	65.7 ±1.2	66.4 ±2.3	64.0 ±4.1	80.8 ±2.0	70.7 ±4.1	70.0 ±5.4	71.8 ±8.8
Mask_{AVG}	79.8 ±1.0	69.1 ±2.4	68.7 ±2.2	70.4 ±3.4	81.6 ±3.4	72.3 ±6.5	69.0 ±6.9	83.3 ±6.8
Mask_{MAX}	80.1 ±0.8	68.5 ±2.1	**69.5 ±2.8**	65.9 ±1.9	81.6 ±5.0	71.1 ±9.1	66.9 ±11.6	83.3 ±6.8
Sampling portion = $4:2:1$ (0.9k samples)								
BERT	80.2 ±3.2	68.4 ±4.3	64.3 ±6.6	74.6 ±4.3	77.2 ±3.0	63.6 ±5.5	60.9 ±7.2	70.3 ±11.3
C. Attn.	80.2 ±3.7	66.9 ±5.6	66.1 ±6.9	68.0 ±5.4	81.6 ±4.1	70.9 ±8.1	67.9 ±10.1	79.8 ±5.1
C. Emb.	79.8 ±2.5	68.8 ±4.0	64.0 ±4.6	76.9 ±3.8	80.4 ±7.1	71.5 ±10.1	68.4 ±9.4	81.2 ±12.6
Mask_{AVG}	81.4 ±1.3	69.9 ±3.4	67.6 ±2.6	73.6 ±6.3	81.6 ±3.2	74.4 ±7.2	71.0 ±8.1	86.1 ±5.6
Mask_{MAX}	**81.8 ±0.9**	**71.1 ±1.5**	68.6 ±1.9	75.3 ±2.2	**85.2 ±4.1**	**78.6 ±4.8**	**75.6 ±6.1**	**88.9 ±0.0**
Sampling portion = $1:1:1$ (0.4k samples)								
BERT	64.6 ±4.6	65.4 ±4.9	51.3 ±3.5	79.3 ±6.0	69.6 ±5.6	62.7 ±3.4	54.3 ±3.6	87.3 ±7.3
Mask_{AVG}	64.6 ±2.9	64.2 ±3.4	46.7 ±7.4	79.6 ±1.7	72.8 ±1.6	63.5 ±2.3	57.4 ±2.3	83.6 ±4.4
Mask_{MAX}	68.8 ±3.5	70.3 ±2.9	55.6 ±4.6	**83.9 ±1.2**	77.6 ±3.2	69.8 ±3.5	63.2 ±2.4	88.4 ±7.6

Table 9: **Performance on text spans RE models on different label sampling settings,** with $K = 3$. Sampled generated test set follows the sampling portion the model trained on while manual matching test set is fixed.