DOCUMENT-LEVEL EVENT EXTRACTION VIA HUMAN-LIKE READING PROCESS

Shiyao Cui[1,2], *Xin Cong*[1,2], *Bowen Yu*[1,2], *Tingwen Liu*[*1,2], *Yucheng Wang*[1], *Jinqiao Shi*[3]

[1]Institute of Information Engineering, Chinese Academy of Sciences. Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences. Beijing, China
[3] Beijing University of Posts and Telecommunications. Beijing, China

ABSTRACT

Document-level Event Extraction (DEE) is particularly tricky due to the two challenges it poses: *scattering-arguments* and *multi-events*. The first challenge means that arguments of one event record could reside in different sentences in the document, while the second one reflects that one document may simultaneously contain multiple such event records. Motivated by humans' reading cognitive to extract information of interests, in this paper, we propose a method called HRE (Human Reading inspired Extractor for Document Events), where DEE is decomposed into these two iterative stages, *rough reading* and *elaborate reading*. Specifically, the first stage browses the document to detect the occurrence of events, and the second stage serves to extract specific event arguments. For each concrete event role, elaborate reading hierarchically works from sentences to characters to locate arguments across sentences, thus the scattering-arguments problem is tackled. Meanwhile, rough reading is explored in a multi-round manner to discover undetected events, thus the multi-events problem is handled. Experiment results show the superiority of HRE over prior competitive methods.

Index Terms— Natural Language Processing, Information Extraction, Event Extraction, Document-level Event Extraction

1. INTRODUCTION

Event Extraction (EE) aims to recognize the specific type of events and extract the corresponding event arguments from given texts. Despite successful efforts [1, 2, 3, 4, 5, 6, 7] to extract events within a sentence, a.k.a. the Sentence-level EE (SEE), these methods seem to struggle in real-world scenarios where events are usually expressed across sentences. Hence, SEE is moving forward to its document-level counterpart, a.k.a. the Document-level EE (DEE).

Typically, DEE faces two challenges, *scattering-arguments* and *multi-events*. As Fig.1 shows, the first challenge indicates that event arguments of one event record may reside in different sentences, thus an event cannot be extracted from a single sentence. The second one reflects that the document may simultaneously contain multiple such event records, which demands a holistic comprehension to the document and understanding to the inter-dependency between events. To date, most DEE methods [8, 9, 10, 11] mainly focus on the first challenge but ignores the second one. Though Zheng et al. (2019) [12] first propose Doc2EDAG to simultaneously tackle the both challenges, the proposed entity-oriented method insufficiently model the dependency between multi-events, resulting in the final performances being unsatisfactory.

Recently, simulating human's reading cognitive process to address specific natural language processing (NLP) tasks [13, 14] has achieved great success. Normally, three stages [15, 16, 17] are involved in humans' reading process: **pre-reading**, **careful reading**, and **post-reading**. During pre-reading, human readers preview the whole document, forming a general cognition to the document content. During careful reading, human readers attentively read each sentence to locate detailed information according to their specific reading purpose. During post-reading, a review is applied to the document, checking missed details and completing the comprehension to document. The multi-stage reading process coarse-to-finely comprehends the document, which makes it effective to extract event facts throughout the document. Still, by far, few references have explored such a reading process in DEE.

To model the above ideas, in this paper, we propose a DEE method called **HRE** (Human Reading inspired Extractor for Document Events), which reformats human's reading process to two stages, **rough reading** and **elaborate reading**. For each specific event type, rough reading is first conducted to detect the occurrence of events. If an event is detected, elaborate reading is applied to extract the complete event record in an argument-wise manner. Concretely, for each event role, elaborate reading first locates in which sentence the corresponding argument resides, and then extracts it. Each extracted event argument is stored by a memory mechanism, which models the inter-dependency between multi-events and enables HRE to be aware of prior extracted events. After one complete event record is obtained, HRE again roughly reads the document to check missing events of the same event type, and the memory mechanism empowers it to detect events without redundancy with previous extracted ones. If another event occurrence is perceived, elaborate reading will be applied again, otherwise, HRE moves to dealing with the next event type via the same logic above until all

[S1.] On Oct. 12th, 2016, HaiTong Co., Ltd received a pledge from Mr. Liu Baichun, the shareholder of Zhengxing Co., Ltd. **[S2.]** Mr. Liu Baichun pledged his 4910000 shares to HaiTong Co., Ltd as liability guarantee. **.....** **[S4.]** Due to market volatility, HaiTong Co., Ltd again asked Mr. Liu Baichun for a pledge of 800000 shares. **[S5.]** On Feb.1st,2018, the pledge of 800000 shares is transfered to HaiTong Co., Ltd **.....** **[S8.]** Since then, Mr. Liu Baichun have pledged 5710000 shares in total and plan to repurchase in Aug. 1st, 2018.

	Examples of Evnt Records				
Event Role →	Pledger	PledgedShares	Pledgee	Begin Date	...
Event Argument →	Liu Baichun	4910000shares	HaiTong Co., Ltd	Oct.12th, 2016	...
Event Record →	Liu Baichun	800000shares	HaiTong Co., Ltd	Feb.1st, 2018	...

Fig. 1. A document example with two *Equity Pledge*-type event records. For each event record, its arguments reside in multiple sentences. Due to space limitation, we only show associated sentences and four event roles of each event. The original corpus is in Chinese, and for clarity, we translate it into English.

*Corresponding author.

event types are tackled. With the memory mechanism modeling the inter-dependency between multiple events, the multi-round exploration to rough reading frees the *multi-events* challenge. Meanwhile, for each event role, the elaborate reading respectively searches the argument-specific sentence, where arguments across sentences could all be located and *scattering-argument* problem is naturally handled. Experiment results on the largest DEE dataset [12] suggest that HRE achieves a new state-of-the-art performance.

2. METHOD

The ultimate goal of DEE is to extract all the event records, with the correct judgement of the event type and arguments. Algirithm 1 demonstrates the overall working flow of HRE. Note that a **memory mechanism** is designed to work throughout the two reading stages, where, for each event type, we use a trainable event-type-specific embedding $\mathbf{e}_e$ to initialize a memory tensor $\mathbf{m}_e$, and $\mathbf{m}_e$ is updated by appending the following extracted event arguments. The memory tensor models the inter-dependency between events, enabling rough reading to discriminate missing events to extracted events, and empowering elaborate reading with argument-level contexts.

2.1. Basic Encoding

Given a document D with N sentences, basic encoding involves three steps to produce contextual representations for characters, sentences and the document. **First**, a Sentence-Encoder is respectively adopted to each sentence S_j, deriving character representations in each sentence as $\mathbf{S}_j = [\mathbf{c}_{j,1}, \mathbf{c}_{j,2}, ..., \mathbf{c}_{j,n}]$, where $\mathbf{c}_{j,k} \in \mathbb{R}^d$, n is the number of characters in S_j and d is the dimension of character representation. **Next**, max-pooling is applied over each sentence $\mathbf{S}_j$ to get raw sentence representations $\mathbf{s}_j^r$, and then, a Document-Encoder is utilized over $[\mathbf{s}_1^r, \mathbf{s}_2^r, ..., \mathbf{s}_N^r]$, obtaining document-aware sentence representations $\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_N]$ with $\mathbf{s}_j \in \mathbb{R}^d$ and $\mathbf{s} \in \mathbb{R}^{N \times d}$. **Finally**, max-pooling is applied over $\mathbf{s}$, generating the document representation $\mathbf{D} \in \mathbb{R}^d$.

2.2. Rough Reading

Rough reading works to detect event occurrence. As Algirithm 1 shows that rough reading also serves to check missing events, it needs the memory to avoid redundant detection with previous extracted events. Specifically, we utilize a Memory-Encoder over the memory tensor $\mathbf{m}_e$ to enable information flow between events and refine prior e-type events as follows:

$$\hat{\mathbf{m}}_e = \texttt{SumPooling}(\texttt{MemEnc}(\mathbf{m}_e)), \tag{1}$$

where $\hat{\mathbf{m}}_e \in \mathbb{R}^d$ is the summarized memory. When rough reading is first applied for an e-type event, $\mathbf{m}_e \in \mathbb{R}^{(1) \times d}$ only contains the randomly initialized event-type embedding $\mathbf{e}_e$; When rough reading is applied to check missing events, $\mathbf{m}_e \in \mathbb{R}^{(1+l_m) \times d}$ contains $\mathbf{e}_e$ and l_m extracted event argument representations.

We remove the information about prior events from the document, and compute the probability p_e to the occurrence of an unextracted e-type event in the document as follows:

$$\begin{aligned} \hat{\mathbf{D}} &= \mathbf{D} - \hat{\mathbf{m}}_e, \\ p_e &= \texttt{sigmoid}(\mathbf{W}_s(\texttt{tanh}(\mathbf{W}_d\hat{\mathbf{D}} + \mathbf{W}_t\mathbf{e}_e))), \end{aligned} \tag{2}$$

where $\mathbf{D} \in \mathbb{R}^d$ and $\hat{\mathbf{D}} \in \mathbb{R}^d$ are respectively the original and redundancy-aware document representation, $\mathbf{W}_s, \mathbf{W}_d$, $\mathbf{W}_t$ are trainable weights. If p_e is greater than the predefined threshold, HRE perceives one e-type unextracted event and elaborate reading is subsequently exploited to extract arguments, otherwise, HRE moves to tackle the next event type.

During training, we use binary cross-entropy loss towards p_e to teach rough reading to detect event occurrence. Since rough reading is employed multiple times in one document, we sum all such losses from each rough reading as L_{rr}.

Algorithm 1: Inference process of HRE

Input: Document D;
Output: Extracted event records;
Encode D for the representations of the document, sentence and character as $\mathbf{D} \in \mathbb{R}^d$, $\mathbf{s}_j \in \mathbb{R}^d$ and $\mathbf{c}_{j,k} \in \mathbb{R}^d$;
for *each event type* e **do**
  Initialize the memory tensor me using a randomly initialized e-type-specific embedding $\mathbf{e}_e$;
  // RoughReading. ;
  Refine $\mathbf{m}_e$ as $\hat{\mathbf{m}}_e$ as Eq.1 ;
  Compute p_e with $\hat{\mathbf{m}}_e$ and $\mathbf{D}$ as Eq.2 ;
  while $p_e >$ *threshold* **do**
    for *each event role* r_e^i **do**
      // Elaborate Reading. ;
      Construct the query $\bar{\mathbf{r}}_e^i$ as Eq.3 ;
      Locate the j_{th} sentence with $\bar{\mathbf{r}}_e^i$ as Eq.4-5 ;
      Extract argument $\mathbf{arg}_{r_e^i}$ from $\mathbf{S}_j$ as Eq.6-8. ;
      Update $\mathbf{m}_e$ with $\mathbf{arg}_{r_e^i}$ and $\mathbf{s}_j$ as Eq.9;
    end
    // RoughReading to check missing events. ;
    Refine $\mathbf{m}_e$ as $\hat{\mathbf{m}}_e$ as Eq.1 ;
    Compute p_e with $\hat{\mathbf{m}}_e$ and $\mathbf{D}$ as Eq.2;
  end
end

2.3. Elaborate Reading

After HRE detects the occurrence of one e-type event, elaborate reading works to one-by-one extract concrete event arguments following a predefined event role order [12]. For each event role, a query, which refines the current event role and inter-dependency between previous extracted arguments, is constructed to clarify the reading target. Specifically, we utilize a Memory-Encoder to inject prior argument contexts into the role embedding as follows:

$$[\bar{\mathbf{r}}_e^i \,;\, \bar{\mathbf{m}}_e] = \texttt{MemEnc}([\mathbf{r}_e^i \,;\, \mathbf{m}_e]) \tag{3}$$

where "$[\cdot \,;\, \cdot]$" means the operation of concatenation, $\mathbf{r}_e^i \in \mathbb{R}^{1 \times d}$ is the trainable role-specific embedding for i_{th} role of e-type event, $\mathbf{m}_e \in \mathbb{R}^{(1+l_m) \times d}$ is the raw memory tensor, and $\texttt{MemEnc}$ is the same encoder used in Eq.1. We leverage $\bar{\mathbf{r}}_e^i \in \mathbb{R}^{1 \times d}$ as the query to extract argument of current event role.

Sentence Location Module locates the sentence where the target argument resides. In one sentence, arguments sharing the same event role are semantically similar to each other to some extent, thus we first filter information about prior extracted arguments as follows:

$$\begin{aligned} \mathbf{g} &= \texttt{sigmoid}(\mathbf{W}_l([\hat{\mathbf{m}}_e \,;\, \mathbf{s}_j])), \\ \hat{\mathbf{s}}_j &= \mathbf{s}_j - \mathbf{s}_j * \mathbf{g}. \end{aligned} \tag{4}$$

where $\hat{\mathbf{m}}_e \in \mathbb{R}^d$ is the same memory summarization from Eq.1, $\mathbf{s}_j \in \mathbb{R}^d$ is the sentence representation, "$[\cdot \,;\, \cdot]$" is the concatenation

operation producing $[\hat{\mathbf{m}}_e \ ; \ \mathbf{s}_j] \in \mathbb{R}^{2d}$, and $\mathbf{g}$ is a gate controlling information redundancy. Then, HRE chooses the j_{th} sentence as:

$$\begin{aligned} \mathbf{z_s} &= \texttt{AttnScore}(\bar{\mathbf{r}}_e^i, \hat{\mathbf{s}}) = \texttt{softmax}(\frac{\bar{\mathbf{r}}_e^i \hat{\mathbf{s}}^{\mathbf{T}}}{\sqrt{d}}), \\ j &= \texttt{argmax}(\mathbf{z_s}) \end{aligned} \tag{5}$$

where $\hat{\mathbf{s}} \in \mathbb{R}^{N \times d}$ is the redundancy-aware sentence representations of all sentences in the document, $\mathbf{z_s} \in \mathbb{R}^{N \times 1}$ is the relevance score of each sentence computed by the scaled dot-product [18] attention, and the sentence receiving the highest score is selected for the corresponding argument extraction.

In training, we use cross-entropy loss towards $\mathbf{z_s}$ to guide the sentence location, with the gold sentence index as label. In one document, we sum all such losses from each sentence location as L_{sl}.

Argument Extraction Module aims to extract the specific event argument and update the memory tensor. For an event role, assuming that HRE decides to extract the argument from the j_{th} sentence, some preliminary operations are conducted as preparation. First, the query embedding $\bar{\mathbf{r}}_e^i$ is added to each character representation $\mathbf{c}_{j,k}$, enriching the sentence with event-related knowledge. And then, a symbol "**[STOP]**", which is represented as the corresponding role embedding $\mathbf{r}_e^i$, is appended to the sentence to denote the end of extraction. These two operations could be formulated as:

$$\hat{\mathbf{S}}_j = [\mathbf{c}_{j,1} + \bar{\mathbf{r}}_e^i, \mathbf{c}_{j,2} + \bar{\mathbf{r}}_e^i, ..., \mathbf{c}_{j,n} + \bar{\mathbf{r}}_e^i, \mathbf{r}_e^i]. \tag{6}$$

The argument is extracted by a series of character copy operations from $\hat{\mathbf{S}}_j$ as follows:

$$\begin{aligned} &\mathbf{v}_0 = \bar{\mathbf{r}}_e^i, \\ &k = \texttt{argmax}(\texttt{AttnScore}(\mathbf{v}_t, \hat{\mathbf{S}}_j)), \\ &\mathbf{v}_{t+1} = \hat{\mathbf{S}}_j[k], \end{aligned} \tag{7}$$

where k_{th} character in the j_{th} sentence is copied. $\mathbf{v}_0$ is initialized as the query representation $\bar{\mathbf{r}}_e^i$ to locate the first character of target argument, and in each time step t, the character receiving the greatest score is copied and will be used as $\mathbf{v}_{t+1}$. The copy operation will not end until "**[STOP]**" is copied. Supposing that characters $c_{j,k}, c_{j,k+1}$, $c_{j,k+2}$ and "**[STOP]**" are copied, max pooling is applied over the valid tokens to derive the argument representation $\mathbf{arg}_{r_e^i} \in \mathbb{R}^d$ as:

$$\mathbf{arg}_{r_e^i} = \texttt{MaxPooling}([\mathbf{c}_{j,k} \ ; \mathbf{c}_{j,k+1} \ ; \mathbf{c}_{j,k+2}]) \tag{8}$$

In training, we use cross entropy loss towards $\texttt{AttnScore}(\mathbf{v}_t, \hat{\mathbf{S}}_j)$ to guide the argument character copy process, where, in each time step t, the gold argument character index is used as the label. We sum all the character copy losses in each document as L_{ae}.

Memory Update Module appends each extracted argument to the memory tensor $\mathbf{m}_e$, making each reading stage aware of prior extracted arguments. Since the semantics of a single entity may be rare, we fuse the entity and corresponding sentence representation to update the memory as follows:

$$\mathbf{m}_e = [\mathbf{m}_e \ ; (\mathbf{arg}_{r_e^i} + \mathbf{s}_i)], \tag{9}$$

where the updated memory tensor $\mathbf{m}_e \in \mathbb{R}^{(l_m+2) \times d}$ contains $l_m + 1$ arguments and will be used in the next reading stage.

2.4. Training Objective

We sum losses from rough reading, sentence location and argument extraction in elaborate reading as $L_{all} = \lambda_1 L_{rr} + \lambda_2 L_{sl} + \lambda_3 L_{ae}$ and jointly optimize them. λ_1=1.0, λ_2=1.0 and λ_3=0.9 are coefficients to balance different sub-tasks.

3. EXPERIMENTS

3.1. Experiment Setup

Dataset and Metrics. We conduct experiments on the largest DEE dataset by far, which is released by Zheng et al. (2019) [12]. The dataset contains 32,040 documents, where five event types are annotated: *Equity Freeze* (EF), *Equity Repurchase* (ER), *Equity Underweight* (EU), *Equity Overweight* (EO) and *Equity Pledge* (EP). Besides, roughly 6 sentences are involved for one event record, and 29% documents express multiple events. We follow the standard dataset split using 25,632/3,204/3,204 documents as training, dev and test set. We evaluate the model using the official scorer in terms of event-level Precision (P), Recall (R) and F_1-score (F_1).

Implementation Details. We follow Zheng et al. (2019) [12] to set hyper-parameters. We set the dimension of character embedding to 768 and threshold in rough reading for event occurrence to 0.5. The maximum number of sentences and sentence length are 64 and 128. Transformer-Encoder [18] is adopted as the Sentence-Encoder, Document-Encoder and Memory-Encoder.

Baselines. We choose the following baselines. **DCFEE** [8] conducts key-sentence event detection and extracts arguments from the key sentence and its surrounding sentences. DFCEE has two variants, where **DCFEE-O** only extracts one event record while **DCFEE-M** extracts multiple events. **Doc2EDAG** [12] designs an entity-based directed acyclic graph (EDAG), transforming the event record extraction to the entity-based path expending. It has a variant, **GreedyDec**, which greedily produces only one event record. **ArgSpan** [10] extracts scattering arguments by enumerating possible argument spans within a specified scope of sentence window. Since **ArgSpan** is proposed for Argument-Linking task which only seeks to extract arguments across sentences, we form a baseline by replacing elaborate reading with **ArgSpan** to extract arguments.

3.2. Main Results

HRE vs. SOTA. The left part of Table 1 reveals the overall experiment results. We could see that HRE shows noticable superiority over baselines, and we owe the improvement to the better performance on *scattering-arguments* and *multi-events* problem. Note that **ArgSpan** performs worst, we infer the reason as that, comparing with Argument-Linking task, no specific sentence window scope is given to scatting arguments in DEE, thus the invalid spans from the span-enumeration based method overwhelms the model.

HRE's Performance on Scattering-Arguments. To measure HRE's performance in scattering-arguments problem, we first count the average number of sentences in which one event record involves per document, and then respectively divide the documents in the test set into five groups as Fig.2 shows. Comparing HRE with its strongest baseline, Doc2EDAG, we could see that their performances both decrease with the increasing number of sentences where the arguments scatters, but HRE always maintains its advantage. This reflects HRE's remarkable ability to dealing with scattering-arguments. We contribute HRE's such remarkable ability to elaborate reading, which explicitly models the inter-sentence (Eq.5) and intra-sentence (Eq.7) semantics for each argument extraction, empowering HRE's ability to deal with scattering-arguments.

HRE's Performance on Multi-Events. To probe HRE's ability in Multi-Events problem, we divide the test set into a single-event set (**Single.**), where each document contains only one event record, and a multi-events set (**Multi.**). The detailed results on these two sets are listed in the right part of Table 1, and we find that (1) Performances of all models present a decreasing trend from **Single.**

Model	P.	R.	F_1	Single. (F_1)	Multi. (F_1)
DCFEE-O [8]	65.8	53.0	58.0	61.5	49.6
DCFEE-M [12]	57.5	54.5	55.7	58.9	47.7
GreedyDec [12]	77.3	50.4	60.5	74.8	39.4
ArgSpan [10]	27.0	31.4	28.7	29.5	26.9
Doc2EDAG [12]	81.5	68.2	74.5	83.1	63.8
HRE	**81.7**	**72.5**	**76.8**(+2.3)	**87.0**(+3.9)	**64.7**(+0.9)

Table 1. Main results: overall event-level precision (P.), recall (R.) and $\mathbf{F}_1$ on the test set (statistically significant with $p < 0.05$).

Model	P.	R.	F_1	Single.	Multi.
HRE	**80.5**	**71.8**	**75.9**	**85.9**	**66.7**
Rough Reading -Memory	25.9	75.9	38.4	31.5	48.8
Elaborate Reading					
-Redundancy-aware	77.5	70.9	74.0	83.8	64.9
-Query construction	77.8	69.8	73.5	85.1	62.7
-Query enrichment	73.7	70.6	72.3	82.5	62.3

Table 2. Ablation Study on the dev set.

to **Multi.**, which shows the increasing difficulty when *scattering-arguments* meets *multi-events*. (2) DCFEE-O, which lacks of multi-events handling strategy, excels DCFEE-M on **Multi.**. This indicates that unreasonable multi-event tackling strategies may negate performance. (3) Doc2EDAG performs best among baselines but lags behind HRE, which confirms HRE's superior ability to Multi-Events.

3.3. Detailed Analysis

Ablation Study. To probe the contribution of different components, we respectively ablate to the Rough Reading and Elaborate Reading. Table 2 reflects that: (1) Removing the memory exploration in Eq.2 leads to the poorest performance, since HRE always decides there are missing events and detect prior extracted events from the document. (2) The result drops by 1.9% on F_1 when sentence location is conducted on the original sentence representation instead of the redundancy-aware representation. This confirms the necessity of removing redundant information. (3) The query, which refines the inter-dependency between prior events, is indispensable, since the ablation hurts the F_1 by 2.4%. (4) Without adding the query into character representations, the result degradation shows the importance of event-related information in argument extraction.

Computational Cost Analysis. We discuss the computational cost between HRE and Doc2EDAG from two aspects. (1) We count the inference speed, which refers to the number of documents that the model could handle per second during model inference, as the time computational cost. Specifically, HRE works with an inference speed of 5.9Docs/s while that of Doc2EDAG is 7.2Docs/s. (2) We utilize the amount of model parameters to represent the space computational cost. Specifically, the parameter amount of HRE is 75.0M while that of Doc2EDAG is 66.8M. Though HRE is slightly more costly than Doc2EDAG, we think that HRE deserves the extra cost, since it achieves 2.3% improvements over Doc2EDAG on the overall F_1 and shows great performance on the two challenges of DEE.

3.4. Case Study

In Fig. 3, we use a case to compare HRE with Doc2EDAG. Specifically, the selected document contains two event records of *Equity*

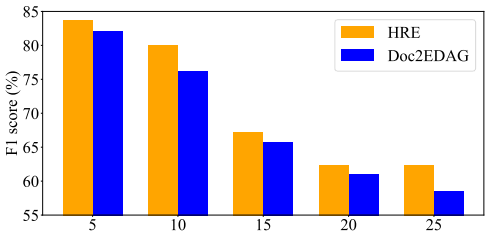

Fig. 2. Performance on documents with arguments scarring in different number of sentences.

..... **[S5.]** On Sep.10th, 2018, 248474132 shares of Hongtu Co.,ltd held by Sanbao Co.,ltd were frozen by Nanjing Court. ..... **[S7.]** On Sep.10th,2018, 66900000 shares of Hongtu Co.,ltd, which were held by Sanbao Co.,ltd were frozen by Beijing Court. **[S8.]** The above two freezing periods are three years, which is calculated from the date of formal freezing. ..... **[S10.]** As the disclosure date of this announcement, Sanbao Co.,ltd holds 253212320 shares of the company, accounting for 21.46% of the total share capital of the company, and the number of shares in pledge is 245.4 million, 98.76% of the total held shares. .....

Event Records						
Model	Equity Holder	Frozen Shares	Legalinstitution	Holding Shares	Holding Ratio	...
Doc2EDAG	Sanbao Co.,ltd	248474132shares	Nanjing Court	253212320shares	21.46%	...
	Sanbao Co.,ltd	66900000shares	Beijing Court	245.4 million	98.76%	...
HRE	Sanbao Co.,ltd	248474132shares	Nanjing Court	253212320shares	21.46%	...
	Sanbao Co.,ltd	66900000shares	Beijing Court	253212320shares	21.46%	...

Fig. 3. Case Study.

Freeze type, where HRE correctly predicts the two event records, while Doc2EDAG wrongly predicts two arguments of the second event record. We contribute the Doc2EDAG's incorrect prediction to two aspects. First, the contexts of candidate argument entities are under-explored in the entity-orient method, which misleads Doc2EDAG to identify event-unrelated entity as event argument. Second, Doc2EDAG insufficiently grasps the interaction between the first extracted event and the second one, thus it makes wrong prediction to the both shared arguments. On the contrary, HRE locates each argument through elaborate reading from sentence to characters, where more fine-grained semantics for discriminating candidate argument entities (eg. 21.46% / 98.76%) could be perceived. Further, the memory mechanism enables HRE to be aware of prior extracted events and arguments, thus HRE performs better.

4. CONCLUSION

In this paper, we propose **HRE** (Human Reading inspired Extractor for Document Events) for DEE task. HRE involves two stages, where **rough reading** detects the occurrence of events and **elaborate reading** extracts concrete event arguments. As far as we know, we take the lead to explore such a reading cognitive process for DEE, and experiments show its effectiveness. In the future, we would like to further adapt HRE into document-level relation extraction task.

5. ACKNOWLEDGEMENTS

This work is supported by the National Key Research and Development Program of China (grant No.2021YFB3100600), the Strategic Priority Research Program of Chinese Academy of Sciences (grant No.XDC02040400) , the Youth Innovation Promotion Association of CAS (Grant No. 2021153) and National Natural Science Foundation of China (Grant No.61902394).

6. REFERENCES

[1] Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao, "Event extraction via dynamic multi-pooling convolutional neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Beijing, China, July 2015, pp. 167–176, Association for Computational Linguistics.

[2] Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao, "Automatically labeled data generation for large scale event extraction," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vancouver, Canada, July 2017, pp. 409–419, Association for Computational Linguistics.

[3] Xiao Liu, Zhunchen Luo, and Heyan Huang, "Jointly multiple events extraction via attention-based graph information aggregation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, Oct.-Nov. 2018, pp. 1247–1256, Association for Computational Linguistics.

[4] Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li, "Exploring pre-trained language models for event extraction and generation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, July 2019, pp. 5284–5294, Association for Computational Linguistics.

[5] Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu, "Event extraction as machine reading comprehension," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, Nov. 2020, pp. 1641–1651, Association for Computational Linguistics.

[6] Xinya Du and Claire Cardie, "Event extraction by answering (almost) natural questions," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, Nov. 2020, pp. 671–683, Association for Computational Linguistics.

[7] Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen, "Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online, Aug. 2021, pp. 2795–2806, Association for Computational Linguistics.

[8] Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao, "DCFEE: A document-level Chinese financial event extraction system based on automatically labeled training data," in *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia, July 2018, pp. 50–55, Association for Computational Linguistics.

[9] Xinya Du and Claire Cardie, "Document-level event role filler extraction using multi-granularity contextualized encoding," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 8010–8020, Association for Computational Linguistics.

[10] Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme, "Multi-sentence argument linking," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 8057–8077, Association for Computational Linguistics.

[11] Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy, "A two-step approach for implicit event argument detection," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 7479–7485, Association for Computational Linguistics.

[12] Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian, "Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, Nov. 2019, pp. 337–346, Association for Computational Linguistics.

[13] L. Luo, Xiang Ao, Y. Song, Feiyang Pan, Min Yang, and Q. He, "Reading like her: Human reading inspired extractive summarization," in *EMNLP/IJCNLP*, 2019.

[14] Zeyang Lei, Yujiu Yang, Min Yang, Wei Zhao, J. Guo, and Y. Liu, "A human-like semantic cognition network for aspect-level sentiment classification," in *AAAI*, 2019.

[15] P. Avery and Michael F. Graves, "Scaffolding young learners' reading of social studies texts.," *Social studies and the young learner*, vol. 9, pp. 10–14, 1997.

[16] A. Sarıçoban, "Reading strategies of successful readers through the three phase approach," *The Reading Matrix : an International Online Journal*, vol. 2, 2002.

[17] Elif Leyla Toprak and Gamze Almacıoğlu, "Three reading phases and their applications in the teaching of english as a foreign language in reading classes with young learners," *Journal of Language and Linguistic Studies*, vol. 5, pp. 20–36, 2009.

[18] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.