applied sciences

Article

Document-Level Event Role Filler Extraction Using Key-Value Memory Network

Hao Wang [1,2,3], Miao Li [1,2,*], Jianyong Duan [2], Li He [1] and Qing Zhang [1]

[1] School of Information Science and Technology, North China University of Technology, Beijing 100144, China
[2] CNONIX National Standard Application and Promotion Lab, Beijing 100144, China
[3] Beijing Urban Governance Research Center, Beijing 102206, China
* Correspondence: limiaoay@163.com

Abstract: Previous work has demonstrated that end-to-end neural sequence models work well for document-level event role filler extraction. However, the end-to-end neural network model suffers from the problem of not being able to utilize global information, resulting in incomplete extraction of document-level event arguments. This is because the inputs to BiLSTM are all single-word vectors with no input of contextual information. This phenomenon is particularly pronounced at the document level. To address this problem, we propose key-value memory networks to enhance document-level contextual information, and the overall model is represented at two levels: the sentence-level and document-level. At the sentence-level, we use BiLSTM to obtain key sentence information. At the document-level, we use a key-value memory network to enhance document-level representations by recording information about those words in articles that are sensitive to contextual similarity. We fuse two levels of contextual information by means of a fusion formula. We perform various experimental validations on the MUC-4 dataset, and the results show that the model using key-value memory networks works better than the other models.

Keywords: event extraction; document-level; key-value memory network

check for updates

Citation: Wang, H.; Li, M.; Duan, J.; He, L.; Zhang, Q. Document-Level Event Role Filler Extraction Using Key-Value Memory Network. *Appl. Sci.* **2023**, *13*, 2724. https://doi.org/10.3390/app13042724

Academic Editor: Rafael Valencia-Garcia

Received: 28 December 2022
Revised: 13 February 2023
Accepted: 16 February 2023
Published: 20 February 2023

1. Introduction

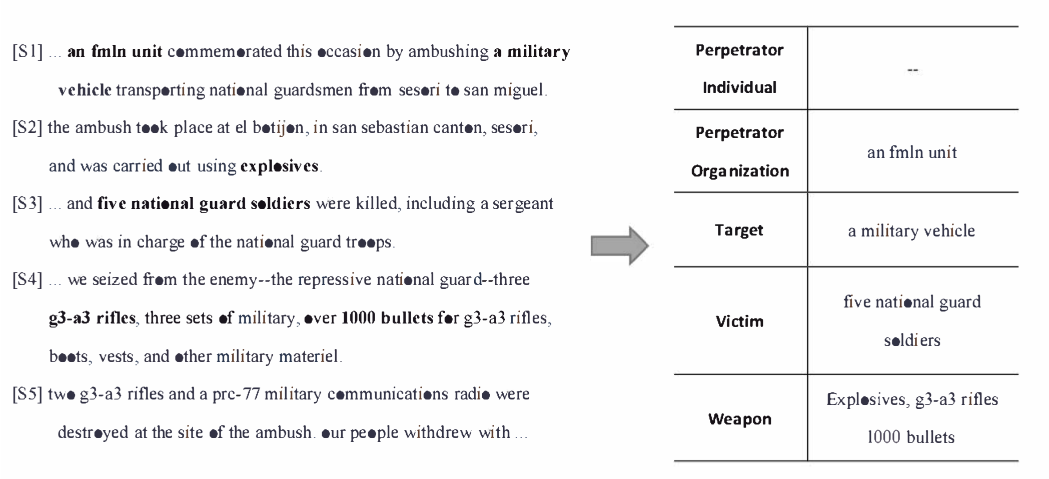
In current practical use, it is unrealistic to extract complete arguments from a sentence alone. Therefore, the main task now is to improve the accuracy of document-level event element extraction. A complete document-level event extraction typically includes event detection, role-filler extraction, and event tracking. Our main task is to study the document-level extraction of event arguments. In Figure 1, the left side shows multiple sentences from an article, and some arguments are extracted from the left side of the article.

For example, in an article containing an attack event, this attack event will have some event actors (e.g., the individual attacker, attacker organization, attack target, victim, weapon, etc.). Our main goal is to identify the textual scope of each event metric in the text, and thus an accurate capture of multi-sentence contextual information is required. In this example, the victim is mentioned implicitly in the first sentence, however, is only explicitly referred to in the third sentence. This event involves a total of three sentences, which shows that solving the problem of crossing sentence boundaries and performing document-level event extraction is important to facilitate downstream tasks.

Back in 2009, document-level event arguments were approached as a template-generation problem, and a new model was proposed. The model first identifies the type of event in a sentence. Then, the corresponding event roles are identified based on the event types. This model is called the pipeline model [1,2]. However, all models have certain problems: (1) they require the help of many external features, such as some syntactic features, which are more useful as an aid to document-level event extraction, and lexical features, which facilitate the identification of individual event arguments; and (2) pipeline models suffer from error propagation.

Previous studies have shown that end-to-end neural sequence models perform well in tasks, such as NER [3] and sentence-level event extraction on the ACE dataset [4]. They are more suitable for downstream tasks involving individual sentences.

The neural network model works well for sentence-level event extraction. However, it still faces many difficulties in document-level event extraction. The arguments of an event usually run through the whole text, and developing a method to establish long-term dependencies across multiple sentences [5] is one of the greatest difficulties encountered thus far. The RNN back-propagation algorithm [4] is typically used to model long sentences. In 2020, Du et al. [6] applied the end-to-end neural sequence model as a new framework for document-level event extraction, which addressed the impact of long sentences on model performance to an extent.



[S1] … **an fmln unit** commemorated this occasion by ambushing **a military vehicle** transporting national guardsmen from sesori to san miguel.

[S2] the ambush took place at el botijon, in san sebastian canton, sesori, and was carried out using **explosives**.

[S3] … and **five national guard soldiers** were killed, including a sergeant who was in charge of the national guard troops.

[S4] … we seized from the enemy--the repressive national guard--three **g3-a3 rifles**, three sets of military, over **1000 bullets** for g3-a3 rifles, boots, vests, and other military materiel.

[S5] two g3-a3 rifles and a prc-77 military communications radio were destroyed at the site of the ambush. our people withdrew with …

Perpetrator Individual	--
Perpetrator Organization	an fmln unit
Target	a military vehicle
Victim	five national guard soldiers
Weapon	Explosives, g3-a3 rifles 1000 bullets

Figure 1. Document-level event fill argument element-extraction task.

End-to-end neural sequence models usually use BiLSTM to update parameters; however, the inputs of BiLSTM are individual word vectors, which cannot consider the contextual information and, thus, cannot capture the contextual information of the whole chapter. Akbik et al. [7] attempted to solve this problem by dynamically aggregating the embedded contextual information of each word. However, this does not work in cases of individual sentence input. To address the above issues, we propose a hierarchical contextual framework to enhance event theory meta-extraction modeling.

The sentence-level representation mainly uses single sentences, which are input into the BERT-base to decode and obtain sentence-level vectors, and then the individual sentence vectors are passed through BiLSTM to obtain the corresponding hidden layer states. A key-value memory network is used at the document level, using sentence-level word vectors as K-values and hidden values as V. Each k-v slot value is updated within a certain time, and the obtained value and word-level representation are used as the k, v, and q of the attention mechanism to calculate the document-level representation. We conducted a comparison of two evaluation approaches on the MUC-4 dataset and verified the effectiveness of key-value memory networks on document-level event arguments from different perspectives.

The main contributions of this paper can be summarized as follows. First, we propose an effective method for extracting document-level information using key-value memory networks based on an end-to-end neural sequence model. Secondly, evaluation results on MUC-4 show that our model outperforms other end-to-end neural sequence models for the task of document-level event theory element-filling extraction. To a certain extent, it solves the problem that BiLSTM only targets one instance and cannot capture document-level information effectively.

2. Related Work

Sentence-level event extraction typically requires extracting event-trigger words and corresponding event parameters from individual sentences. For example, in the sentence "a man is being mobbed by a group of monkeys", it is first determined that this is an attack event based on an "attack", and then the victim ("man") and the attacker ("monkeys") are extracted, followed by some other arguments (time, place, etc.). For sentence-level event extraction, a number of effective approaches have been explored by several scholars. For example, the more primitive of these feature is an engineering-based framework [8] using semi-supervised or unsupervised approaches.

Using a deep-learning-based model [9] (such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs)), the authors of [10] studied a tree-based convolutional neural network for event detection, and integrated syntax into neural event detection for the first time. Their framework uses the proposed model with GCNs [11] and entity-mention-based pooling. Later, Yang et al. [12] proposed PLMEE, an event-extraction framework based on a pre-trained language model, to overcome the role overlap problem by separating the parameter prediction according to the parametric roles. These methods often focus on extracting the sentence-level context of event triggers and parameters and rarely generalize to document-level event extraction.

Due to the flexibility in natural language expression, the entirety of the theoretical elements that constitute a single event are usually distributed in multiple sentences in the context of a single document, and thus scholars began to study document-level event-extraction techniques. The core difficulty of document-level event extraction is that modeling captures semantic connections across sentences, thus, extracting event arguments distributed in places other than key event sentences. In terms of the research method, document-level event extraction mostly assumes that the number elements that compose the events appear in a key event sentence, and then this method complements the missing theory elements through a manually designed heuristic method on the basis of sentence-level event extraction.

The latest research is dedicated to document-level modeling extraction elements obtained directly through deep-learning methods. Liao Tao proposed a supplementary strategy of missing theory based on the co-occurrence of events. Yang et al. [13] first implemented individual event extraction for each sentence and then used heuristics to complement document-level elements. Subsequently, they found that important events were mentioned repeatedly in the document text, and then conducted global event co-reference inference through the integer linear programming method for document-level event extraction. Although work such as [14] also achieved the joint extraction of document-level events and entities, it made excessive use of feature engineering, which makes the model less easy to use.

In the field of Chinese finance, Zheng et al. [15] proposed a new Doc2EDAG model. This model can efficiently generate entity-based directed acyclic graphs for document-level event extraction. It solves a problem encountered in previous models, which were unable to completely identify the event theory elements and event roles because of fragmented information. Liu and Xu et al. [16] proposed the CIT model, and this model can effectively solve both the problem of dispersing event parameters into multiple sentences and the problem of not exploiting the correlation between events.

During the same year, Yang and Sui et al. [17] proposed a parallel prediction network (DE-PPN), which also solves the problem of dispersion of argument elements and multiple events in document-level event extraction to an extent. They introduced a new matching loss function to train the model to achieve an optimized global effect.

Our model merges sentence-level information with document-level information, and instead of using manually designed feature information, the model learns it automatically and is able to dynamically merge the information at both levels. Moreover, in contrast to previous pipeline-based work, we train the model on the end-to-end sequence labeling problem as a task in a supervised manner, and the event patterns are predefined. In document-level information, we use a key-value memory network to remember the contextual representation of each word vector, and an attention mechanism to compute the document-level representation of each slot value. The fused document-level and sentence-level information is then fed into the CRF decoder to obtain the final result.

3. Methods

Figure 2 shows the diagram of our overall model, where the encoder is divided into two parts (a document-level encoder on the left and sentence-level encoder on the right), which are merged and fed into the CRF decoder. We divide a paragraph into multiple sentence inputs, and each sentence goes through BERT and then BiLSTM to obtain sentence-level representations. At the sentence level, each word is embedded as w_i, the key-value part of the key-value memory network, and it then goes through the BiLSTM encoder to output its corresponding hidden layer state h_i. h_i serves as the value part of the key-value memory network.

The document-level part will update the slot values of k-v periodically for each iteration. The word embedding Wq_i is used as the q of the attention mechanism, the key as the k of the attention mechanism, and the value as the v. Then, the document-level information representation is calculated based on the obtained q, k, and v using the attention formula. The sentence-level information and the document-level information are fused and input to the CRF decoding layer to obtain the final score representation.

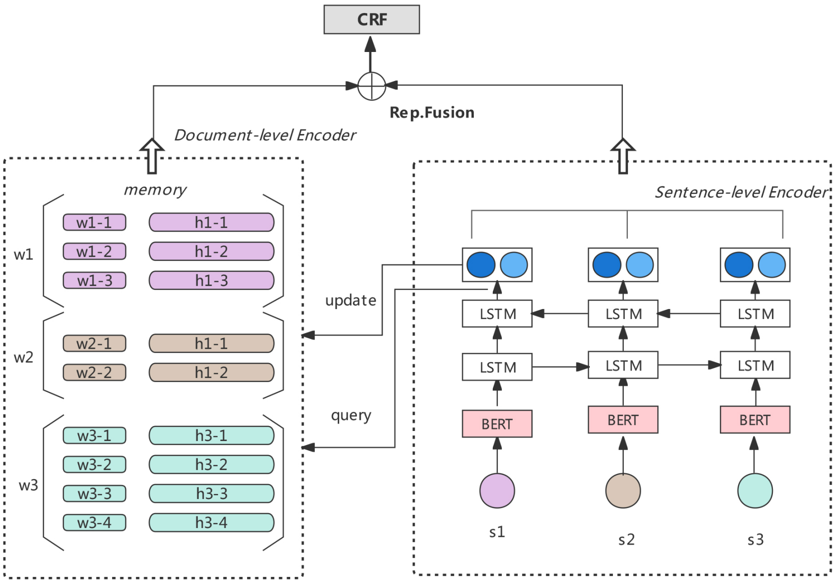


Figure 2. Hierarchical model diagram based on the key-value memory network.

3.1. *Sentence-Level Representation*

Our model uses sentence-level information because it has been shown to be indispensable in information extraction tasks [18,19]. We complemented our focus on inter-sentence information with sentence-level information to improve the accuracy of document-level argumentative meta-recognition. The sentence-level encoder consists of the following aspects.

Embedded layer: The sentence-level encoder studied in this paper does not recognize the boundary of the sentence, so the sentence-level input representation is $\{X_1, X_2, X_3, \dots X_n\}$. Using the input representation X_i as the connection of the word embedding and the context representation, in this paper, we use the 100-dimensional Glove (Pennington et al.) [20] as pre-trained word embedding, and we make the word embedding fixed. With X_i, we obtain the word embedding $xe_i = \mathrm{E}(\mathrm{X_i})$. The obtained word embedding xe_i is then input into BERT.

Pre-trained LM representation: Peters et al. [21] and Devlin [22] proposed a pre-trained language model. The model is advantageous in inter-sentence and paragraph-level text tasks. We use a BERT-base to generate sentence-level contextual information; we have $\{X_1, X_2, X_3, \ldots X_n\}$ and $\{Xb_1, Xb_2, Xb_3, \ldots Xb_n\}$ (see Equation (1)). Thus, the results obtained using BERT can prove the importance of the pre-trained language model.

$$\mathbf{xb}_1, \mathbf{xb}_2, \ldots, \mathbf{xb}_m = \mathrm{BERT}(x_1, x_2, \ldots, x_m) \tag{1}$$

We forward the concatenation of the two representations for each token to the upper layers:

$$\mathbf{x}_i = \mathrm{concat}(\mathbf{x}_{ei}, \mathbf{xb}_i) \tag{2}$$

BiLSTM layer: To help the model better capture features between sequence tokens, we add a bi-directional LSTM encoder layer between the tokens and input the contextual representation obtained from the pre-trained language layer to the BiLSTM layer of the sequence tokens. The specific equation is shown below.

$$\begin{aligned} h_i &= \left[\overrightarrow{h_i}; \overleftarrow{h_i}\right] \\ \overrightarrow{h_i} &= \mathrm{LSTM}\left(x_i, \overrightarrow{h_{i-1}}; \vec{\theta}\right) \\ \overleftarrow{h_i} &= \mathrm{LSTM}\left(x_i, \overleftarrow{h_{i-1}}; \overleftarrow{\theta}\right) \end{aligned} \tag{3}$$

where i indicates the number of tokens and $\vec{\theta}$ and $\overleftarrow{\theta}$ denote trainable parameters.

3.2. Document-Level Representation

Document-level encoders mainly use key-value memory networks. Memory networks were first proposed in 2014 by Weston and Chopra et al. within the field of QA [12]. After which, Miller et al. in 16 further proposed the key-value memory network model. This was later applied to the NER task by Luo et al. [23], significantly improving the overall performance. In order to address the problem wherein BiLSTM cannot make good use of document-level contextual information, this paper also borrows an idea from Luo et al. and uses a key-value memory network to memorize the document-level contextual representation.

If there are m slots, then $(\mathrm{K}_1, \mathrm{V}_1), \ldots, (\mathrm{K}_\mathrm{m}, \mathrm{V}_\mathrm{m})$. In each vector pair, we take the word-vector output from the pre-trained language as the K and the hidden layer state h_i corresponding to each instance of the BiLSTM encoder output as the value. The updated k-v are used as k and v, respectively, in the attention mechanism, and then the document-level contextual representation is calculated according to the formula.

Memory update: Since each word may appear multiple times in the same article, it will appear to occupy multiple slots. This requires a constant update of the key-value: the word embedding updates the corresponding k, and the hidden layer state h_i updates the v. For example, the i-th word X_i may occupy more than one slot. After the calculation, the value of the next slot will be different from the current value, and then this slot value will be rewritten in order for the slot value to be updated when the same word is encountered.

Memory query: For a word i in a sentence, the model extracts the position of the word in all sentences and forms a subset of size T by reverse indexing: $\left(k_{sub_1}, v_{sub_1}\right), \ldots, \left(k_{sub_T}, v_{sub_T}\right)$. Information about the location of each word in the memory network is recorded to the index set. The reverse index set records information about the position of each word in the memory network. Words appear in the article as many times as T.

The key-value memory network calculates the weights of the document-level representations mainly through an attention mechanism. For each word, it uses $k_j \in \left[k_{sub_1}; \ldots; k_{sub_T}\right]$ as attention keys and $v_j \in \left[v_{sub_1}; \ldots; v_{sub_T}\right]$ as attention values. The embedding Wq_i of the query term is then used as the attention query q_i . In this paper, we attempt the following three different compatibility functions to explore their impacts on document-level information $u_{ij} = o\left(q_i, k_j\right)$:

(1) Dot-product attention

$$\mathrm{o}_1\left(q_i, k_j\right) = q_i k_j^T. \tag{4}$$

(2) Scaled dot-product attention [24]:

$$o_2(q_i, k_j) = \frac{q_i k_j^T}{\sqrt{d_w}}. \tag{5}$$

The d_w denotes the dimension of the word embedding and T denotes the number of occurrences of this word in the training set.

(3) Cosine similarity

$$o_3(q_i, k_j) = \frac{q_i k_j^T}{\|q_i\|\|k_j\|}. \tag{6}$$

The document-level representation calculation formula is:

$$\begin{aligned} \alpha_{\mathrm{ij}} &= \frac{\exp(u_{ij})}{\sum_{z=1}^{T}\exp(u_{ij})} \\ r_i &= \sum_{j=1}^{T}\alpha_{ij}v_j \end{aligned} \tag{7}$$

3.3. Fusion Layer

We dynamically merge the obtained sentence-level and document-level representations using fusion formulas and then input the obtained results into the CRF decoder. In total, we tested three different fusion mechanisms:

(1) Simple fusion:

$$\mathrm{g}_i = h_i + \mathrm{r}_i. \tag{8}$$

(2) Condition fusion:

$$g_i = \lambda h_i + (1-\lambda) r_i. \tag{9}$$

(3) Gated fusion:

$$\begin{aligned} \mathrm{p}_i &= \mathrm{sigmoid}(W_1 h_i + W_2 r_i + b) \\ \mathrm{g_i} &= p_i h_i + (1-p_i) r_i. \end{aligned} \tag{10}$$

where W_1 and W_2 are the weight values. When the value of the superparameter is 1, it means that the document-level information is fully used, and when the value is 0, the sentence-level information is fully used. s_i denotes the sentence-level information, and r_i denotes the document-level information.

3.4. CRF Layer

As with other end-to-end sequence models, we adopt CRF as the decoding part of the whole model. It will consider the marker information of the previous data when marking the data. Joint modeling can improve the performance of the model compared with individual label modeling.

We evaluate the accuracy of argumentative element recognition based on the maximum score sequence. The document-level and sentence-level information is fused to obtain $\{\mathrm{G}_1, \mathrm{G}_2, \ldots, \mathrm{G}_\mathrm{m}\}$, then the set—after passing a linear layer—can obtain a label space G of size m. After the CRF decoding layer, we obtain the labeling sequence $\mathrm{Y} = \{\mathrm{y}_1, \mathrm{y}_2, \ldots, \mathrm{y}_\mathrm{m}\}$, and the output sequence of the maximum score can be derived according to Equation (11) as follows:

$$\mathrm{S}(x, y) = \sum_{i=0}^{m} B_{y_i, y_{i+1}} + \sum_{i=1}^{m} G_{i, y_i} \tag{11}$$

where B is the score transfer matrix, and $B_{i,j}$ represents the transfer score transferred from label i to label j. During the decoding process, the present model is able to predict the output sequence that obtains the maximum score.

4. Experiments

We focus on the evaluation of the model's performance on the MUC-4 dataset and compare the results with previous work. We compare two models—plain cross-sentence level and a model using key-value memory networks—in terms of how well they work for document-level argument extraction.

4.1. Datasets

The MUC-4 dataset is composed of 1700 documents and associated answer key (role-filler) templates. To ensure that the results of the model in this paper can be compared with the results previously reported for this dataset, 1300 documents are used as the training set, 200 documents as the validation set, and 200 documents as the test set.

We used the entity-level assessment metric of Du et al. from 2021 using the precision (P), recall (R), and F1 mean to represent the macro mean of all event roles. In addition, two evaluation methods (head noun phrase match and exact match) are used to evaluate the accuracy of the model in capturing the theoretical element role-filling boundary separately. Table 1 shows the results of the previous models and this paper's models on MUC-4 for both evaluation methods. Table 2 shows the F1 scores refined to each event role (individual perpetrator, perpetrator organization, physical target, victim, and weapon) and also compares with the previous model to demonstrate more clearly how the model works for each event role.

Table 1. Macro averaging results for a document-level event extraction task.

Models	**Head Noun Match**			**Exact Match**		
	Prec.	Recall	F-1	Prec.	Recall	F-1
GLACTER	47.80	57.20	52.08	-	-	-
TIER	50.80	61.40	55.60	-	-	-
Cohesion Extract	57.80	59.40	58.59	-	-	-
Multi-Granularity Reader	56.44	62.77	59.44	52.03	56.81	54.32
KDR	56.20	**64.83**	**60.20**	52.91	57.63	**55.16**

Table 2. The results for each event role based on the head-noun matching metric.

Models	**PerpInd**			**PerpOrg**			**Target**			**Victim**			**Weapon**		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
GLACTER	51	58	54	34	45	38	42	72	53	55	58	56	57	53	55
TIER	52	58	55	55	48	51	55	61	58	63	59	61	62	64	63
Cohesion Extract	54	57	56	54	49	51	55	68	61	63	59	61	62	64	63
Multi-Granularity Reader	53	52	52	51	68	58	60	64	62	49	62	55	68	67	68
KDR	54	51	52	56	65	**60**	58	61	59	47	53	50	69	66	**68**

4.2. Experimental Settings

We compared all other models on the same task and dataset. (1) GLACTER [1] is composed of a sentence classifier and a set of per-event theoretical elements initially populated with templates, and its results are obtained from the probability product of normalized sentences and phrases.

(2) TIER [2] proposes a multi-stage approach. The processing is divided into three stages: the classification of event documents, event sentence recognition, and noun phrase analysis.

(3) Cohesion Extract [25] uses a bottom-up approach to first identify candidate-fillable theorems in a document and then refine the set of candidate theorems using a cohesive sentence classifier.

(4) Multi-Granularity Reader [6] is a multi-granularity model architecture that explores the effects of different sentence lengths on the model results and demonstrates that the model results grow and then decline with sentence length. We also use the end-to-end neural sequence model key-value memory network as the model for the document-level part—with the word embedding in the sentence level as the key value k and the corresponding hidden layer state h_i as the value—and then calculate the document-level contextual information through the attention mechanism, which incorporates the sentence-level information to a greater extent. The last row of Table 1 shows a significant improvement in the overall results.

We use two evaluation results with the macro average results in Table 1. For a detailed understanding of how the model extracts filling results for each event argument, the results for each event role are presented in Table 2. From these two tables, it can be seen that the multi-level end-to-end sequence model can achieve almost the same results as the pipeline model. In the results for the document-level model, it is clear that the effects of the key-value memory network are better than the results of splicing after direct multi-sentence input.

We used two evaluation approaches, and the macroscopic average results are shown in Table 1. From both evaluation approaches, on the MUC-4 dataset and with the same end-to-end neural sequence model, the model with the key-value memory network was able to achieve nearly 61 percent effectiveness—significantly higher than the multi-granularity reader model for the same task. The overall results are higher than other pipeline models by two percentage points, which is sufficient to prove the superiority of this paper's model and demonstrate that, to a certain extent, the key-value memory network solves the problem wherein BiLSTM only targets one instance and cannot effectively capture document-level information.

To understand in detail how the model extracts the populated results for each event argument element, Table 2 reflects the results for each event role. As can be seen from this table, the end-to-end sequence model with both sentence-level and document-level layers achieves almost the same or better results compared with the pipeline model. The results of the model with the key-value memory network are better than the results from the same task model over several years for the two roles of "attack organization" and "weapon".

4.3. Ablation Experiments

To further explore the degree of influence of each module of the model on the experimental results, we explored the influence of three different gating fusion mechanisms on the overall model. Table 3 shows that the use of the sigmoid function and setting superparameters are almost comparable for the experimental results; however, the effects of simple summation are not as good as those of the remaining two ways.

In the gated fusion mechanism, the effect is not as good when the sentence level and the document level account for exactly the same proportion, as it works better when the document level accounts for a larger proportion. This indicates that the document-level contextual information is more important; however, the sentence-level information cannot be missing either, and the overall model effect tends to decrease when there is no sentence-level information at all.

In order to eliminate the impact of other modules on the overall model performance, we demonstrate the effectiveness of key-value memory networks for document-level event element extraction. We performed relevant ablation experiments, as shown in Table 4, where the input length is optimal and sentence-level inputs are not considered.

A normal end-to-end neural sequence model achieved an accuracy of about 55 for document-level event element extraction. With the use of key-value memory networks, the results reached 59, which is nearly 4 percentage points higher and is sufficient to demonstrate the effectiveness of key-value memory networks.

Table 3. The effects of different fusion mechanisms on the experimental results.

	Head Noun Match			**Exact Match**		
	P	R	F1	P	R	F1
Simple Fusion	51.82	54.63	53.18	42.38	52.03	46.71
Condition Fusion	56.20	64.83	60.20	52.91	57.63	55.16
Gated Fusion	56.73	60.89	58.74	53.27	57.95	55.51

Table 4. Ablation study on modules' influence on the KDR.

	Head Noun Match			**Exact Match**		
	P	R	F1	P	R	F1
w/ key-value memory	57.35	59.80	58.55	54.82	53.27	54.03
w/o key-value memory	56.78	52.64	54.64	53.36	49.65	51.44

In this paper, we also explore three compatibility functions in studying the impact of key-value memory networks on the overall model. With all other conditions exactly the same, only the control compatibility function is used as a variable. From the experimental results in Table 5, it can be seen that the cosine similarity works best in both evaluation methods, whereas the dot product attention function is the least effective. The cosine similarity calculates the degree of association between the word embedding and the query word, and the closer the value is to 1, the higher the degree of association. It is more favorable to calculate the similarity of words in documents. Therefore, finally, the cosine similarity is used as the compatibility function in this paper.

Table 5. Exploring the effects of different compatibility functions on the overall performance of the model.

Compatibility Functions	**F1**	
	Head Noun Match	**Exact Match**
dot-product attention	59.67	53.87
scaled dot-product attention	59.96	55.05
cosine similarity	60.20	55.16

4.4. Further Analysis

We use the following two cases to further demonstrate that key-value memory networks can compensate for the problem of document-level information loss caused by ordinary end-to-end neural sequence models that target only a single input instance. To ensure overall fairness, cases were randomly selected and were performed with inputs of equal length and under the same external conditions. In the first case, the yellow part is one of the victims of this event; however, the ordinary document-level event argument extraction model does not identify it. However, our model does not ignore this argument, which is in a moderately low position in the whole article and needs to be combined with certain contextual information as our model works well in these conditions.

...meanwhile, troops of the civiplan battalion killed an fmln guerrilla in a skirmish on the slopes of la cruz hill in chalatenango department.....

The second example is found at the end of a long article, where the green part of the "attacker organization" is only implicitly mentioned at the beginning of the article, which requires the model to be able to remember contextual information over a large span of time. Again, this attacker organization is identified by the model in this paper, which shows that the key-value memory network is more advantageous than other models when the context span is large.

> ...night, while residents slept or watched television, 30 guerrillas dressed in military uniforms set the shacks on fire, ...the announcer says the attack is thought to have been carried out by an army of national liberation group .

5. Conclusions and Future Work

In this paper, we explored the effectiveness of bond-valued memory networks for document-level event-fill extraction based on an end-to-end sequence model. For the problem of information loss, this paper proposed a new hierarchical model based on a key-value memory network. The analysis of the results obtained on the MUC-4 dataset show that the present model achieved substantial improvements compared with the previous work. In the future, we intend to further explore how to improve the recognition degree of the more obscure theoretical elements of the model without the help of external knowledge, and this will be the focus of our next work.

Author Contributions: Conceptualization, H.W., J.D., L.H. and Q.Z.; writing—original draft preparation, H.W. and M.L.; writing—review and editing, H.W.; data curation, M.L.; validation, J.D., L.H. and Q.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the RD Program of the Beijing Municipal Education Commission (KM202210009002), the National Natural Science Foundation of China (61972003),and the Beijing Urban Governance Research Center Fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We would like to thank the anonymous reviewers for their helpful comments. We would like to thank the referees for their comments, which helped improve this paper considerably.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Patwardhan, S.; Riloff, E. A unified model of phrasal and sentential evidence for information extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore, 6–7 August 2009; pp. 151–160.
2. Huang, R.; Riloff, E. Peeling back the layers: Detecting event role fillers in secondary contexts. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June, 2011; pp. 1137–1147.
3. Chiu, J.P.; Nichols, E. Named entity recognition with bidirectional LSTM-CNNs. *Trans. Assoc. Comput. Linguist.* **2016**, *4*, 357–370. [CrossRef]
4. Wadden, D.; Wennberg, U.; Luan, Y.; Hajishirzi, H. Entity, relation, and event extraction with contextualized span representations. *arXiv* **2019**, arXiv:1909.03546.
5. Trinh, T.; Dai, A.; Luong, T.; Le, Q. Learning longer-term dependencies in rnns with auxiliary losses. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 4965–4974.
6. Du, X.; Cardie, C. Document-level event role filler extraction using multi-granularity contextualized encoding. *arXiv* **2020**, arXiv:2005.06579.
7. Akbik, A.; Bergmann, T.; Vollgraf, R. Pooled contextualized embeddings for named entity recognition. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; pp. 724–728.
8. Surdeanu, M.; Harabagiu, S.; Williams, J.; Aarseth, P. Using predicate-argument structures for information extraction. In Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan, 7–12 July 2003; pp. 8–15.
9. Chen, Y.; Xu, L.; Liu, K.; Zeng, D.; Zhao, J. Event extraction via dynamic multi-pooling convolutional neural networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the seventh International Joint Conference on Natural Language Processing, Beijing, China, 26–31 July 2015; pp. 167–176.
10. Nguyen, T.; Grishman, R. Graph convolutional networks with argument-aware pooling for event detection. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.

11. Yao, L.; Mao, C.; Luo, Y. Graph convolutional networks for text classification. In Proceedings of the AAAI conference on artificial intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 7370–7377.
12. Yang, S.; Feng, D.; Qiao, L.; Kan, Z.; Li, D. 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, 28 July–2 August 2019; pp. 5284–5294.
13. Yang, H.; Chen, Y.; Liu, K.; Xiao, Y.; Zhao, J. Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In Proceedings of the ACL 2018, System Demonstrations, Melbourne, Australia, 15–20 July, 2018; pp. 50–55.
14. Yang, B.; Mitchell, T. Joint extraction of events and entities within a document context. *arXiv* **2016**, arXiv:1609.03632.
15. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. *arXiv* **2019**, arXiv:1904.07535.
16. Document-level event extraction via heterogeneous graph-based interaction model with a tracker. *arXiv* **2021**, arXiv:2105.14924.
17. Document-level event extraction via parallel prediction networks In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Virtual Conference, 1–6 August 2021; pp. 6298–6308.
18. Zhang, Y.; Liu, Q.; Song, L. Sentence-state lstm for text representation. *arXiv* **2018**, arXiv:1805.02474.
19. Zhou, J.; Zhao, H. Head-driven phrase structure grammar parsing on Penn treebank. *arXiv* **2019**, arXiv:1907.02684.
20. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
21. Peters, M.E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. *Deep Contextualized Word Representations*; Association for Computational Linguistics: New Orleans, LA, USA, 2018.
22. Devlin, J.; Chang, M. W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the NAACL-HLT, Minneapolis, Minnesota, 2–7 June 2019; pp. 4171–4186.
23. Luo, Y.; Xiao, F.; Zhao, H. Hierarchical contextualized representation for named entity recognition. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, New York, USA, 7–12 February 2020, Volume 34; pp. 8441–8448.
24. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2017; Volume 30.
25. Huang, R.; Riloff, E. Modeling textual cohesion for event extraction. In Proceedings of the AAAI Conference on Artificial Intelligence, Toronto, ON, Canada, 22–26 July 2012; Volume 26, pp. 1664–1670.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.