A Hierarchical Neural Network for Information Extraction of Product Attribute and Condition Sentences

Yukinori Homma, Kugatsu Sadamitsu, Kyosuke Nishida,
Ryuichiro Higashinaka, Hisako Asano and Yoshihiro Matsuo
NTT Media Intelligence Laboratories, NTT Corporation
1-1 Hikari-no-oka, Yukosuka, 239-0847, Japan
{`homma.yukinori, sadamitsu.kugatsu, nishida.kyosuke,`
`higashinaka.ryuichiro, asano.hisako, matsuo.yoshihiro`}
`@lab.ntt.co.jp`

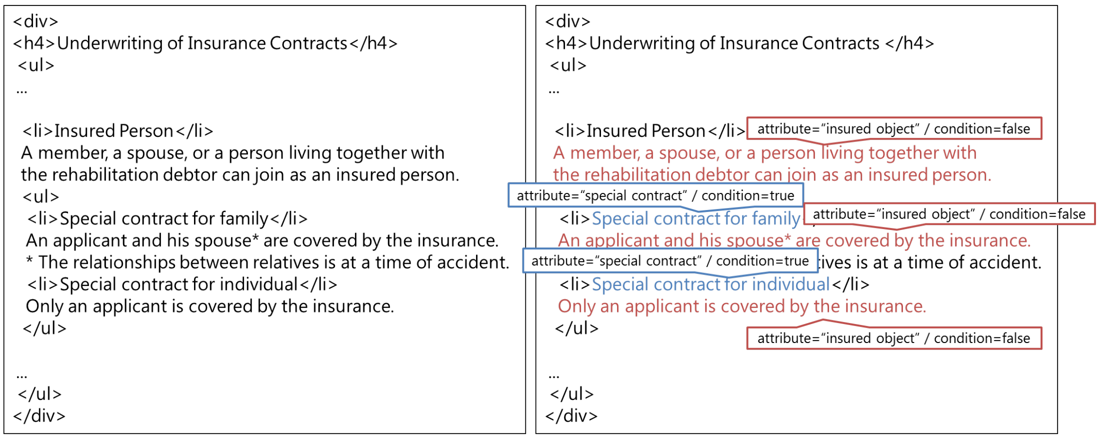
Abstract

This paper describes a hierarchical neural network we propose for sentence classification to extract product information from product documents. The network classifies each sentence in a document into attribute and condition classes on the basis of word sequences and sentence sequences in the document. Experimental results showed the method using the proposed network significantly outperformed baseline methods by taking semantic representation of word and sentence sequential data into account. We also evaluated the network with two different product domains (insurance and tourism domains) and found that it was effective for both the domains.

1 Introduction

With the increase in the number of product documents in electronic form, it is becoming increasingly important to build technologies to extract information from these documents. In particular, it is useful to extract information about product attributes (such as *"Insurance Premiums"*) and their values (such as *"$ 0.50 per day"*) from web product documents for many applications such as commodity comparison, product recommendation and question answering systems about products. For instance, to provide a question answering system that compares particular attributes of products, we need to extract the values of common attributes from each product document.

In this study, we tackled the following two problems for extracting information from product documents on the Web. The first and main problem is to classify each sentence into attribute classes and the second one is to distinguish whether or not each sentence includes condition information, which is helpful in subdividing the attribute. Figure 1 shows an example insurance product document and an example of classification results of attribute and condition.



```
<div>
<h4>Underwriting of Insurance Contracts</h4>
 <ul>
 ...

  <li>Insured Person</li>
  A member, a spouse, or a person living together with
  the rehabilitation debtor can join as an insured person.
  <ul>
   <li>Special contract for family</li>
   An applicant and his spouse* are covered by the insurance.
   * The relationships between relatives is at a time of accident.
   <li>Special contract for individual</li>
   Only an applicant is covered by the insurance.
  </ul>

 ...
 </ul>
</div>
```

attribute="insured object" / condition=false
attribute="special contract" / condition=true
attribute="insured object" / condition=false
attribute="special contract" / condition=true
attribute="insured object" / condition=false

Figure 1: Left: Example of insurance product document. Right: Example of labeled attribute and condition classes.

This work is licensed under a Creative Commons Attribution 4.0 International Licence. Licence details: `http://creativecommons.org/licenses/by/4.0/`

Attribute sentence classification Product documents in the Web describe product attributes in different ways, e.g. tables, listings and plain sentences. Although most previous studies have tried to extract *words* or *phrases* presenting product attribute values such as *"size"* or *"fee"*, from the above components, we think that it is also useful to extract *sentences* presenting values such as *"overview"* or *"payment terms"* to provide a question answering system that will explain products in the form of sentences. We therefore tackled the following problem in this study.

Problem 1. Given an HTML/XML document, we classify each sentence into pre-defined attribute classes, where a sentence consists of a sequence of words and their tag information.

Condition sentence classification There are some cases in which common attribute classes are not enough for obtaining concrete information that can be used in question answering systems, and this problem is not able to be simply solved by subdividing the pre-defined classes since the concrete information is product specific. For example, we can see from Figure 1 that there are two special contracts (for family and for individual) that have different values of the attribute *Insured object*. We consider that it is helpful to extract a sentence that presents necessary conditions in each document instead of constructing a detailed taxonomy of product attributes. We therefore focus on extracting sentences describing *condition* information.

Problem 2. We also classify each sentence into condition or non-condition classes.

Given information about the classification results of attribute and condition sentences, we can provide a question answering system which can provide a proper answer based on the terms of conditions. As an example for Figure 1, when a user asks a question about insured objects of the product, such as *"Who is covered by this insurance?"*, and the system understands that the user made *a special contract for an individual*, the system can provide an answer such as *"Since you made a special contract for an individual, the insurance covers only the applicant"*.

To classify each sentence accurately, it is important to consider the semantic meanings of a sequence of sentences and their HTML tag information. For example, sentences in a listing structure will belong to the same class. A sentence also has a sequence of words, and each word has different importance in forming the semantic meanings of the sentence. Recently, recurrent neural networks (RNNs) have been very successful in capturing semantic representations of word and sentence sequential data in several tasks, including machine translation (Bahdanau et al., 2014), Named Entity Recognition (Joshi et al., 2015; Jagannatha and Yu, 2016) and document classification (Yang et al., 2016).

In the work reported in this paper, we attempted to develop a neural network model to capture semantic representations of word and sentence sequential data and classify each sentence in a document into attributes and condition classes. We developed and here propose a hierarchical neural network that classifies each sentence into attribute and condition classes by learning two classification problems jointly. Experimental results demonstrated that our network performed better than baseline methods by capturing the semantics and structures of sentences. We also evaluated the network in experiments with two different product domains (insurance products, tourism products) and found that it is effective for both the domains.

2 Related work

2.1 Word-level attribute extraction

Many researchers have studied the task of extracting values of attributes in a word or a phrase level from product documents. The work they have done can be classified into two approaches: the pattern matching approach based on structured-tag information (Auer et al., 2007; Muslea et al., 1999; Gulhane et al., 2011) and the machine learning approach based on the predefined attribute-values dictionary (Nagy and Farkas, 2012; Jagannatha and Yu, 2016).

The pattern matching approach relies on a set of extraction based on structured-tag information. DBpedia (Auer et al., 2007) is one of the huge structured datasets using hand-made patterns. It extracts structured information from Wikipedia such as infobox templates. Although the hand-made pattern based approach can extract information with high accuracy from documents that have the same document structure such as infobox templates, it takes considerable costs to make patterns to extract information from documents that have a different document structure. Some researchers have proposed methods to acquire patterns automatically on the basis of machine learning (Muslea et al., 1999; Gulhane et al., 2011). Muslea et al. (1999) proposed a method to extract the node path of the DOM tree as patterns. Gulhane et al. (2011) proposed a method to group similar structured pages in a Web site and automatically change patterns to extract information based on the structure of each document. The method to acquire patterns automatically is effective when there are many documents that have similar structures, such as online shopping site documents. However, it is difficult to extract patterns with high accuracy from web product documents that describe product attributes in different ways.

The machine learning approach has been the one most widely studied during the last decade, and many methods were proposed to extract values of product attributes (Nagy and Farkas, 2012; Jagannatha and Yu, 2016). Nagy and Farkas (2012) proposed a method to extract personal information such as phone number, occupation, and address from search result pages corresponding to personal name queries. Since their method focuses on extraction of a value for each attribute from a document and narrows down a range for finding personal information on the basis of the paragraph title, it is difficult to apply this method for extraction of several values that are scattered in a document. Jagannatha and Yu (2016) proposed a method to extract medical events written by a word or a phrase from unstructured text in electronic health record notes using recurrent neural network frameworks. Their model focuses mainly on word sequence information, which is effective for extracting word or phrase values about attributes in the documents. However, since our aim is to extract sentence-level values of attributes in a web product document, we use HTML tags as features and focus on capturing the importance of words in a sentence to classify the sentence by using the attention architecture.

2.2 Sentence-level attribute extraction

A number of related studies have been performed for extracting sentence values of attributes in several tasks, such as event information extraction (Naughton et al., 2008), extractive summarization (Nishikawa et al., 2015) and emotion classification (Li et al., 2015). Naughton et al. (2008) evaluated the performance of a support vector machine classifier and a language modeling approach for the task of identifying the sentences in a document that describe one or more instances of a specified event type. They use the words of a sentence as features and do not focus on the sentence sequences. Nishikawa et al. (2015) proposed a method for query-oriented extractive summarization to extract information especially from Wikipedia article for a question answering system. This method can extract sentences that present values of product attributes using semi hidden Markov models that capture the semantic meaning of sentence sequences in the document. However, since the method depends on a summarization model and extracts sentences only as a value for each attribute, it cannot extract several values for all attributes and conditions in the documents. In contrast, our method learns to classify each sentence into attribute and condition classes by learning two classification problems jointly and extracts information for attributes and conditions from web documents. These documents are not limited to Wikipedia articles.

Li et al. (2015) proposed a method for sentence-level emotion classification in documents. Their method is based on a factor graph with two layers to model the emotional label dependence in a variable layer and model the sentence context dependence in a factor layer. Their experimental results showed that it is effective for sentence-level emotion classification to use both label and context dependence information. While they did not address the importance of words in a sentence to classify sentences, this paper addresses that topic in reporting on evaluation results. Moreover, our method uses HTML tag information as features to extract structural information of documents and classify each sentence into attribute and condition classes.

3 Proposed network

In our study, we focused on a network that would classify sentences into attribute and condition classes by learning two classification problems jointly.

Assume that a document has L sentences and the i-th sentence contains T_i words and tag information

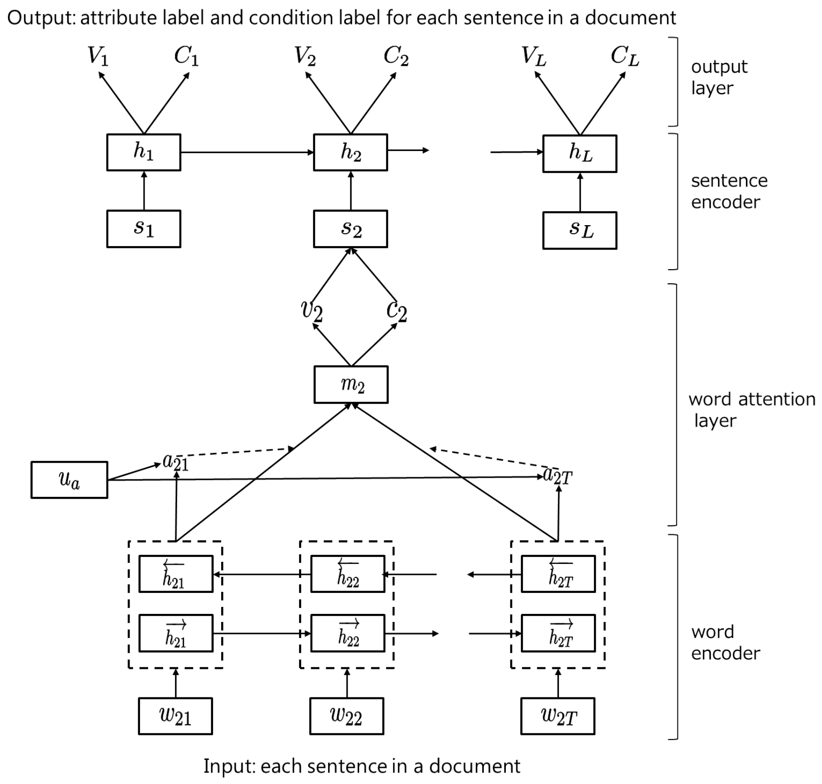


Figure 2: Architecture of proposed network.

t_i. w_{ij} with $j \in [1, T_i]$ represents the word in the i-th sentence. t_i represents the tag name of a parent node of the DOM tree corresponding to the i-th sentence.

3.1 Overview of architecture

The overall architecture of the proposed network is shown in Figure 2. The network contains four components: word encoder, word attention layer, sentence encoder and output layer.

Our network does not learn in an end-to-end manner; we conducted block-wise learning for speeding up the leaning. The learning of the first block of word layers is conducted to obtain a sentence vector, and that of the second block is conducted to classify each sentence into classes. First, it builds a sentence vector that represents the classification probability of attribute classes and condition classes by aggregating important words into hidden sentence vectors m_i in the word encoder (subsection 3.2) and the word attention layer (subsection 3.3) to capture semantic meanings of word sequential data. It then classifies each sentence into pre-defined attribute classes and condition classes (subsection 3.5), taking context sentences into account (subsection 3.4) and using a sentence vector to capture semantic representations of sentence sequential data.

In the following subsections, we will present the details of each component.

3.2 Word encoder

The word encoder builds a word embedding vector for every word in the sentence.

Given a sentence with words w_{ij}, $j \in [1, T_i]$, all words are first encoded into one-hot vectors. A one-hot vector x_{ij} is a binary vector whose elements are all zeros except for the v-th element, which corresponds to the v-th token in a vocabulary V. Then, the one-hot vector x_{ij} is encoded into an E-dimensional vector e_{ij} as the following equation.

$$e_{ij} = W^{wemb} w_{ij} \tag{1}$$

where $W^{wemb} \in \mathcal{R}^{E*|V|}$ is a weight matrix and $|V|$ is the size of the vocabulary.

Next, the network focuses on the context of words in the sentence. Recurrent neural networks (RNNs) with long short-term memory (LSTM) units (Hochreiter and Schmidhuber, 1997) have been successfully applied to a wide range of natural language processing tasks, such as machine translation (Sutskever et al., 2014), language modeling (Zaremba et al., 2014) and so on. However, since standard LSTM networks process sequences in temporal order, they ignore future context. Bi-directional LSTM (BiLSTM) networks introduce a layer where the hidden-to-hidden connections flow in opposite temporal order. These networks are able to exploit information from the past to the future and vice versa.

Our network also has a BiLSTM network architecture. Given an embedded vector sequence $\{e_{i1}, e_{i2}, \cdots, e_{iT}\}$, the network outputs vectors $\{h_{i1}, h_{i2}, \cdots, h_{iT}\}$ as the following equations.

$$\begin{pmatrix} f_j \\ i_j \\ o_j \\ g_j \end{pmatrix} = W^{wh}\tilde{h_{j-1}} + W^{we}e_j + b^{wh} \tag{2}$$

$$c_j = \sigma(f_j) \cdot c_{j-1} + \sigma(i_j) \cdot \tanh(g_j)\tilde{h_j} = \sigma(o_j) \cdot \tanh(c_j) \tag{3}$$

$$\overrightarrow{h_j} = \tilde{h_j}, j \in [1, T] \tag{4}$$

$$\overleftarrow{h_j} = \tilde{h_j}, j \in [T, 1] \tag{5}$$

$$h_j = [\overrightarrow{h_j}, \overleftarrow{h_j}] \tag{6}$$

where $W^{wh} \in \mathcal{R}^{4H*E}, W^{we} \in \mathcal{R}^{4H*H}, b^{wh} \in \mathcal{R}^{4H}$ are weight parameters. The σ and $\tanh$ are respectively a sigmoid function and a hyperbolic tangent function.

3.3 Word attention layer

Recently, attentive neural networks have shown success in several NLP tasks such as machine translation (Bahdanau et al., 2014), image captioning (Xu et al., 2015), speech recognition (Chorowski et al., 2015) and document classification (Yang et al., 2016).

We introduced an attention mechanism (Yang et al., 2016) into the proposed network to extract words that are important to capture the meaning of the sentence. The network outputs the hidden sentence vector m_i in the following equations.

$$u_j = tanh(W^a h_j + b^a) \tag{7}$$

$$a_j = \frac{\exp(u_j^T u_a)}{\sum_j \exp(u_j^T u_a)} \tag{8}$$

$$m_i = \sum_j a_j h_j \tag{9}$$

where $W^a \in \mathcal{R}^{2H*H}, b^a \in \mathcal{R}^{2H}, u_a \in \mathcal{R}^H$ are weight parameters.

The network predicts attribute label v_i and condition label c_i for the sentence, given the hidden sentence vector m_i as input.

$$p(v_i|m_i) = softmax(W^v m_i + b^v) \tag{10}$$

$$p(c_i|m_i) = \sigma(W^c m_i + b^c) \tag{11}$$

where $W^v \in \mathcal{R}^{V*H}, b^v \in \mathcal{R}^H, W^{c_i} \in \mathcal{R}^{V*H}, b^c \in \mathcal{R}^H$ are weight parameters.

The cost function L is the negative log-likelihood as the following equation:

$$L = -\sum_i (\log p(\hat{v_i}|m_i) + \log p(\hat{c_i}|m_i)) \tag{12}$$

where $\hat{v_i}$ is the true attribute label and $\hat{c_i}$ is the true condition label for the i-th sentence.

3.4 Sentence encoder

Given a document, the sentence encoder considers a sentence sequence as input to capture semantic representations of sentence sequential data. All sentences are encoded into sentence vectors $\{s_1, s_2, \cdots, s_L\}$.

The network takes a bag of words vector bw_i, one-hot vector of HTML tag t_i and the output vectors of the word attention layer: $p(v_i|m_i)$ and $p(c_i|m_i)$ for the i-th sentence as input.

$$s_i = W^{semb}[bw_i, t_i, p(v_i|m_i), p(c_i|m_i)] + b^{semb} \tag{13}$$

where $W^{semb} \in \mathcal{R}^{(|V|+|V^t|+|v|+|c|)*H}, b^{semb} \in \mathcal{R}^H$ are weight parameters and $|V^t|$ is the size of the HTML tag vocabulary. $|v|$ and $|c|$ are respectively the number of labels for attributes and conditions.

Then, the network outputs vectors $\{h_1, h_2, \cdots, h_L\}$ as the following equations, given a sentence embedded vector sequence $\{s_1, s_2, \cdots, s_L\}$.

$$\begin{pmatrix} f_l \\ i_l \\ o_l \\ g_l \end{pmatrix} = W^{wh}h_{l-1} + W^{we}e_t + b^{wh} \tag{14}$$

$$c_t = \sigma(f_l) \cdot c_{l-1} + \sigma(i_l) \cdot \tanh(g_l) \tag{15}$$

$$h_l = \sigma(o_l) \cdot \tanh(c_l) \tag{16}$$

where $W^{wh} \in \mathcal{R}^{4H*H}, W^{we} \in \mathcal{R}^{4H*H}, b^{wh} \in \mathcal{R}^{4H}$ are weight parameters.

3.5 Output layer

The output layer predicts the attribute label V_i and the condition label C_i in a way similar to that of the word attention layer.

$$p(V_i|h_i) = softmax(W^V h_i + b^V) \tag{17}$$

$$v_s = argmax_{V_i} p(V_i|h_i) \tag{18}$$

$$p(C_i|h_i) = \sigma(W^C h_i + b^C) \tag{19}$$

where $W^V \in \mathcal{R}^{V*H}, b^V \in \mathcal{R}^H, W^C \in \mathcal{R}^{V*H}, b^c \in \mathcal{R}^H$ are weight parameters.

The cost function L is the negative log-likelihood as the following equation:

$$L = -\sum_i (\log p(\hat{v}_i|h_i) + \log p(\hat{c}_i|h_i)) \tag{20}$$

4 Experiments

4.1 Data set

To evaluate the proposed network, we utilized two domain document sets: seven insurance product leaflets (4,695 sentences) and 44 Wikipedia documents (2,655 sentences) about Kamakura, a famous sightseeing place in Japan. All documents are written (in Japanese) in HTML format.

Each sentence in the documents is annotated with labels that represent the value of pre-defined attributes and conditions as determined by an expert. For example, the sentence *"Special contract for individual"* is labeled as the value of attribute *"special contract"* and as *condition*.

We defined 37 attributes for insurance domain, such as *"Special contract"*, *"Reasonable cause for payment"*, *" No reasonable cause for payment"* and *"NIL"* and 27 attributes for tourism domain, such as *"Overview"*, *"Origin of the name"*, *"Famous product"* and *"NIL"* .

4.2 Experimental settings

We evaluated the quality of information extraction by judging whether the extracted values matched the annotated labels, excluding the *"NIL"* label which means that the sentence cannot be represented as pre-defined attributes. We defined the correct values as those for which the annotated labels of the sentence include the extracted values of attributes except the *"NIL"* label. We used a micro-averaged F_1 score as the evaluation metric for the seven insurance domain documents by applying leave-one-out cross-validation and for the 44 tourism domain documents by applying 10 fold cross-validation.

A comparison of our method with other methods follows.

- Baseline MaxEnt: This is a method using a maximum entropy model that selects the $|V|$ most frequent words from the training dataset and uses the count of each word as features. We consider this to be the baseline method for classifying a sentence into value and condition classes using simply words as input features.

- (proposed) HN: This is a method using the hierarchical network described in Section 3. The network captures semantic representations of word- and sentence- sequential data and classifies each sentence in a document into attributes and condition classes.

- HN-word: This is a method using a network that has the same architecture as the proposed network but has no output layer or sentence encoder. The network takes only the word-sequential information as input features to classify a sentence into value and condition classes. We used this method to evaluate the effects of using sentence-sequential information to classify a sentence.

- HN-sent: This is a method using a network that has the same architecture as the proposed network but has no word encoder or word attention layer. The network ignores the word-sequential information and uses the count of each word as features in classifying a sentence into value and condition classes. We used this method to evaluate the effects of using word-sequential information and an attention mechanism to classify a sentence.

4.3 Model parameters

The hyper parameters of the models for the four methods above were tuned experimentally. In our experiments, we set the word embedding dimension E to be 100 and hidden layer dimension H to be 200. The size of the vocabulary for words V and HTML tags V^t respectively are set to be 4000 and 50. We selected words in the vocabulary as V words of the highest frequency in all datasets.

We used Chainer (Tokui et al., 2015), a framework of neural networks, for implementing our architecture. We used Adadelta (Zeiler, 2012) to train all models and a mini-batch size of 32. In total, 20 training epochs were used.

4.4 Results

Table 1 shows micro-averaged precision, recall, and F_1 scores for the test dataset when each method was trained with a train dataset. The F_1 scores seemed to be generally low as classification tasks since we ignored the *"NIL"* label, which accounts for about 60% of the total dataset. These scores were used to evaluate the quality of information extraction.

Method HN, the use of the proposed network, achieved the best F_1 values in both the insurance and tourism domains and performed statistically significantly better than the baseline method MaxEnt in all of the experiments. Except for condition classification results in the tourism domain, methods HN-sent and HN performed significantly better than methods MaxEnt and HN-word. These results shows that sentence-sequential information is effective in classifying sentences into attribute and condition classes.

Table 2 shows some examples classification results for each method. It can be seen that some sentences in a listing structure were classified into the same attribute class *"No reasonable cause for payment"* correctly by methods HN-sent and HN. This is because they capture semantic representation of sentence-sequential data and thus classified sentences in a listing structure that are close in meaning into the same class *"No reasonable cause for payment"*. On the other hand, MaxEnt and HN-word, which

		Attribute			Condition		
domain	method	P	R	F_1	P	R	F_1
insurance	MaxEnt (baseline)	0.504	0.463	0.483	0.227	0.344	0.274
	HN-word	0.537	0.461	0.496^{**}	0.406	0.301	0.346^{**}
	HN-sent	0.592	0.564	0.578^{**}	0.461	0.291	0.357^{**}
	HN	**0.611**	**0.582**	$\mathbf{0.596}^{**,\dagger}$	**0.455**	**0.328**	$\mathbf{0.381}^{**,\dagger}$
tourism	MaxEnt (baseline)	0.436	0.293	0.350	0.500	0.382	0.433
	HN-word	0.443	0.320	0.371^{**}	0.652	0.417	0.509^{*}
	HN-sent	0.556	0.438	0.490^{**}	0.650	0.361	0.464
	HN	**0.562**	**0.459**	$\mathbf{0.505}^{**}$	**0.667**	**0.444**	$\mathbf{0.533}^{**}$

Table 1: Micro-averaged F_1 scores for test datasets. Asterisks mean there is a significant difference between the F_1 score obtained for the method indicated and the F_1 score obtained for the baseline method. Daggers mean there is a significant difference between the F_1 score obtained for the method and the next largest F_1 score obtained for another method. (*,†: $p < .05$, **: $p < .01$)

		Predicted label				
HTML tag	Sentence	MaxEnt	HN-word	HN-sent	HN	Correct label
<h3>	Reason for not paying benefits	NIL	NIL	NIL	NIL	NIL
<h4>	(the primary contract)	NIL	NIL	NIL	NIL	NIL
<li>	State of health differs from that reported.	*No reasonable cause for payment*	*No reasonable cause for payment*	*No reasonable cause for payment*	*No reasonable cause for payment*	*No reasonable cause for payment*
<li>	Hospitalization due to injury caused before indemnity period.	*Reasonable cause for payment*	*Reasonable cause for payment*	*No reasonable cause for payment*	*No reasonable cause for payment*	*No reasonable cause for payment*
<li>	Hospitalization for reasons other than treatment / unnecessary hospitalization.	*Reasonable cause for payment*	*Reasonable cause for payment*	*Reasonable cause for payment*	*No reasonable cause for payment*	*No reasonable cause for payment*
<li>	Check policy summary for details.	NIL	NIL	NIL	NIL	NIL

Table 2: Example classification results for each method

ignore the sentence-sequential information, incorrectly classified such sentences into the attribute class *”Reasonable cause for payment”*. The reason for these results is that the word information for these sentences is not sufficient for classifying the sentences into the correct attribute classes.

5 Conclusion

This paper described a hierarchical neural network for extracting structured data from product descriptions. The network classifies each sentence into attribute and condition classes jointly in two steps on the basis of word sequences and sentence sequences in the document. First, the network obtains sentence semantics by aggregating important words into sentence vectors. Then it classifies each sentence into pre-defined attribute classes and condition classes incorporated with sentence sequences.

Experimental results demonstrated that the method using the proposed network significantly outperformed baseline methods by taking semantic representation of word and sentence sequential data into account. We found that sentence-sequential information was effective in extracting sentence-level values of product attributes from web documents while word information was insufficient for extracting sentence-level values.

To obtain concrete information that can be used in question answering systems, it is helpful to extract relational information between attribute value and condition sentences. Addressing the problems involved in extracting relationships between sentences remains as a subject for our future work.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th Int ' l Semantic Web Conference, Busan, Korea*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in Neural Information Processing Systems*, pages 577–585.

Pankaj Gulhane, Amit Madaan, Rupesh Mehta, Jeyashankher Ramamirtham, Rajeev Rastogi, Sandeep Satpal, Srinivasan H Sengamedu, Ashwin Tengli, and Charu Tiwari. 2011. Web-scale information extraction with vertex. In *Proceedings of the IEEE 27th International Conference on Data Engineering*, pages 1209–1220. IEEE.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Abhyuday N Jagannatha and Hong Yu. 2016. Bidirectional rnn for medical event detection in electronic health records. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 473–482.

Mahesh Joshi, Ethan Hart, Mirko Vogel, and Jean-David Ruvini. 2015. Distributed word representations improve ner for e-commerce. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 160–167.

Shoushan Li, Lei Huang, Rong Wang, and Guodong Zhou. 2015. Sentence-level emotion classification with label and context dependence. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1045–1053.

Ion Muslea, Steve Minton, and Craig Knoblock. 1999. A hierarchical approach to wrapper induction. In *Proceedings of the third annual conference on Autonomous Agents*, pages 190–197. ACM.

István Nagy and Richárd Farkas. 2012. Person attribute extraction from the textual parts of web pages. *Acta Cybern.*, 20(3):419–440.

Martina Naughton, Nicola Stokes, and Joe Carthy. 2008. Investigating statistical techniques for sentence-level event classification. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 617–624. Association for Computational Linguistics.

Hitoshi Nishikawa, Kugatsu Sadamitsu, Chiaki Miyazaki, hisako Asano, Toshiro Makino, and Yoshihiro Matsuo. 2015. Query-oriented extractive summarization for the wikipedia articles using document structure. In *Proceedings of the annual meeting of the Association for Natural Language Processing*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. 2015. Chainer: a next-generation open source framework for deep learning. In *Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Twenty-ninth Annual Conference on Neural Information Processing Systems (NIPS)*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2(3):5.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.