Chinese Zero Pronoun Resolution with Deep Memory Network

Qingyu Yin, Yu Zhang, Weinan Zhang, Ting Liu[*]
Research Center for Social Computing and Information Retrieval
Harbin Institute of Technology, China
{qyyin, yzhang, wnzhang, tliu}@ir.hit.edu.cn

Abstract

Existing approaches for Chinese zero pronoun resolution typically utilize only syntactical and lexical features while ignoring semantic information. The fundamental reason is that zero pronouns have no descriptive information, which brings difficulty in explicitly capturing their semantic similarities with antecedents. Meanwhile, representing zero pronouns is challenging since they are merely gaps that convey no actual content. In this paper, we address this issue by building a deep memory network that is capable of encoding zero pronouns into vector representations with information obtained from their contexts and potential antecedents. Consequently, our resolver takes advantage of semantic information by using these continuous distributed representations. Experiments on the OntoNotes 5.0 dataset show that the proposed memory network could substantially outperform the state-of-the-art systems in various experimental settings.

1 Introduction

A zero pronoun (ZP) is a gap in a sentence, which refers to an entity that supplies the necessary information for interpreting the gap (Zhao and Ng, 2007). A ZP can be either anaphoric if it corefers to one or more preceding noun phrases (antecedents) in the associated text, or non-anaphoric if there are no such noun phrases. Below is an example of ZPs and their antecedents, where “ϕ” denotes the ZP.

[警方] 表示 他们 自杀 的 可能性 很高， 不过 ϕ_1 也 不 排除 ϕ_2 有 他杀 的 可能。

([The police] said that they are more likely to commit suicide, but ϕ_1 could not rule out ϕ_2 the possibility of homicide.)

In this example, the ZP “ϕ_1” is an anaphoric ZP that refers to the antecedent “警方/The police” while the ZP “ϕ_2” is non-anaphoric. Unlike overt pronouns, ZPs lack grammatical attributes such as gender and number that have been proven to be essential in pronoun resolution (Chen and Ng, 2014a), which makes ZP resolution a more challenging task than overt pronoun resolution.

Automatic Chinese ZP resolution is typically composed of two steps, i.e., anaphoric zero pronoun (AZP) identification that identifies whether a ZP is anaphoric; and AZP resolution, which determines antecedents for AZPs. For AZP identification, state-of-the-art resolvers use machine learning algorithms to build AZP classifiers in a supervised manner (Chen and Ng, 2013, 2016). For AZP resolution, literature approaches include unsupervised methods (Chen and Ng, 2014b, 2015), feature-based supervised models (Zhao and Ng, 2007; Kong and Zhou, 2010), and neural network models (Chen and Ng, 2016). Neural network models for AZP resolution are of growing interest for their capacity to learn task-specific representations without extensive feature engineering and to effectively exploit lexical information for ZPs and their candidate antecedents in a more scalable manner than feature-based models.

Despite these advantages, existing supervised approaches (Zhao and Ng, 2007; Chen and Ng, 2013, 2016) for AZP resolution typically utilize only syntactical and lexical information through features. They overlook semantic information that is regarded as an important factor in the resolution of common noun phrases (Ng, 2007). The fundamental reason is that ZPs have no descriptive information, which results in difficulty in calculating semantic similarities and relatedness scores

[*] Email corresponding.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1318–1327
Copenhagen, Denmark, September 7–11, 2017. ©2017 Association for Computational Linguistics

between the ZPs and their antecedents. Therefore, the proper representations of ZPs are required so as to take advantage of semantic information when resolving ZPs. However, representing ZPs is challenging because they are merely gaps that convey no actual content.

One straightforward method to address this issue is to represent ZPs with supplemental information provided by some available components, such as contexts and candidate antecedents. Motivated by Chen and Ng (2016) who encode a ZP's lexical contexts by utilizing its preceding word and governing verb, we notice that a ZP's context can help to describe the ZP itself. As an example of its usefulness, given the sentence "ϕ taste spicy", people may resolve the ZP "ϕ" to the candidate antecedent "red peppers", but can hardly regard "my shoes" as its antecedent, because they naturally look at the ZP's context "taste spicy" to resolve it ("my shoes" cannot "taste spicy"). Meanwhile, considering that the antecedents of a ZP provide the necessary information for interpreting the gap (ZP), it is a natural way to express a ZP by its potential antecedents. However, only some subsets of candidate antecedents are needed to represent a ZP[1]. To achieve this goal, a desirable solution should be capable of explicitly capturing the importance of each candidate antecedent and using them to build up the representation for the ZP.

In this paper, inspired by the recent success of computational models with attention mechanism and explicit memory (Sukhbaatar et al., 2015; Tang et al., 2016; Kumar et al., 2015), we focus on AZP resolution, proposing the zero pronoun-specific memory network (**ZPMN**) that is competent for representing a ZP with information obtained from its contexts and candidate antecedents. These representations provide our system with an ability to take advantage of semantic information when resolving ZPs. Our **ZPMN** consists of multiple computational layers with shared parameters. With the underlying intuition that not all candidate antecedents are equally relevant for representing the ZP, we develop each computational layer as an attention-based model, which first learns the importance of each candidate antecedent and then utilizes this information to calculate the continuous distributed representation of the ZP. The attention weights over candidate antecedents with respect to the ZP's representation obtained by the last layer are regarded as the ZP coreference classification result. Given that every component is differentiable, the entire model could be efficiently trained end-to-end with gradient descent.

We evaluate our method on the Chinese portions of the OntoNotes 5.0 corpus by comparing with the baseline systems in different experimental settings. Results show that our approach significantly outperforms the baseline algorithms and achieves state-of-the-art performance.

2 Zero Pronoun-specific Memory Network

We describe our deep memory network approach for AZP resolution in this section. We first give an overview of our model and then describe its components. Finally, we present the training and initialization details.

2.1 An Overview of the Method

In this part, we present an overview of the zero pronoun-specific memory network (**ZPMN**) for AZP resolution. Given an AZP zp, we first extract a set of candidate antecedents. Following Chen and Ng (2016), we regard all and only those maximal or modifier noun phrases (NPs) that precede zp in the associated text and are at most two sentences away from it, to be its candidate antecedents. Suppose k candidate antecedents are extracted, our task is to determine the correct antecedent of zp from its candidate antecedent set $\mathcal{A}(zp) = \{c_1, c_2, ..., c_k\}$.

Specifically, these candidate antecedents are represented in form of vectors $\{v_{c_1}, v_{c_2}, ..., v_{c_k}\}$, which are stacked and regarded as the external memory $mem \in \mathbb{R}^{l \times k}$, where l is the dimension of v_c. Meanwhile, we represent each word as a continuous and real-valued vector, which is known as word embedding (Bengio et al., 2003). These word vectors can be randomly initialized, or be pre-trained from text corpus with learning algorithms (Mikolov et al., 2013; Pennington et al., 2014). In this work, we adopt the latter strategy since it can better exploit the semantics of words. All the word vectors are stacked in a word embedding matrix $L_w \in \mathbb{R}^{d \times |V|}$, where d is the dimension of the word vector and $|V|$ is the size of the word vocabulary. The embedding of word w is

[1] A common way to do this task is to first extract a set of candidate antecedents, and then select antecedents from the candidate set. Therefore, only those candidates who are possibly the correct antecedent of the given ZP are suitable for interpreting it.

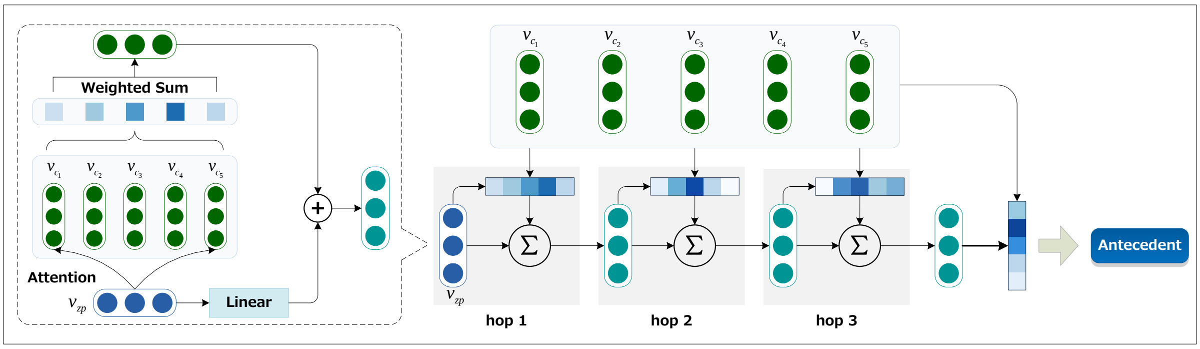


Figure 1: Illustration of the zero pronoun-specific memory network with three computational layers (hops). v_{zp} and v_c denote the vector representation of an AZP and its candidate antecedents. The left part in dashed box shows the details of the first hop.

notated as $e \in \mathbb{R}^{d \times 1}$, which is the column in L_w.

An illustration of **ZPMN** is given in Figure 1, which is inspired by the memory network utilized in question answering (Sukhbaatar et al., 2015). Our model consists of multiple computational layers, each of which contains an attention layer and a linear layer. First, we represent the AZP zp by utilizing its contextual information, that is, proposing the ZP-centered LSTM that encodes zp into its distributed vector representation (i.e. v_{zp} in Figure 1). We then regard v_{zp} as the initial representation of zp, and feed it as the input to the first computational layer (hop 1). In the first computational layer, we calculate the attention weight across the AZP for each candidate antecedent, by which our model adaptively selects important information from the external memory (candidate antecedents). The output of the attention layer and the linear transformation of v_{zp} are summed together as the input of to the next layer (hop 2).

We stack multiple hops by repeating the same process for multiple times in a similar manner. We call the abstractive information obtained from the external memory the "*key extension*" of the AZP. Note that the attention and linear layer parameters are shared in different hops. Regardless of the number of hops the model employs, they utilize the same number of parameters. Finally, after going through all the hops, we regard the attention weight of each candidate antecedent with respect to the AZP representation generated by the last hop as the probability that the candidate antecedent is the correct antecedent, and predict the highest-scoring (most probable) one to be the antecedent of the given AZP.

2.2 Modeling Zero Pronouns by Contexts

A vector representation of AZP is required when computing the **ZPMN**. As aforementioned, a ZP contains no actual content, it is therefore needed to employ some supplemental information to generate its initial representation. To achieve this goal, we develop the ZP-centered LSTM that encodes an AZP into a vector representation by utilizing its contextual information.

Admittedly, one efficient method to model a variable-length sequence of words (context words) is to utilize a recurrent neural network (Elman, 1991). A recurrent neural network (RNN) stores the sequence history in a real-valued history vector, which captures information of the whole sequence. LSTM (Hochreiter and Schmidhuber, 1997) is one of the classical variations of RNN that mitigate the gradient vanish problem of RNN. Assuming $x = \{x_1, x_2, ..., x_n\}$ is an input sequence, each time step t has an input x_t and a hidden state h_t. The internal mechanics of the LSTM is defined by:

$$i_t = \sigma(W^{(i)} \cdot [x_t; h_{t-1}] + b^{(i)}) \tag{1}$$

$$f_t = \sigma(W^{(f)} \cdot [x_t; h_{t-1}] + b^{(f)}) \tag{2}$$

$$o_t = \sigma(W^{(o)} \cdot [x_t; h_{t-1}] + b^{(o)}) \tag{3}$$

$$\tilde{C}_t = tanh(W^{(c)} \cdot [x_t; h_{t-1}] + b^{(c)}) \tag{4}$$

$$C_t = i_t \odot \tilde{C}_t + f_t \odot C_{t-1} \tag{5}$$

$$h_t = o_t \odot tanh(C_t) \tag{6}$$

where $\odot$ is an element-wise product and $W^{(i)}$, $b^{(i)}$, $W^{(f)}$, $b^{(f)}$, $W^{(o)}$, $b^{(o)}$, $W^{(c)}$, and $b^{(c)}$ are the parameters of the LSTM network.

Intuitively, the words near an AZP generally contain richer information to express it. To bet-

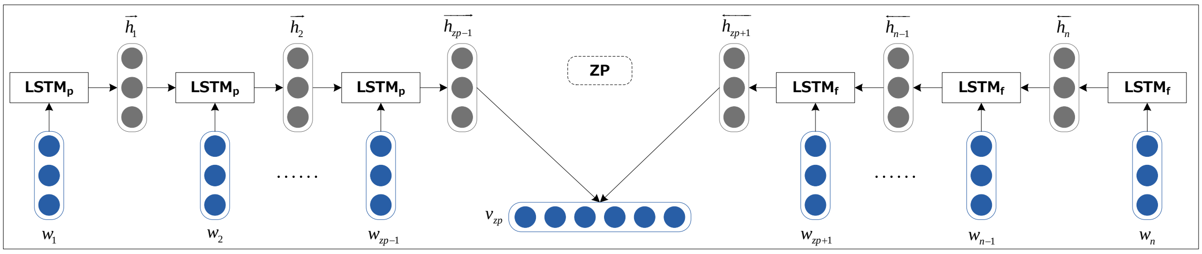


Figure 2: ZP-centered LSTM for encoding the AZP by its context words. w_i means the i-th word in the sentence, w_{zp-i} is the i-th last word before the ZP and w_{zp+i} is the i-th word behind the ZP.

ter utilize the information of words surrounding the AZP, on the basis of the traditional LSTM, we propose the ZP-centered LSTM to encode the AZPs. A graphical representation of this model is displayed in Figure 2. Specifically, the ZP-centered LSTM contains two standard LSTM neural networks, i.e., the LSTM$_p$ that encodes the preceding context of the AZP in a left-to-right manner, and the LSTM$_f$ that models the following context in the reverse direction. Ideally, the ZP-centered LSTM models the preceding and following contexts of the AZP separately, so that the words near the AZP are regarded as the last hidden units and could contribute more in representing the AZP. Afterward, we obtain the representation of the AZP by concatenating the last hidden vectors of LSTM$_p$ and LSTM$_f$, which summarizes the useful contextual information centered around the AZP. Averaging or summing the last hidden vectors of LSTM$_p$ and LSTM$_f$ could also be attempted as alternatives. We regard it as the initial vector representation of the AZP and feed it to the first computational layer to go through the remaining procedures of our system.

2.3 Generating the External Memory

We describe our method for generating the external memory in this subsection. For a given AZP, a set of noun phrases (NPs) is extracted as its candidate antecedents. Specifically, we generate the external memory by utilizing these candidate antecedents. One way to encode an NP candidate is to utilize its head word embedding (Chen and Ng, 2016). However, this method has a major drawback of not utilizing contextual information that is essential for representing a phrase. Besides, some approaches (Socher et al., 2013; Sun et al., 2015) encode a phrase by utilizing the average word embedding it contains. We argue that such an averaging operation simply treats all the words in a phrase equally, which is inaccurate because some words might be more informative than others.

A helpful property of LSTM is that it could keep useful history information in the memory cell by exploiting input, output and forget gates to decide how to utilize and update the memory of previous information. Given a sequence of words $\{w_1, w_2, ..., w_n\}$, previous research (Sutskever et al., 2014) utilizes the last hidden vector of LSTM to represent the information of the whole sequence. For word w_t in a sequence, its corresponding hidden vector h_t can capture useful information before and including w_t.

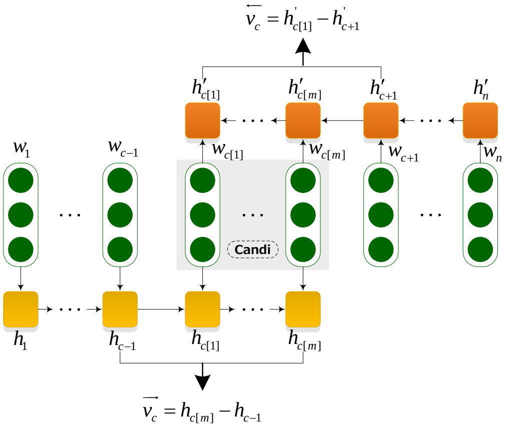


Figure 3: Illustration for modeling a candidate antecedent through its context and content words. *Candi* represents the candidate antecedent. Suppose the candidate antecedent contains m words, $w_{c[j]}$ denotes its j-th word. w_i is the i-th word in the sentence, and $w_{c+1(-1)}$ is the word appears immediately after (before) the candidate antecedent.

Inspired by this, we propose a novel method to produce representations of the candidate antecedents by utilizing both their contexts and content words. Specifically, we use the subtraction between LSTM hidden vectors to encode the candidate antecedents, as illustrated in Figure 3. Given a candidate antecedent c with m words, two standard LSTM neural networks are employed for encoding c in the forward and backward direction, respectively. For the forward LSTM, we extract a sequence of words related with c in a left-to-right manner, i.e., $\{w_1, w_2, ..., w_{c-1}, w_{c[1]}, ..., w_{c[m]}\}$. Subsequently, the forward vector representation of c can be calculated as $\overrightarrow{v_c} = h_{c[m]} - h_{c-1}$, where $h_{c[m]}$ and h_{c-1} indicate the hidden vectors of the forward LSTM corresponding to $w_{c[m]}$ and w_{c-1}, respectively. Meanwhile, the backward LSTM models a sequence of words that are extracted in the reverse direction, that is, $\{w_n, w_{n-1}, ..., w_{c+1}, w_{c[m]}, ..., w_{c[1]}\}$. We then perform the similar operation, computing the backward representation of c as $\overleftarrow{v_c} = h'_{c[1]} - h'_{c+1}$, where $h'_{c[1]}$ and h'_{c+1} indicate the hidden vectors of the backward LSTM corresponding to $w_{c[1]}$ and w_{c+1}. Finally, we concatenate these two vectors together as the ultimate vector representation of c, $v_c = \overrightarrow{v_c} || \overleftarrow{v_c}$.

This method enables our model to encode a candidate antecedent by the information both outside and inside the phrase, which provides our model a strong ability to access to sentence-level information when modeling the candidate antecedents. In this manner, we generate the vector representations of the candidate antecedents, and regard them as the external memory, i.e., $mem = \{v_{c_1}, v_{c_2}, ..., v_{c_k}\}$.

2.4 Attention Mechanism

In this part, we introduce our attention mechanism. This strategy has been widely used in many nature language processing tasks, such as factoid question answering (Hermann et al., 2015), entailment (Rocktäschel et al., 2015) and disfluency detection (Wang et al., 2016). The basic idea of attention mechanism is that it assigns a weight/importance to each lower position when computing an upper-level representation (Bahdanau et al., 2015). With the underlying intuition that not all candidate antecedents are equally relevant for representing the AZP, we employ the attention mechanism as to dynamically align the more informative candidate antecedents from the external memory, $mem = \{v_{c_1}, v_{c_2}, ..., v_{c_k}\}$ with regard to the given AZP, and use them to build up the representation of the AZP.

As shown in Chen and Ng (2016), traditional hand-crafted features are crucial for the resolver's success since they capture the syntactic, positional and other relationships between an AZP and its candidate antecedents. Therefore, to evaluate the importance of each candidate antecedent in a comprehensive manner, following Chen and Ng (2016) who encode hand-crafted features as inputs to their network, we integrate a set of features that are utilized in Chen and Ng (2016), in the form of vector ($v^{(feature)}$) into our attention model. For each multi-valued feature, we convert it into a corresponding set of binary-valued features[2].

Specifically, for the t-th candidate antecedent in the memory, v_{c_t}, taking the vector representation of the AZP v_{zp} and the corresponding feature vector $v_t^{(feature)}$ as inputs, we compute the attention score as $\alpha_t = G(v_{c_t}, v_{zp}, v_t^{(feature)})$. The scoring function G is defined by:

$$s_t = tanh(W^{(att)} \cdot [v_{c_t}; v_{zp}; v_t^{(feature)}] + b^{(att)}) \tag{7}$$

$$\alpha_t = \frac{exp(s_t)}{\sum_{t'=1}^{k} exp(s_{t'})} \tag{8}$$

where $W^{(att)}$ and $b^{(att)}$ are the attention parameters and k indicates the number of candidate antecedents. After obtaining the attention scores for all the candidate antecedents $\{a_1, a_2, ..., a_k\}$, our attention layer outputs a continuous vector vec that is computed as the weighted sum of each piece of memory in mem:

$$vec = \sum_{i=1}^{k} \alpha_i v_{c_i} \tag{9}$$

2.5 Training Details

We initialize our word embeddings with 100 dimensional ones produced by the $word2vec$ toolkit (Mikolov et al., 2013) on the Chinese portion of the training data from the OntoNotes 5.0 corpus. We randomly initialize the parameters from a uniform distribution $U(-0.03, 0.03)$ and minimize the training objective using stochastic gradient descent with learning rate equals to 0.01. In addition, to regularize the network, we apply L2 regularization to the network weights and dropout with a rate of 0.5 on the output of each hidden layer.

[2] If one feature has k different values, we will convert it into k binary features.

The model is trained in a supervised manner by minimizing the cross-entropy error of ZP coreference classification. Suppose the training set contains N AZPs $\{zp_1, zp_2, ..., zp_N\}$. Let $\mathcal{A}(zp_i)$ denote the set of candidate antecedents of an AZP zp_i, and $P(c|zp_i)$ represents the probability of predicting candidate c as the antecedent of zp_i (i.e., the attention weight of candidate antecedent c with respect to the AZP representation generated by the last hop), the loss is given by:

$$loss = -\sum_{i=1}^{N} \sum_{c \in \mathcal{A}(zp_i)} \delta(zp_i, c) log(P(c|zp_i)) \tag{10}$$

where $\delta(zp, c)$ is 1 or 0, indicating whether zp and c are coreferent.

3 Experiments

3.1 Experimental Setup

Datasets: Following Chen and Ng (2016, 2015), we run experiments on the Chinese portion of the OntoNotes Release 5.0 dataset[3] used in the CoNLL 2012 Shared Task (Pradhan et al., 2012). The dataset consists of three parts, i.e., a training set, a development set and a test set. Since only the training set and the development set contain ZP coreference annotations, we train our model on the training set and utilize the development set for testing purposes. Meanwhile, we reserve 20% of the training set as a held-out development set for tuning the hyperparameters of our network. The same experimental data setting is utilized in the baseline system (Chen and Ng, 2016). Table 1 shows the statistics of our corpus. Besides, documents in the datasets come from six sources, i.e., broadcast news (BN), newswires (NW), broadcast conversations (BC), telephone conversations (TC), web blogs (WB) and magazines (MZ).

	Documents	Sentences	Words	AZPs
Training	1,391	36,487	756K	12,111
Test	172	6,083	110K	1,713

Table 1: Statistics on the training and test corpus.

Evaluation metrics: Same as previous studies on Chinese ZP resolution (Zhao and Ng, 2007; Chen and Ng, 2016), we use three metrics to evaluate the quality of our model: recall, precision and F-score (denoted as R, P and F, respectively).

Experimental settings: We employ three Chinese ZP resolution systems as our baselines, i.e., Zhao and Ng (2007); Chen and Ng (2015, 2016). Consistent with Chen and Ng (2015, 2016), three experimental settings are designed to evaluate our approach. In Setting 1, we directly employ the gold syntactic parse trees and gold AZPs that are obtained from the OntoNotes dataset. In Setting 2, we utilize gold syntactic parse trees and system (automatically identified) AZPs[4]. In Setting 3, we employ system AZP and system syntactic parse trees that obtained through the Berkeley parser[5], which is the state-of-the-art parsing model.

3.2 Experimental Results

Table 2 shows the experimental results of the baseline systems and our model on entire test set. Our approach is abbreviated to ZPMN (k), where k indicates the number of hops. The best methods in each of the three experimental settings are in **bold** text. From Table 2, we can observe that our approach outperforms all previous baseline systems by a substantial margin. Meanwhile, among all our models from single hop to six hops, using more computational layers could generally lead to better performance. The best performance is achieved by the model with six hops under experimental Setting 1 and 2, and with four hops in experimental Setting 3. Furthermore, the **ZPMN** (with six hops) significantly outperforms the state-of-the-art baseline system (Chen and Ng, 2016) under three experimental settings by 2.7%, 2.7%, and 3.9% in terms of overall F-score[6], respectively. In all words, our model is an extremely strong performer and substantially outperforms baseline methods, which demonstrate the efficiency of the proposed zero pronoun-specific memory network.

It is well accepted that computational models that are composed of multiple processing layers could learn representations of data with multiple levels of abstraction (LeCun et al., 2015). In our approach, multiple computation layers allow the model to learn representations of AZPs with multiple levels of abstraction generated by candidate antecedents. Each layer/hop retrieves important candidate antecedents, and transforms the repre-

[3] http://catalog.ldc.upenn.edu/LDC2013T19

[4] In this study, we adopt the learning-based method utilized in (Chen and Ng, 2016) to identify system AZPs, including the location and identification of AZPs.

[5] https://github.com/slavpetrov/berkeleyparser

[6] All significance tests are paired t-tests, with $p < 0.05$.

	Setting 1			Setting 2			Setting 3		
	Gold Parse + Gold AZP			Gold Parse + System AZP			System Parse + System AZP		
	R	P	F	R	P	F	R	P	F
Zhao and Ng (2007)	41.5	41.5	41.5	22.4	24.4	23.3	12.7	14.2	13.4
Chen and Ng (2015)	50.0	50.4	50.2	35.7	26.2	30.3	19.6	15.5	17.3
Chen and Ng (2016)	51.8	52.5	52.2	**39.6**	27.0	32.1	21.9	15.8	18.4
ZPMN (1)	53.0	53.3	53.1	37.9	30.0	33.4	27.8	17.4	21.4
ZPMN (2)	53.7	54.0	53.9	38.8	30.6	34.0	28.1	18.2	22.1
ZPMN (3)	53.9	54.2	54.1	38.6	30.4	34.2	28.2	17.7	21.7
ZPMN (4)	54.4	54.7	54.5	39.0	30.7	34.3	**29.3**	**18.5**	**22.7**
ZPMN (5)	54.1	54.4	54.3	38.8	30.6	34.2	28.6	17.8	22.0
ZPMN (6)	**54.8**	**55.1**	**54.9**	39.4	**31.1**	**34.8**	28.9	18.2	22.3

Table 2: Experimental results on the test data. ZPMN represents the proposed zero pronoun-specific memory network model, and the number beside ZPMN in each row denotes the number of hops.

	Setting 1: Gold Parse + Gold AZP						Setting 2: Gold Parse + System AZP						Setting 3: System Parse + System AZP					
	Baseline			**ZPMN**			Baseline			**ZPMN**			Baseline			**ZPMN**		
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
NW (84)	48.8	48.8	48.8	48.8	48.8	**48.8**	34.5	26.4	29.9	39.5	34.3	**36.7**	11.9	12.8	12.3	21.0	19.9	**20.5**
MZ (162)	41.4	41.6	41.5	46.3	46.3	**46.3**	34.0	22.4	27.0	34.6	35.0	**34.8**	9.3	7.3	8.2	17.1	15.7	**16.4**
WB (284)	56.3	56.3	56.3	59.8	59.8	**59.8**	44.7	25.1	32.2	41.2	28.7	**33.8**	23.9	16.1	19.2	31.3	17.6	**22.6**
BN (390)	55.4	55.4	55.4	58.2	58.6	**58.4**	36.9	31.9	34.2	43.8	30.0	**35.6**	22.1	23.2	22.6	35.1	20.7	**26.1**
BC (510)	50.4	51.3	50.8	52.9	53.6	**53.2**	37.6	25.6	30.5	35.6	29.4	**32.2**	21.2	14.6	17.3	25.6	15.6	**19.4**
TC (283)	51.9	54.2	53.1	54.8	54.8	**54.8**	46.3	29.0	**35.6**	36.9	32.9	34.8	31.4	15.9	21.1	33.2	21.0	**25.8**

Table 3: Experimental results on each source of test data. The strongest F-score in each row is in **bold**.

sentation at previous level into a representation at a higher, slightly more abstract level. We regard this representation as the "*key extension*" of the AZP, by which our model learns to encode the AZP in an efficient manner.

For per-source results, we conduct experiments by comparing the **ZPMN** (with six hops) with the state-of-the-art baseline system (Chen and Ng, 2016) on six sources of test data, as shown in Table 3. The rows in Table 3 are the results from different sources and the parenthesized numbers beside the source names are their corresponding numbers of AZPs. In experimental Settings 1 and 3, **ZPMN** improves results further across all the six sources of data. Under experimental Setting 2, our model outperforms the baseline system in five of the six sources of data, only slightly underperforms in source TC. All these prove that our approach achieves a considerable improvement in Chinese ZP resolution.

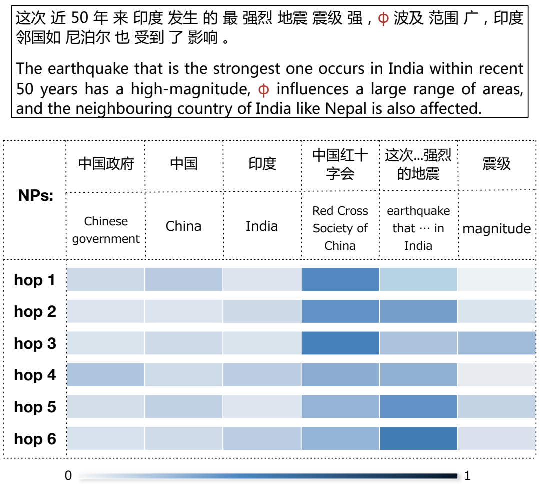
Moreover, to evaluate the effectiveness of our methods for modeling the AZP and candidate antecedents proposed in Section 2.2 and 2.3, we compare with three models that are all simplified versions of the **ZPMN**, namely, **ZPContextFree** where an AZP is initially represented by its governing verb and preceding word; **AntContentAvg** where the candidate antecedents are encoded by their averaged content word embeddings; and **AntContentHead** where each candidate antecedent is represented by the embedding of its head word. To make comparison as fair as possible, we keep the other parts of these models unchanged from the **ZPMN** with six computational layers (hop 6). To minimize the external influence, we run experiments under experimental Setting 1 (gold parse and gold AZPs). Table 4 shows the results.

	R	P	F
ZPContextFree	53.5	53.8	53.6
AntContentAvg	52.6	52.9	52.7
AntContentHead	53.8	54.1	53.9
ZPMN (hop 6)	54.8	55.1	**54.9**

Table 4: Experimental results of different models.

With an intuition that contexts of an AZP provide more sufficient information than only a few specific of words in expressing the AZP, the performance of **ZPContextFree** is unsurprisingly worse than that of the **ZPMN**, which reflects the effects of the ZP-centered LSTM proposed to generate the initial representation for the AZP. In addition, the performance of **AntContentAvg** is relatively low. We attribute this to the model assigning the same importance to all the content words in a phrase, which causes difficulty for the model to capture informative words in a candidate antecedent. Meanwhile, **AntContent-Head** only models limited information when encoding candidate antecedents, thereby underperforms the **ZPMN** whose external memory contains sentence-level information both outside and inside the candidate antecedents. These demonstrate the utility of the method for modeling candidate antecedents.

3.3 Attention Model Visualization



这次 近 50 年 来 印度 发生 的 最 强烈 地震 震级 强，φ 波及 范围 广，印度 邻国如 尼泊尔 也 受到 了 影响。

The earthquake that is the strongest one occurs in India within recent 50 years has a high-magnitude, φ influences a large range of areas, and the neighbouring country of India like Nepal is also affected.

Figure 4: Example of attention weights in different hops. ZP is denoted as ϕ. The rows show the attention weights of candidates in each hop. Darker color means higher weight.

To obtain a better understanding of our deep memory network, we visualize the attention weights of the **ZPMN**, as is shown in Figure 4. We can observe that in the first three hops, the fourth candidate "中国红十字会/Red Cross Society of China" gains a higher attention weight than the others. Nevertheless, in hop 5 and 6, the attention weight of "这次...强烈的地震/the earthquake that ... in India" increases and the model finally predicts it correctly as the antecedent. This case illustrates the effects of multiple hops.

4 Related Work

4.1 Zero Pronoun Resolution

Chinese zero pronoun resolution. Early studies utilize heuristic rules to resolve ZPs in Chinese (Converse, 2006; Yeh and Chen, 2007). More recently, supervised approaches have been vastly explored. Zhao and Ng (2007) first present a machine learning approach to identify and resolve ZPs. By employing the J48 decision tree algorithm, various kinds of features are integrated into their model. Kong and Zhou (2010) develop a kernel-based approach, employing context-sensitive convolution tree kernels to model syntactic information. Chen and Ng (2013) further extend the study of Zhao and Ng (2007) by proposing several novel features and introducing the coreference links between ZPs. Despite the effectiveness of feature engineering, it is labor intensive and highly relies on annotated corpus. To handle these weaknesses, Chen and Ng (2014b) propose an unsupervised method. They first recover each ZP into ten overt pronouns and then apply a ranking model to rank the antecedents. Chen and Ng (2015) propose an end-to-end unsupervised probabilistic model, utilizing a salience model to capture discourse information. In recent years, Chen and Ng (2016) develop a deep neural network approach to learn useful task-specific representations and effectively exploit lexical features through word embeddings. Different from previous studies, in this work, we propose a novel memory network to perform the task. By encoding ZPs and candidate antecedents through the composition of texts based on the representation of words, our model benefits from the semantic information when resolving the ZPs.

Zero pronoun resolution for other languages. There have been various studies on ZP resolution for other languages besides Chinese. Ferrández and Peral (2000) propose a set of hand-crafted rules for resolving ZPs in Spanish texts. Recently, supervised approaches have been widely exploited for ZP resolution in Korean (Han, 2006), Italian (Iida and Poesio, 2011) and Japanese (Isozaki and Hirao, 2003; Iida et al., 2006, 2007; Imamura et al., 2009; Sasano and Kurohashi, 2011; Iida and Poesio, 2011; Iida et al., 2015). Iida et al. (2016) propose a multi-column convolutional neural network for Japanese intra-sentential subject zero anaphora resolution, where both the surface word sequence and dependency tree of a target sentence are exploited as clues in their model.

4.2 Attention and Memory Network

Attention mechanisms have been widely used in many studies and have achieved promising performances on a variety of NLP tasks (Rocktäschel et al., 2015; Rush et al., 2015; Liu et al., 2017). Recently, the memory network has been proposed and applied to question answering task (Weston et al., 2014), which is defined to have four components: input (I), generalization (G), output (O) and response (R). After then, memory networks have been adopted in many other NLP tasks, such as aspect sentiment classification (Tang et al., 2016), dialog systems (Dodge et al., 2015), and information extraction (Xiaocheng et al., 2017).

5 Conclusion

In this study, we propose a novel zero pronoun-specific memory network that is capable of encoding zero pronouns into the vector representations with supplemental information obtained from their contexts and candidate antecedents. Consequently, these continuous distributed vectors provide our model with an ability to take advantage of the semantic information when resolving zero pronouns. We evaluate our method on the Chinese portion of OntoNotes 5.0 dataset and report substantial improvements over the state-of-the-art systems in various experimental settings.

Acknowledgments

We greatly thank Yiming Cui and Xuxiang Wang for their tremendously helpful discussions. We also thank the anonymous reviewers for their valuable comments. This work was supported by the National High Technology Development 863 Program of China (No.2015AA015407), National Natural Science Foundation of China (No.61472105 and No.61472107).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*.

Yoshua Bengio, Rejean Ducharme, and Pascal Vincent. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Chen Chen and Vincent Ng. 2013. Chinese zero pronoun resolution: Some recent advances. In *EMNLP*, pages 1360–1365.

Chen Chen and Vincent Ng. 2014a. Chinese overt pronoun resolution: A bilingual approach. In *AAAI*, pages 1615–1621.

Chen Chen and Vincent Ng. 2014b. Chinese zero pronoun resolution: An unsupervised approach combining ranking and integer linear programming. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Chen Chen and Vincent Ng. 2015. Chinese zero pronoun resolution: A joint unsupervised discourse-aware model rivaling state-of-the-art resolvers. In *Proceedings of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, page 320.

Chen Chen and Vincent Ng. 2016. Chinese zero pronoun resolution with deep neural networks. In *Proceedings of the 54rd Annual Meeting of the ACL.*

Susan P Converse. 2006. Pronominal anaphora resolution in chinese.

Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander Miller, Arthur Szlam, and Jason Weston. 2015. Evaluating prerequisite qualities for learning end-to-end dialog systems. *arXiv preprint arXiv:1511.06931*.

Jeffrey L Elman. 1991. Distributed representations, simple recurrent networks, and grammatical structure. *Machine learning*, 7(2-3):195–225.

Antonio Ferrández and Jesús Peral. 2000. A computational approach to zero-pronouns in spanish. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 166–172. Association for Computational Linguistics.

Na-Rae Han. 2006. *Korean zero pronouns: analysis and resolution*. Ph.D. thesis, Citeseer.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

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

Ryu Iida, Kentaro Inui, and Yuji Matsumoto. 2006. Exploiting syntactic patterns as clues in zero-anaphora resolution. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 625–632. Association for Computational Linguistics.

Ryu Iida, Kentaro Inui, and Yuji Matsumoto. 2007. Zero-anaphora resolution by learning rich syntactic pattern features. *ACM Transactions on Asian Language Information Processing (TALIP)*, 6(4):1.

Ryu Iida and Massimo Poesio. 2011. A cross-lingual ilp solution to zero anaphora resolution. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 804–813. Association for Computational Linguistics.

Ryu Iida, Kentaro Torisawa, Chikara Hashimoto, Jong-Hoon Oh, and Julien Kloetzer. 2015. Intra-sentential zero anaphora resolution using subject sharing recognition. *Proceedings of EMNLP'15*, pages 2179–2189.

Ryu Iida, Kentaro Torisawa, Jong-Hoon Oh, Canasai Kruengkrai, and Julien Kloetzer. 2016. Intra-sentential subject zero anaphora resolution using multi-column convolutional neural network. In *Proceedings of EMNLP*.

Kenji Imamura, Kuniko Saito, and Tomoko Izumi. 2009. Discriminative approach to predicate-argument structure analysis with zero-anaphora resolution. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 85–88. Association for Computational Linguistics.

Hideki Isozaki and Tsutomu Hirao. 2003. Japanese zero pronoun resolution based on ranking rules and machine learning. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 184–191. Association for Computational Linguistics.

Fang Kong and Guodong Zhou. 2010. A tree kernel-based unified framework for chinese zero anaphora resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 882–891. Association for Computational Linguistics.

Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. 2015. Ask me anything: Dynamic memory networks for natural language processing. *arXiv preprint arXiv:1506.07285*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Ting Liu, Yiming Cui, Qingyu Yin, Shijin Wang, Weinan Zhang, and Guoping Hu. 2017. Effective deep memory networks for distant supervised relation extraction. In *ACL*.

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

Vincent Ng. 2007. Semantic class induction and coreference resolution. In *AcL*, pages 536–543.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40. Association for Computational Linguistics.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*.

Ryohei Sasano and Sadao Kurohashi. 2011. A discriminative approach to japanese zero anaphora resolution with large-scale lexicalized case frames. In *IJCNLP*, pages 758–766.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*, pages 1333–1339.

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.

Duyu Tang, Bing Qin, and Ting Liu. 2016. Aspect level sentiment classification with deep memory network. In *EMNLP*.

Shaolei Wang, Wanxiang Che, and Ting Liu. 2016. A neural attention model for disfluency detection. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 278–287, Osaka, Japan. The COLING 2016 Organizing Committee.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.

Feng Xiaocheng, Guo Jiang, Qin Bing, Liu Ting, and Liu Yongjie. 2017. Effective deep memory networks for distant supervised relation extraction. In *IJCAI*.

Ching-Long Yeh and Yi-Chun Chen. 2007. Zero anaphora resolution in chinese with shallow parsing. *Journal of Chinese Language and Computing*, 17(1):41–56.

Shanheng Zhao and Hwee Tou Ng. 2007. Identification and resolution of chinese zero pronouns: A machine learning approach. In *EMNLP-CoNLL*, volume 2007, pages 541–550.