3.3 Discourse Coherence Model

3.3.1 Deep Neural Network Architecture

There is no clearly result of how coherent a paragraph is, it's hard to evaluate. We build a deep neural network (DNN) and attempt to evaluate the discourse coherence of paragraphs, this model is used as a reward function in MCTS. The

[8] https://radimrehurek.com/gensim/

architecture of our model is shown in Figure 3.5. A paragraph consists of sentences, and a sentence consists of words. The word embedding we build in section 3.2 are used as an embedding layer to encode plain text to word vectors. In addition, we need a word sequence encoder which encodes words to a sentence, and a sentence encoder which encodes sentences to a paragraph. Since the contribution of each word in a sentence is not equal, and each sentence doesn't contribute equally either. Also, the importance of the same words or sentences in different context are different. The relations between words or sentences and their context are highly dependent. Therefore, we include two levels of attention mechanisms inspired by [87]. One in word level, and one in sentence level.

Word Level

Word Encoder First, we need to encode plain text $w_{il}, l \in [1, L]$ to word vectors x_{il} by an embedding layer W_e. Different words have different indexes in embedding matrix. Input sequence of word vectors to a BiLSTM (Bidirectional Long Short-Term Memory), and output the concatenation of forward and backward hidden states of BiLSTM.

$$\begin{aligned} x_{il} &= W_e w_{il}, l \in [1, L] \\ \overrightarrow{h}_{il} &= \overrightarrow{LSTM}(x_{il}), l \in [1, L] \\ \overleftarrow{h}_{il} &= \overleftarrow{LSTM}(x_{il}), l \in [L, 1] \end{aligned} \qquad \begin{aligned} &i : \text{ith sentence} \\ &l : \text{lth word} \end{aligned} \tag{3.6}$$

Word Attention Since not every word has the same importance in a sentence. The attention mechanism is introduced by [87]. u_{il} is the hidden representation of h_{il}. W_w and b_w is the weight and bias in word level. u_w is context vector in

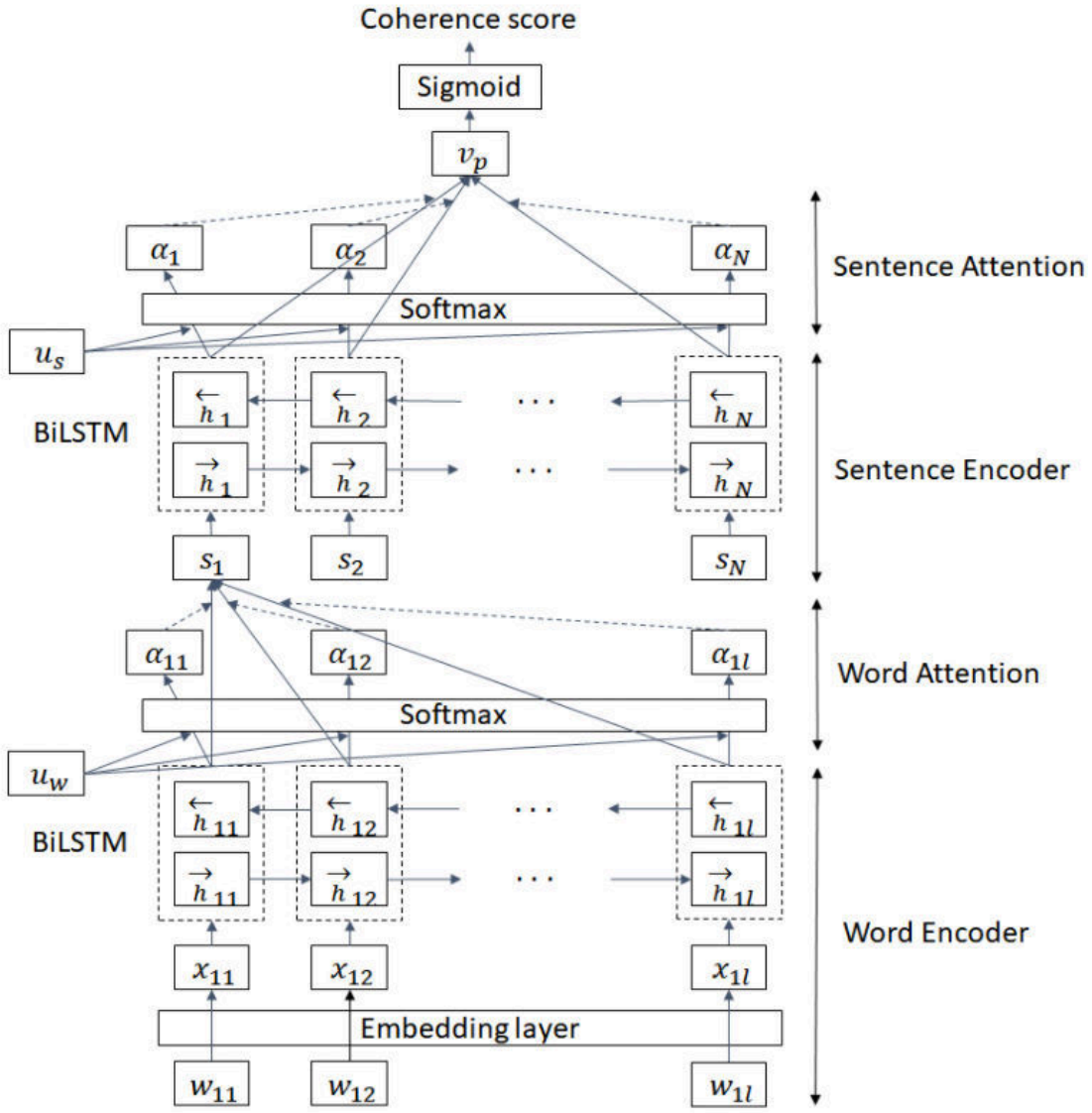

Figure 3.5: Discourse coherence model.

word level, it is randomly initialized and jointly trained. α_{il} is the normalized importance weight by softmax of the ith sentence and the lth word. Weighted sum importance weight α_{il} and the output of BiLSTM h_{il}

$$
\begin{aligned}
u_{il} &= tanh(W_w h_{il} + b_w) \\
\alpha_{il} &= \frac{exp(u_{il}^T u_w)}{\sum_t exp(u_{il}^T) u_w)} \\
s_i &= \sum_t \alpha_{il} h_{il}
\end{aligned}
\tag{3.7}
$$

Sentence Level

Sentence level is the same as word level. v_p is the vector representation of paragraph.

Sentence Encoder

$$\begin{aligned} \overrightarrow{h}_i &= \overrightarrow{LSTM}(s_i), i \in [1, N] \\ \overleftarrow{h}_i &= \overleftarrow{LSTM}(s_i), i \in [N, 1] \end{aligned} \tag{3.8}$$

Sentence Attention

$$\begin{aligned} u_i &= tanh(W_s h_i + b_s) \\ \alpha_i &= \frac{exp(u_i^T u_s)}{\sum_i exp(u_i^T) u_s)} \\ v_p &= \sum_i \alpha_i h_i \end{aligned} \tag{3.9}$$

Lastly, input paragraph vector v_p to a sigmoid activation function, and output a coherence score which in range 0 (incoherent) to 1 (coherent). We use layer normalization (LN) [88] to reduce Internal Covariate Shift (ICS). ICS is the covariate shift between hidden layers. The input of each layer is affected by previous layers, the following layers need to adapt new inputs. There are several methods to solve ICS problem.

- Smaller learning rate can limit the shift of distribution, but it spends longer training time.
- Proper parameters initialization, but it may spend more time to tune hyperparameters.
- Non-saturated activation function, e.g., ReLU.

- Batch normalization (BN) [89]. It calculates the mean (μ) and standard deviation (σ) of a mini-batch instead of a whole set, and $BN(z)$ is zero mean and unit variance. It prevents the training from getting stuck in the saturation regions (gradient close to 0), and lead to gradient vanishing.

$$BN(z) = \gamma \frac{z - \mu}{\sigma} + \beta \tag{3.10}$$

  z : activation function input
  γ : re-scale parameter
  β : re-shift parameter

- Layer normalization. BN is sensitive to mini-batch size. The performance of BN degrades when training on small batch size. The mean and variance are not accurate when the batch size is too small. Also, BN is difficult to apply to recurrent neural network (RNN) or LSTM. The length of sequence in RNN is not the same in different time steps. LN calculates mean and standard deviation of a sample instead of a mini-batch. There is no restriction with the batch size. LN has better performance in stacked RNN-like structure.

3.3.2 Training of the neural network

Corpora

The training corpora are shown in table 3.2. We use 71 popular boards in PTT. The total capacity is 4.82 GB after preprocessing.

	PTT	Chinatimes	LTN	Total
# of raw articles	5,794,074	2,696,580	957,146	9,447,800
# of segmented words	341,819,433	275,505,429	132,413,960	749,738,822
Capacity	6.79 GB	3.79 GB	1.78 GB	12.4 GB
Date	~2018/12	2009/10~2018/12	2005/1~2018/12	

Chinatimes: 中時新聞網
LTN: 自由時報

Table 3.2: Coherence model training corpora.

Corpus Preprocessing

Most of the corpus preprocessings are the same with word embedding in section 3.2. Besides those preprocessings, the corpus in NN model is a bit different from word embedding.

- Exclude articles with fixed format in PTT, e.g., 交易、廣告、自介、團購、情報,etc. The sentences in these types of articles are listed one by one. They may be a name, address, date or site. Each item is not coherent at all.

- Exclude upvote and downvote replies. Each reply is short, it may be a word, a single sentence or the number of sentences < 3 because of the limited width in one line. Adjacent replies may or may not related depending on the users reply to article or other users' replies. We exclude replies to ensure the coherence of sentences. Most of the pages in Wiki are introduction of people, locations or events. They aren't suitable for positive samples.

- Period and exclamation point represent the end of a paragraph. Comma represents the end of a sentence. Therefore, we split articles to paragraphs according to period and exclamation point.

- Exclude the number of sentences < 5, it would not be a paragraph if the

number of sentences is too few. Exclude paragraphs which include single word sentences, they provide little information to the paragraphs. Exclude single words if they account for more than 30% in a paragraph, more than 75% in one sentence or the number of continuous single character words exceeds 2 in a sentence. Single word may have ambiguity or it may be a fragment of a word after segmentation or it is an OOV. No matter in which situation, it is better to remove. Exclude the number of OOV words > 3.

- Remove the last name before words, which are occupations or titles, e.g., [陳 醫生] →[醫生]. The last name in words is not important.

- Combine separated proper noun and people's name. Most of the proper nouns and people's name are not in the self-defined dictionary. The text segmentation API can't detect those words properly. Combine these words to reduce the number of OOV words.

Training Data

The coherence of paragraph is stronger than article. It contains fewer words and the strength of relations between words is stronger than articles. Paragraph is local coherence and article is global coherence. Therefore, we use paragraphs as our training data instead of articles. The positive samples are the original paragraphs, and the negative samples are paragraphs with replacement of other concepts. Words in a sentence or a paragraph are closely related. Concepts replacement makes paragraph less coherent even a single replacement.

Because we use MCTS to select concepts randomly and find the suboptimal

paths. If the replaced concept is less related to other concepts in the paragraph, the model can distinguish the incoherence easily. Therefore, we imitate the pattern that MCTS simulates randomly to get negative samples. We first select a target concept (a word or words) and check whether it exists in ConceptNet or not. If it doesn't exist in ConceptNet, it won't be used as a target concept. MCTS doesn't select concepts randomly with arbitrary concepts in vocabulary table, it selects concepts randomly in ConceptNet. Therefore, we replace adjacent concepts (context) of target concept by connected concepts in ConceptNet. The positions of adjacent concepts can be different, we have three modes to replace concepts. The first one is replacing two words, one before target and one after target. The second one is replacing two words, both of words are before or after target. The last one is replacing one word, before or after target.

The concepts used to replace the context are not arbitrary connected concepts. Not each connected concept can replace the context, they may have different parts-of-speech. The paragraph becomes very incoherent if it is replaced by arbitrary concepts, and it's not compatible to our MCTS-based model. Therefore, the POS of replacing concept must be the same as replaced concept. We use CKIP tagger [9] to tag POS. (There is an another old version of CKIP tagger [10].) The word segmentation we use is Jieba, and the POS tagging we use CKIP tagger. After comparing the result of word segmentation, CKIP tagger segments words to smaller unit which means more segmented words. For example, CKIP tagger:[一 大 堆 垃 圾] and Jieba:[一 大 堆 垃 圾] (use same self-defined dictionary). As mentioned in data cleaning subsection (3.1.2), average word embedding of segmented words

[9] https://github.com/ckiplab/ckiptagger
[10] http://ckipsvr.iis.sinica.edu.tw/

can't fully represent or even differ from original meaning. Jieba segments words to fewer number of segmented words which meet our needs. There is no good or bad here, it depends on users' tasks, and the CKIP performs better in POS tagging. However, we use different model of word segmentation and POS tagging. Some of the words segmented by Jieba are segmented by CKIP tagger again to tag POS. Consequently, we have to recombine the segmented words back to their original words, e.g., 航空 (Na) 公司 (Nc) → 航空公司 (Nc). Additionally, CKIP tagger divides POS into lots of classes. We don't have to get detailed POS, hence we simplify the POS in appendix D.

The paragraph templates can ensure grammatically correct. Although we don't define which template slot corresponds to which specific POS and the template slots may have different types of concepts, the POS in specific slot is basically the same. For example, a compound relation "CapableOf-HasSubevent" has a template "__在__會__", "人在公園會散步" (someone walks in the park). The first slot is location which is adverbs of place, and the second slot is an action which is a verb.

The replacing concepts can't be the synonyms of replaced concept. Otherwise, there is no difference after replacement. We find synonyms of replaced concept by word embedding (top 30 most similar concepts), Cilin and relation "Synonym" in ConceptNet.

The two examples of negative samples are shown in Figure 3.6. The text in red is target concept, and the text in blue is replaced (in upper) and replacing (in lower) concept which connected to target concept in ConceptNet. We can see that only a word or two words are replaced, the paragraph becomes incoherent.

Figure 3.6: Examples of negative samples.

In order to ensure the negative samples are truly replaced by other concepts. We iterate the replacing process until the number of replaced sentences > 1. We also tag processed concepts so that they won't be used in following iterations to reduce computational cost.