Chinese Discourse Relation Recognition

Team name: “That Good Good Good” (This report can be shared)

Name	Student ID	Email registered
林祐萱	B03902055	linamy85@gmail.com (mailto:linamy85@gmail.com)
廖瑋中	B02902105	b02902105@ntu.edu.tw (mailto:b02902105@ntu.edu.tw)
鄭筱樺	B03902024	jke.evie@gmail.com (mailto:jke.evie@gmail.com)

Methodology

Segmentation

We used Jieba (https://github.com/fxsjy/jieba) as segmentation tool, and further translated the documents into simplified Chinese, since most of the word2vec resources only contains such language instead of traditional Chinese version.

Pretrained embedding

Since our training data is not large enough to train the embeddings well, some of our trials used the 64-dimensional pretrained embedding, which is from Ploylglot [2], a multi-lingual word embedding vector resource.

Trial 0: Traditional Statistics Method

We try to calculate probabilities of each relation for each words. The probability P_{wr} of word w belonging to relation r is

$$p_{wr} = \frac{\#\text{ of paragraphs word } w \text{ appears and the relation is } r}{\#\text{ of paragraphs word } w \text{ appears}}$$

$$P_{wr} = \frac{p_{wr}}{\#\text{ of paragraphs whose relation is } r}$$

For the 1st formula, we calculate each relation’s probability for word w. However, since some relations appear more often in training data, the probability tends to become higher than others. Therefore, in 2nd formula, we try to normalize it. In the 2 formulus, we only use the words which appear enought times. Words which do not appear enough time are considered untrustable and therefore discarded. Now we have the probabilities of each word belonging to each relation.

Subsequently, we have to determine the relation of each paragraphs, which contains 2 sentences in this task. We simply choose the word which has the highest probability and use its relation as the paragraph’s relation.

Then, we figure out that “expansion” tends to appear most frequently. Therefore, we try to set a threshold to decrease the number of expansion. The paragraph is determined as “expansion” only when its probability is higher than the threshold. Otherwise, choose the highest probability among 3 relations, except for “expansion.”

Also, we try to use polarity to determine “comparison.” If the polarities of 2 sentences are different, we determine the paragraph as “comparison.” Otherwise, use the above method to determine the relation.

Trial 1: SVM

First, similar to trial 0, we calculate the probabilities P_{wr} of each relation for each word. Then, we can determine the relation of each word.

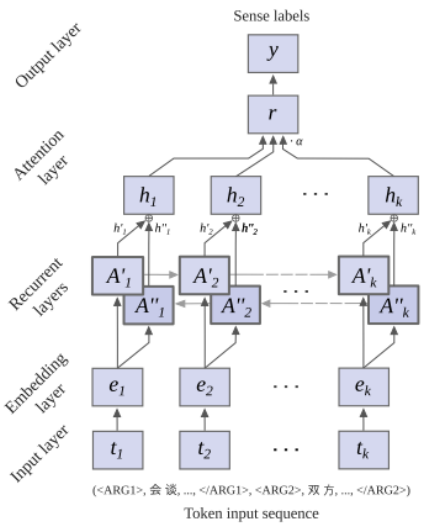
Secondly, we choose the words whose relation probability are higher than a threshold we set. The chosen words form the feature space. Say there are N training data and k features (words), we embed each training data on the feature space, and therefore we have a matrix $X \in R^{N \times k}$. The elements in X are calculated by TF or TF-IDF, where

$$TF_{ij} = \frac{\#\text{ of word } j \text{ appearing in paragraph } i}{\#\text{ of words in paragraph } i} \quad i \in [1, N], j \in [1, k]$$

$$TFIDF_{ij} = TF_{ij} \times \log \frac{N}{\#\text{ of paragraphs containing word } j} \quad i \in [1, N], j \in [1, k]$$

After having $X \in R^{N \times k}$ and $Y \in N^N$, we can use SVM to train it.

Trial 2: A Recurrent Neural Model with Attention [1]



This model is introduced by researchers from Finland and Germany. They used attention-based Bi-LSTM for solving Chinese implicit discourse relations, and modeled the argument pairs as a joint sequence. Their method achieves state-of-the art performance on the CoNLL 2016 CDTB sets. Specially, There are several properties that worth mentioned:

- 300-dimensional Chinese Gigaword pretrained embedding vectors
- Each pair of sentence is transformed into a sequence, for example, (“天氣真好”, “我們去吃冰”) will become “<Arg1>天氣真好</Arg1><Arg2>我們去吃冰</Arg2>”, and further use

zero-padding to fixed length.

- Partial argument sampling is used. That is, a data point (a1, a2, y) is expanded into {(a1, a2, y), (a1, a2, y), (a1, y), (a2, y)}

However, this model does not perform well on this assignment. Given several possible reasons:

- Too many parameters are trained in Bi-LSTM, however, there are only 6639 training data for us to build the model. So we might be easily overfitting to the training data, especially when the distribution of training and testing data shares big difference.
- This model is targetting implicit discourse relations, meaning that it might miss some explicit clues in out dataset.

Trial 3: CNN

- 2 convolutional layers: pretrained word embedding + Conv1D * 2 + MaxPooling1D + Dropout1 + (Dense) + Dropout2 + Dense
- 3 convolutional layers: pretrained word embedding + Conv1D * 2 + MaxPooling1D + Dropout1 + Conv1D + MaxPooling1D + (Dense) + Dropout2 + Dense

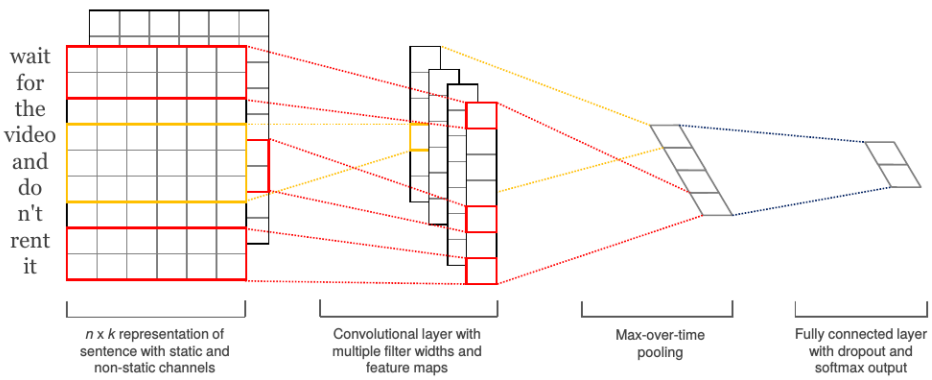
Trial 4: CNN + LSTM

The general structure is:
pretrained word embedding + zero_padding
➡️ Conv1D + MaxPooling1D + LSTM + {dense + dropout} x n + dense

And n is ranging from 0 to 2.

Trial 5: CNN + Global max-over-time pooling

- First, since training data is unbalanced, we adjust the distribution of training data to 3:3:2:2 by duplicating data that labeled as Contingency, Comparison, or Temporal
- We use the pretrained embeddings as described before.
- Our CNN model is according to Convolutional Neural Networks for Sentence Classification [3]
  - 1 convolution layer
  - Global max-over-time pooling
  - 1 fully connected layer with dropout and softmax output



- We try this CNN models with different number of conv layers, different number of filters, different filter window sizes, different dropout probability, and different number of dense layers to get higher performance.

Experiments

Trial 0: Traditional Statistics Method

- frequency threshold: Only words which appear at least this number are used.
- probability threshold: We calculate each word's probability for each relation at first. Only words whose highest probability is higher than this number are used.
- polarity usage: Use polarity to predict "comparison," and use original method to predict others.

frequency threshold	probability threshold	polarity Usage	public accuracy	private accuracy
10	-	-	0.57800	0.57800
20	-	-	0.60400	0.58200
50	-	-	0.57200	0.56600
20	0.4	-	0.60000	0.57200
20	0.5	-	0.61200	0.60200
20	0.5	Yes	0.50800	0.51400

Trial 1: SVM

TF/TF-IDF	probability threshold	public accuracy	private accuracy
TF	0.3	0.52200	0.49600
TF-IDF	-	0.60800	0.57400
TF-IDF	0.3	0.60600	0.58400
TF-IDF	0.4	0.59200	0.56800
TF-IDF	0.5	0.58600	0.56200

- TF/TF-IDF: Use TF or TF-IDF to calculate the values of features.
- probability threshold: The same as trial 0.

Trial 2: A Recurrent Neural Model with Attention

The hyper parameters are the same as the paper used.

epoch	public accu	private accu
10	0.52400	0.49000
20	0.49800	0.47800

Trial 3: CNN

- batch size = 128, learning rate = 0.001

filter dim	conv layer#	drop prob1	drop prob2	add 1 dense	epoch	cv_acc	public/private_acc
32	2	0.25	0.50	Y:16	70	0.6441	-
32	2	0.25	0.50	N:xx	70	0.6568	-
24	2	0.25	0.50	N:xx	70	0.6690	-
24	2	0.50	0.50	N:xx	70	0.6669	0.60800/0.63600
32	2	0.50	0.50	N:xx	120	0.6462	-
32	3	0.50	0.50	N:xx	70	0.6462	-

Trial 4: CNN + LSTM

- batch size = 64
- learning rate = 0.001
- filter dimension = 32 (the number output of filters in the convolution)

LSTM units	epoch	dense n	public	private
100	30	0	0.60800	0.60400
100	50	0	0.62000	0.57800
100	100	0	0.61000	0.59200
50	60	0	0.61600	0.58200
32	60	1	0.63200	0.61000
32	80	1	0.60200	0.62800
32	80	2	0.58600	0.61200

Trial 5: CNN + Global max-over-time pooling

- batch size = 128, learning rate = 0.001

filter#	filter size	conv layer#	drop_prob	add 1 more dense	epoch	cv_acc	public / private
24	3	1	0.50	N:xx	70	0.6759	-
32	3	1	0.50	N:xx	70	0.6753	-
24	3	1	0.50	N:xx	100	0.6779	0.6600/0.6400
24	3,3,4	1	0.50	N:xx	50	0.6871	0.6560/0.6320
24	3,4,5	1	0.50	N:xx	50	0.6803	0.6480/0.6380
32	3,4,5	1	0.50	N:xx	50	0.6730	-
24	3,4,5	1	0.50	Y:96	50	0.6425	-
100	3,3,3	1	0.50	N:xx	50	0.6873	0.6420/0.6340

Discussions

- The traditional statistics method can reach a not bad result.
- SVM does not perform well in this task. Even traditional statistics method performs better than SVM.
- CNN with 2 convolutinal layers performs better than CNN with 1 or 3 convolutional layers
- LSTM can captures information in sequence nicely, however it contains too many parameters to train
    - might be easy for it to overfit the training data
- “CNN + Global max-over-time pooling” performs better than the traditional CNN

Reference

[1] A Recurrent Neural Model with Attention for the Recognition of Chinese Implicit Discourse Relations (https://arxiv.org/abs/1704.08092)
[2] Building Sentiment Lexicons for All Major Languages (http://aclweb.org/anthology/P14-2063)
[3] Convolutional Neural Networks for Sentence Classification (https://arxiv.org/pdf/1408.5882.pdf)