Cosi 134 - Project 2: Neural Discourse Relation Classification

Due: October 20, 2020

The task for Project 2 is discourse relation classification, the same as in Project 1.

1. What is Discourse Relation Classification?

A typical text consists of sentences that are glued together in a systematic way to form a coherent discourse. Discourse Relation Classification is the task of classifying the discourse relation between two adjacent or non-adjacent discourse units. This project is motivated by the CoNLL 2016 Shared Task: Shallow Discourse Parsing. More information can be found here: http://www.cs.brandeis.edu/~clp/conll16st/index.html (http://www.cs.brandeis.edu/~clp/conll16st/index.html)

Assignment Project Exam Help

1.1 Discourse Analysis in the Penn Discourse Treebank (PDTB)

https://powcoder.com

We use the Penn Discourse Treebank as the data set. The PDTB annotates a text with a set of discourse relations. A discourse relation is composed of:

Add WeChat powcoder

- a discourse connective, which can be a coordinating conjunction (e.g., "and", "but"), subordinating conjunction (e.g. "if", "because"), or a discourse adverbial (e.g., "however", "also"). In an implicit discousre relaiton, a discourse connective is omitted;
- two Arguments of the discourse connective, Arg1 and Arg2, which are typically text spans the size of clauses or sentences;
- the sense of the discourse connective, which characterizes the nature of the relationship between the two arguments of the connective (e.g., contrast, instantiation, temporal precedence).

1.2 Examples discourse relations from the PDTB

Here is a paragraph taken from the document wsj 1000 in the PDTB. A discourse relation classifier will output a discourse relation for each pair of arguments, as visualized below. Arg1 is shown in red, Arg2 is shown in blue, and the discourse connective is underlined.

Explicit Discourse relations:

According to Lawrence Eckenfelder, a securities industry analyst at Prudential-Bache Securities Inc., "Kemper is the first firm to make a major statement with program trading." He added that "having just one firm do this isn't going to mean a hill of beans. But this prompts others to consider the same thing, then it may become much more important."

The discourse connective is "but", and the sense is Comparison.Concession.

Implicit Discourse Relations

According to Lawrence Eckenfelder, a securities industry analyst at Prudential-Bache Securities Inc., "Kemper is the first firm to make a major statement with program trading." He added that "having just one firm do this isn't going to mean a hill of beans. But if this prompts others to consider the same thing, then it may become much more important."

Assignment Project Exam Help

The omitted discourse connective is "however", and the sense is Comparison.Contrast.

https://powcoder.com

2 Useful information about how to process the PDTB data set

Add WeChat powcoder

- It is located in `data/pdtb` in the starter package
- PDTB comes with pre-defined dataset splits of training, dev and test sets
- The training data are from Sections 2-21, the development set is from Section 22, and Section 23 is used as the evaluation/test set.
- By default, the features include separate unigrams for Arg1, Arg2 and Connectives. Use these to output a single `feature vector`.

2.2 Format of a discourse relation instance in json:

Each discourse relation instance has the following components:

- Arg1 : the text span of Arg1 of the relation
- Arg2 : the text span of Arg2 of the relation
- Connective : the text span of the connective of the relation. This could be optional.
- DocID : document id where the relation is in.
- ID : the relation id, which is unique across training, dev, and test sets.
- Sense : the sense of the relation, which is what your model needs to predict.
- Type : the type of relation (Explicit, Implicit, Entrel, AltLex, or NoRel)

The text span is in the same format for Arg1, Arg2, and Connective. A text span has the following fields:

- CharacterSpanList : the list of character offsets (beginning, end) in the raw untokenized data file.
- RawText : the raw untokenized text of the span
- TokenList : the list of the addresses of the tokens in the form of (character offset begin, character offset end, token offset within the document, sentence offset, token offset within the sentence)

Assignment Project Exam Help

In [2]:
```
import json
pdtb_file = open("../data/pdtb/train.json", encoding='utf-8')
relations = [json.loads(x) for x in pdtb_file]
example_relation = relations[10]
example_relation
```

https://powcoder.com

Add WeChat powcoder

Out[2]:
```
{'Arg1': {'CharacterSpanList': [[9, 50]],
  'RawText': 'Solo woodwind players have to be creative',
  'TokenList': [[9, 13, 0, 0, 0],
   [14, 22, 1, 0, 1],
   [23, 30, 2, 0, 2],
   [31, 35, 3, 0, 3],
   [36, 38, 4, 0, 4],
   [39, 41, 5, 0, 5],
   [42, 50, 6, 0, 6]]},
 'Arg2': {'CharacterSpanList': [[54, 77]],
  'RawText': 'they want to work a lot',
  'TokenList': [[54, 58, 8, 0, 8],
   [59, 63, 9, 0, 9],
   [64, 66, 10, 0, 10],
   [67, 71, 11, 0, 11],
   [72, 73, 12, 0, 12],
   [74, 77, 13, 0, 13]]},
 'Connective': {'CharacterSpanList': [[51, 53]],
  'RawText': 'if',
  'TokenList': [[51, 53, 7, 0, 7]]},
 'DocID': 'wsj_0207',
 'ID': 3183,
 'Sense': ['Contingency.Condition'],
 'Type': 'Explicit'}
```

2.1 The list of discourse relation labels (senses):

- The complete list of labels are given to you as `SENSES` in `corpus.py` See below for details.

In [3]:
```
SENSES = [
    'Temporal',
    'Temporal.Asynchronous',
    'Temporal.Asynchronous.Precedence',
    'Temporal.Asynchronous.Succession',
    'Temporal.Synchrony',
    'Contingency',
    'Contingency.Cause',
    'Contingency.Cause.Reason',
    'Contingency.Cause.Result',
    'Contingency.Condition',
    'Comparison',
    'Comparison.Contrast',
    'Comparison.Concession',
    'Expansion',
    'Expansion.Conjunction',
    'Expansion.Instantiation',
    'Expansion.Restatement',
    'Expansion.Alternative',
    'Expansion.Alternative.Chosen alternative', # `alternative` can be s
afely ignored
    'Expansion.Exception',
    'EntRel'
]
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

In [4]:
```
def to_level(sense: str, level: int = 2) -> str:
    """converts a sense in string to a desired level

    There are 3 sense levels in PDTB:
    Level 1 senses are the single-word senses like `Temporal` and `Conti
ngency`.
    Level 2 senses add an additional sub-level sense on top of Level 1 s
enses,
      as in `Expansion.Exception`
    Level 3 senses adds yet another sub-level sense, as in
      `Temporal.Asynchronous.Precedence`.

    This function is used to ensure that all senses do not exceed the de
sired
    sense level provided as the argument `level`. For example,

    >>> to_level('Expansion.Restatement', level=1)
    'Expansion'
    >>> to_level('Temporal.Asynchronous.Succession', level=2)
    'Temporal.Asynchronous'

    When the input sense has a lower sense level than the desired sense
 level,
    this function will retain the original sense string. For example,

    >>> to_level('Expansion', level=2)
    'Expansion'
    >>> to_level('Comparison.Contrast', level=3)
    'Comparison.Contrast'

    Args:
    sense (str): a sense as given in a `relations.json` file
    level (int): a desired sense level

    Returns:
    str: a sense below or at the desired sense level
    """
    s_split = sense.split(".")
    s_join = ".".join(s_split[:level])
    return s_join
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

3 Project Requirements:

The goal of this project is to implement a model to classify discourse relations, the same task as in Project 1. This time though, you are asked to implement neural network models in PyTorch that allow you to perform the following experiments:

- Experiment 1. Implement Logistic Regression as a two-layer neural network, and tune the hyperparameters (learning rate, optimization methods, etc.) to obtain the best results you could. Compare the results with what you have obtained in PA1. The results should be similar if they are implemented correctly.
- Experiment 2: Adding a hidden layer to the Logistic Regression model to make it a deep feedforward network (also known as multi-layer perceptron). Optimize the size of the hidden layer to get the best results. You can optionally experiment with multiple hidden layers. Does adding hidden layers help improve your classification accuracy?
- Experiment 3: Experiment with using pretrained word embeddings such as GLOVE in your nueural network model. Experiment with different word embedding sizes. Start with a smaller embedding size (e.g., 50) to make sure your network is not too slow, and gradually increase moves to larger embedding sizes. Does using pretrained word embeddings help with your classification accuracy? The GLOVE embeddings can be Downloaded here: https://nlp.stanford.edu/projects/glove (https://nlp.stanford.edu/projects/glove)
- Experiment 4 (optional, for extra credit): Experiment with using a convolutional network for the discourse classification problem. When switching to convolutional networks, you may need to change the input representation of your network to a sequence.

Assignment Project Exam Help

When you work on this project, you need to perform the following steps:

https://powcoder.com

1. Implement data processing code that reads in training/develop/test data in the format as described in Section 2, and outputs sense classifications in the format as described in Section 3.1.
2. Evaluate your system on explicit, implicit, and all relations, using the evaluation script described in Section 3.2.
3. Write a project report:
    - Describe the code structure of your project, and provide instructions as to how to run/test your system.
    - Describe your input representation, neural network architecture, and the hyperparameters that you have used in each of the experiment settings outlined above. Report your experimental results and analyze of what has worked and what hasn't.
    - Limit your the length of your report to 5 pages.
    - Grading will be based on the correctness of your implementation, the performance of your models, and the clarity of your report.

Add WeChat powcoder

3.1 What should the system output look like?

The system output must be in json format. It is very similar to the training set except for the TokenList field. The TokenList field is now a list of document level token indices. Even if the relation is not explicit, the connective field must still be there, and its TokenList must be an empty list. You may however add whatever field into json to help yourself debug or develop the system. Below is an example of a relation given by the system.

```
Out[13]: {'Arg1': {'TokenList': [275, 276, 277, 278, 279, 280, 281, 282]},
          'Arg2': {'TokenList': [329, 330, 331, 332, 333, 334, 335, 336, 337]},
          'Connective': {'TokenList': []},
          'DocID': 'wsj_1000',
          'Sense': ['Expansion.Conjunction'],
          'Type': 'Implicit'}
```

3.2 Evaluation

Precision, recall, and F of the discourse relation classification are used as the evaluation metrics. To evaluate, run: python scorer.py /relations.json /output.json

The scorer should not be modified. If you run into issues, contact me or the TAs

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder