Convert Stanford CoreNLP training data to Dataturks NER JSON output

DataTurks: Data Annotations Made Super Easy
Jul 19, 2018 · 2 min read

Generate a convenient JSON format for shifting from Stanford CoreNLP to SpaCy or other NER modules

Shameless plugin: We are a data annotation platform to make it super easy for you to build ML datasets. Just upload data, invite your team and build datasets super quick. ***Check us out!***

. . .

A simple script to create dataset in Dataturks output format for NER, which can be very conveniently converted to any other training formats for NER API's like SpaCy, etc.

Dataturks NER output is very close to the format used by Spacy, just that Spacy used Python tuples which are not supported by JSON standard. So if you have your data in Stanford CoreNLP format and wish to check out the performance of SpaCy NER mode on your data, you can easily convert Stanford NER data to Dataturks output JSON format, which can be subsequently processed to SpaCy format.

We assume every word is assigned only one entity label in the script below.

Input File : Stanford CoreNLP NER Training Data .tsv File as shown below :

Search this file…

I	0
bought	0
a	0
hp	Brand
spectre	ModelName
x360	ModelName
.	0

stanford_data.tsv hosted with ♡ by GitHub view raw

NOTE : Each training sample in Stanford NER format are separated by a newline.

Here's a script to convert Stanford CoreNLP format to Dataturks JSON format :

```
import json
import logging
import sys
##############################################  NOTE  ##############################################
#
#           Creates NER training data in Dataturks format from Stanford Core NLP format.
#           NOTE :This Function assumes that each token in the input fie is assigned only one l
#           Outputs the dataturks training data json format which can be used for conversion to
#
#########################################################################################################
def Stanford_to_dataturks_format(path_to_stanford_ner_data,output_json_file_path,unknown_label)
    try:
            f=open(path_to_stanford_ner_data,'r')#input file in Stanford NER format
            fp=open(output_json_file_path, 'w')#output format in Dtaturks JSON File
            data_dict={}
            annotations=[]
            label_dict={}
            s=''
            start=0
            for line in f:
                if line!='\n':
                    word,entity=line.split('\t')
                    s+=word+" "
                    entity=entity[:len(entity)-1]
                    if entity!=unknown_label:
                        d={}
                        d['text']=word
                        #dataturks indices are both inclusive [start, end]
                        d['start']=start
                        d['end']=start+len(word)-1
                        try:
```

```
                            label_dict[entity].append(d)
                        except:
                            label_dict[entity]=[]
                            label_dict[entity].append(d)
                    #Increment start index and accomodate the position of an added space as wel
                    start+=len(word)+1
                else:
                    data_dict['content']=s
                    s=''
                    #create groups of points having same text
                    label_list=[]
                    for ents in list(label_dict.keys()):
                        for i in range(len(label_dict[ents])):
                            if(label_dict[ents][i]['text']!=''):
                                l=[ents,label_dict[ents][i]]
                                for j in range(i+1,len(label_dict[ents])):
                                    if(label_dict[ents][i]['text']==label_dict[ents][j]['text']
                                      di={}
                                      di['start']=label_dict[ents][j]['start']
                                      di['end']=label_dict[ents][j]['end']
                                      di['text']=label_dict[ents][i]['text']
                                      l.append(di)
                                      label_dict[ents][j]['text']=''
                                label_list.append(l)

                    for entities in label_list:
                       label={}
                       label['label']=[entities[0]]
                       label['points']=entities[1:]
                       annotations.append(label)
                    data_dict['annotation']=annotations
                    annotations=[]
                    #one json object for a training sample written to output file
                    json.dump(data_dict, fp)
                    fp.write('\n')
                    data_dict={}
                    start=0
                    label_dict={}
        except Exception as e:
            logging.exception("Unable to process file" + "\n" + "error = " + str(e))
            return None


if(len(sys.argv)<2):
    print("Please provide input and output data path and the label used for tagging as unknown
    exit(0)
Stanford_to_dataturks_format(sys.argv[1],sys.argv[2],sys.argv[3])
```

The output Dataturks data would look something like he following :


```
{
   "content":"I bought a hp spectre x360 . ",
   "annotation":[
      {
         "points":[
            {
               "start":11,
               "end":12,
               "text":"hp"
            }
         ],
         "label":[
            "Brand"
         ]
      },
      {
         "points":[
            {
               "start":14,
               "end":20,
               "text":"spectre"
            }
         ],
         "label":[
            "ModelName"
         ]
      },
      {
         "points":[
            {
               "start":22,
               "end":25,
               "text":"x360"
            }
         ],
         "label":[
            "ModelName"
         ]
      }
   ]
}
```


NOTE : The above script is capable of handling only such situations wherein you do not have multi-word entities or consecutive words with same labels as shown below :

Search this file…

Have	0
You	0
repaired	0
the	0
home	Category
theater	Category
system	0
?	0

multi_word_entities.tsv hosted with ♡ by GitHub view raw

In such a case, it is wise to have ‘home theater’ as a span in the annotations.

In that case, it is possible to combine the consecutive words with same label with an added script in the previous code as follows :

```
import json
import logging
import sys
##############################################  NOTE  ##############################################
#
#           Creates NER training data in Dataturks format from Stanford Core NLP format.
#           NOTE :This Function assumes that each token in the input fie is assigned only one l
#           Outputs the dataturks training data json format which can be used for conversion to
#
#####################################################################################################
def Stanford_to_dataturks_format(path_to_stanford_ner_data,output_json_file_path,unknown_label):
    try:
            f=open(path_to_stanford_ner_data,'r')#input file in Stanford NER format
            fp=open(output_json_file_path, 'w')#output format in Dtaturks JSON File
            data_dict={}
            annotations=[]
            label_dict={}
            s=''
            start=0
            for line in f:
                if line!='\n':
                    word,entity=line.split('\t')
                    s+=word+" "
                    entity=entity[:len(entity)-1]
                    if entity!=unknown_label:
                        d={}
                        d['text']=word
                        #dataturks indices are both inclusive [start, end]
                        d['start']=start
```

```
                    d['start']=start
                    d['end']=start+len(word)-1
                    try:
                        label_dict[entity].append(d)
                    except:
                        label_dict[entity]=[]
                        label_dict[entity].append(d)
                #Increment start index and accomodate the position of an added space as wel
                start+=len(word)+1
            else:
                data_dict['content']=s
                s=''

                #combine consecutive words with same labels
                for entities in list(label_dict.keys()):
                    for i in range(len(label_dict[entities])-1):
                        if(len(list(label_dict[entities][i].keys()))!=0):
                            c=i+1
                            while(c<len(label_dict[entities]) and label_dict[entities][i]['
                                label_dict[entities][i]['end']=label_dict[entities][c]['enc
                                label_dict[entities][i]['text']+=" "+label_dict[entities][c
                                label_dict[entities][c]={}
                                c+=1
                    label_dict[entities]=list(filter(lambda a: len(list(a.keys()))!=0, labe
                #create groups of points having same text
                label_list=[]
                for ents in list(label_dict.keys()):
                    for i in range(len(label_dict[ents])):
                        if(label_dict[ents][i]['text']!=''):
                            l=[ents,label_dict[ents][i]]
                            for j in range(i+1,len(label_dict[ents])):
                                if(label_dict[ents][i]['text']==label_dict[ents][j]['text']
                                  di={}
                                  di['start']=label_dict[ents][j]['start']
                                  di['end']=label_dict[ents][j]['end']
                                  di['text']=label_dict[ents][i]['text']
                                  l.append(di)
                                  label_dict[ents][j]['text']=''
                            label_list.append(l)

                for entities in label_list:
                   label={}
                   label['label']=[entities[0]]
                   label['points']=entities[1:]
                   annotations.append(label)
                data_dict['annotation']=annotations
                annotations=[]
                #one json object for a training sample written to output file
```

```
                    json.dump(data_dict, fp)
                    fp.write('\n')
                    data_dict={}
                    start=0
                    label_dict={}
        except Exception as e:
            logging.exception("Unable to process file" + "\n" + "error = " + str(e))
            return None


if(len(sys.argv)<2):
    print("Please provide input and output data path and the label used for tagging as unknown
    exit(0)
```

follow the following tutorial :

Creates NER training data in Spacy format from JSON downloaded from Dataturks.

Dataturks NER output is very close to the format used by Spacy, just that Spacy used Python tuples which are not...

dataturks.com

If you have any queries or suggestions, I would love to hear about it. Please write to me at abhishek.narayanan@dataturks.com.

Shameless plugin: We are a data annotation platform to make it super easy for you to build ML datasets. Just upload data, invite your team and build datasets super quick. ***Check us out!***

Best online platform for your ML data annotation needs.

Just upload your data, invite your team members and start tagging. The best way to tag training/evaluation data for...

www.dataturks.com

NLP Named Entity Recognition