NLP Course - Assignment 4

Otmazgin, Shon
shon711@gmail.com

Rassin, Royi
isroei5700@gmail.com

January 20th, 2021

Introduction

Relation Extraction is the task of extracting semantic relationships from text, which typically occurs between two or more entities. In this work, we center our attention on the 'Work-For' relation, and give a full account of the implementation behind our hybrid system; a Machine Learning model enhanced by rules, and the motivation behind our decisions.

Dataset

The dataset is made from two parts; a corpus, and an annotations-file (the gold-standard). The corpus consists of rows with a sentence and an id, similarly for the annotations-file, but with the addition of the relation between two entities in that sentence. Thus, sentences which contain more than one relation appear multiple times in the annotations-file and the number of times an id appears in the file can be interpreted as the number of relations hidden in a sentence.

A row in the corpus is of the structure:
sentence_id<TAB>sentence_text

A row in the annotations-file is of the structure:
sentence_id<TAB>ent1<TAB>rel<TAB>ent2<TAB>(sentence_text)

The dataset is split to two parts; train and dev.

The train corpus and annotations contain 354 sentences and 546 annotations respectively, 109 of them are a *Work-For* relation.

The dev corpus and annotations contain 340 sentences and 536 annotations respectively, 106 of them are a *Work-For* relation.

Model

Model Architecture

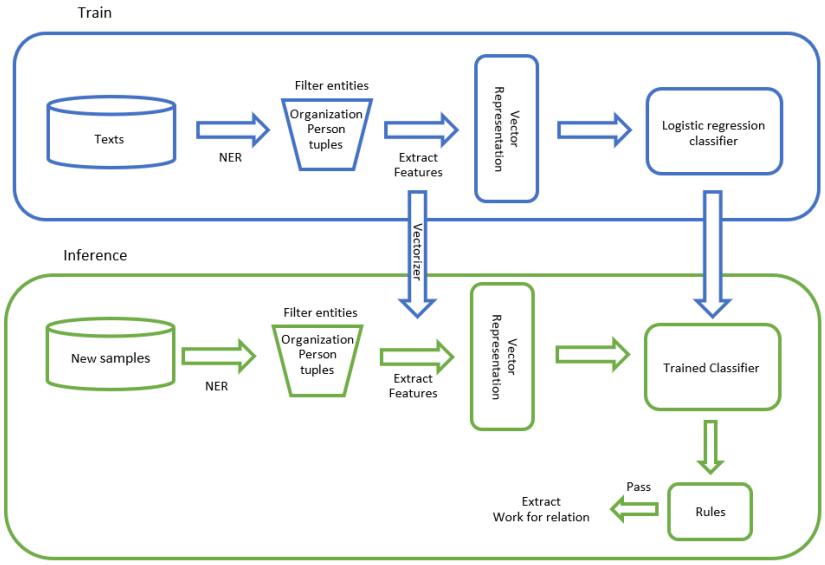

Figure 1: During training, the sentences go through the NER classifier and are broken down to PERSON-ORGANIZATION tuples. Then, features are built around the tuples and the vector-representation of each one is sent to a Log-Reg model. During inference, sentences are broken down to tuples too, and learned mappings from the training phase are used to build the inference-data features. Finally, a trained classifier makes predictions and we filter them with rules.

Name Entity Recognition

The first step to extract who works where is to recognize the entities from the sentences. Since the Work-For relation is between 'Person' and 'Organization', these are the only ones which are relevant to our task. We tested three NER models: spaCy's large-OntoNotes-model, Stanza's OntoNotes-model, and Stanza's [1] CoNLL03 model. spaCy provides an easy and well documented API with fast performance. On the other hand, Stanza achieves better F1 NER score on OntoNotes-5 (88.8 vs. 86.4) but with slower performance and not as easy-to-use API. Furthermore, Stanza provides an interesting advantage; their CoNLL03 model only has four types (OntoNotes-5 has 18 types): PERSON, ORGANIZATION, LOCATION, and MISC. As a result, we believe that the model's emphasis was on the first three entities, while the latter was trained to

[1] Stanza (Stanford NLP)

be everything else. Since we look for relations between PERSON and ORGANIZATION, this provided a clear benefit to us (side note: Stanza's CoNLL03 scored 92.1 for F1). Eventually, we settled on a library that wraps Stanza with spaCy's API; spacy-stanza.

Feature Extraction

The slides from the Relation Extraction lecture inspired us to come up with the following features:

- One-hot of all the PERSON and ORGANIZATION entities (unigram and bigram versions)
- One-hot of all the words after the most-left entity (unigram and bigram versions)
- One-hot of all the words after the most-right entity (unigram and bigram versions)
- One-hot of all the words between the PERSON and ORGANIZATION entities (unigram and bigram versions)
- One-hot of all the dependency paths between PERSON and ORGANIZATION
- The average GloVe vector of the lemmas (lower-cased) in the PERSON entity
- The average GloVe vector of the lemmas (lower-cased) in the ORGANIZATION entity.

Example: **W. Dale Nelson** covers the White House for The **Associated Press** .

- W., Dale, Nelson, Associated, Press, W. Dale, Dale Nelson, Nelson Associated, Associated Press (all of these words are entries in the one-hot and they get a value of one in it)
- <START> (no word precedes W. Dale Nelson)
- <END> (no word succeeds Associated Press)
- covers, the, White, House, for, The, covers the, the White, White House, House for, for The
- PER <- nsubj <- cover -> obl -> ORG (this is an entry in the one-hot dependency vector)
- average(GloVe(w.) + GloVe(dale) + GloVe(nelson))
- average(GloVe(associated) + GloVe(press))

We used the 6B-300d version of GloVe for the vectors.

Vector representation

In order to feed our classifier with the extracted features, we need to transform the categorical features to numerical ones. We selected Scikit-Learn's CountVectorizer to apply one-hot encoding to transform words. It provides a convenient interface for using n-gram, stop words, and especially, converting unseen features at test set to zero. In addition, we used GloVe-6B-300d-vectors to embed the two entities (PERSON and ORGANIZATION). Finally we end up with two matrices; one for the categorical features, and the second for the entities' embedding. The rows represent the tuple, while the columns represent the features. Consequently, both matrices have the same number of rows while the number of columns is different. We concatenate horizontally the two matrices, resulting in the shape: (265, 3599).

Classifier

The classifier expects a feature-vector as input and is trained on the train set to predict whether the vector signifies a Work-For relation or not. We trained a LogisticRegression classifier, and opted for the hyper-parameters: 'liblinear' as our solver, since the dataset is small, and that is scikit-learn's recommendation in that regard. Furthermore, we selected the class-weight parameter 'balanced' to correct some of the discrepancy between the class-frequencies, and 'C', the inverse of regularization-strength is 0.01 (the smaller value, the stronger regularization). We have also experimented with Gaussian Naive Bayes and SVM, but the results were incomparable.

Training

As previously mentioned, we build features from ORGANIZATION and PERSON tuples, since these are the tuples that are potentially Work-For related. After creating the features, we go through the training-set, and check if we have a 'soft-match'; if our observation contains the annotation or vice-versa. For instance, the observation=(W. Dale Nelson, Associated Press) and the annotation=(W. Dale Nelson, The Associated Press) is considered a soft-match, because 'The' is missing from the observation. Out of the 265 PERSON-ORGANIZATION tuples that were extracted from the corpus, 101 were soft Work-For related. The motivation behind this type of matching is to create a more balanced training set, and in return, create a model that generalizes better.

Rules

On top of our classifier results we implemented rule based system to improve the software's precision. Each observation was checked for the following rules:

1. Retired - if the person worked at the organization in the past and retired, our classifier didn't learn it from the context features. For Example "...said

Tidwell , a retired CIA officer... " the classifier detected Tidwell Work-For CIA. Our rule searches for the word 'retired' in the words between the person-org entities, if it exist, we check for a connection between the retired token to the organization-entity and if it does, we do not consider this a Work-For relation.

2. Possessive 's - if we have a possessive 's between an ORGANIZATION and PERSON, it makes sense that there is a relation between the two of them. We check if the possessive 's is a single index to the right of the ORGANIZATION entity. If so, we check if the ORGANIZATION entity exists in the location lexicon. If so, we ignore this prediction. From our experience, the NER classifier mixes up between locations and organization (especially when big cities are mentioned) and this helps us detect such cases and ignore them. Two examples of sentences that our rule solves:

   Location with 's -> our classifier think that's an Organization

   sent1147 Pittsburgh 's superintendent , Richard C. Wallace , who arrived in 1980 as an outsider to a city enmeshed in a bitter dispute over school desegregation , helped move the school system beyond that dispute and establish it as a national model of reform .

   sent1145 Rochester 's Peter McWalters , named in 1986 , signaled willingness to form a working partnership with teachers .

Error analysis

For both types of errors we randomly sampled five Recall and five Precision errors, came up with a hypothesis to treat these errors, and tested it. Some of the analysis led us to add the aforementioned rules, and some of it made us understand that our model is limited by its NER classifier.

Precision

Precision errors occur when we identify a relation between two entities that are not Work-For related, or when a relation exists, but is not in the annotations-file. Some sentences have a Work-For relation that existed in the past, while the classifier deems them to be current (e.g., a retired employee), some are between a PERSON and a LOCATION, which is not relevant to Work-For, and some are mislabeled by the classifier as several entities instead of one; "Harvard University 's John F. Kennedy School of Government" was broken down to two entities, Harvard University, and John F. Kennedy. This can lead to a precision mistake if one exists in the annotation, and the other does not.

In this particular case, we do not have this problem, and our classifier detects both entities and their PERSON counterpart's relation. However, there are other instances where only part of the entity exists in the annotations-file (only Harvard University, for instance), or that the NER classifier detects only part of the entity, but the annotations-file has a more complete one (which we deal

with soft-matching, explained in the Training section) - this situation can also lead to a Recall error.

The most problematic type of error we have encountered is two organizations that share similar context, and one person that works for one of them. In these cases, our classifier seems to predict both tuples to have a Work-For relation, while only one is actually correct. For instance: "**Ralph K. Winter** of the **2nd U.S. Circuit Court of Appeals** in New York and **Kenneth Starr** of the **U.S. Circuit Court of Appeals**", Ralph works for the 2nd U.S. Circuit Court of Appeals, while Kenneth works for the other one. However, the classifier predicts Ralph works for both, and that Kenneth works for both.

Recall

Recall is naturally hindered by the NER's classifier performance. For instance, if the NER misses all of the PERSON entities, and classifies them as MISC, our Log-Reg model will not have a chance to detect the Work-For relation! Furthermore, if the NER extracts only part of the entity (e.g., John works for Korean CIA - only CIA was detected as an organization, but the annotation contains 'Korean CIA'), and the classifier will predict that John Work-For CIA, then we have a Recall error (not detecting John Work-For Korean CIA), and a Precision error (detecting John Work-For CIA). In other words, our Log-Reg model is as good as the NER classifier we use. It is worth mentioning that while testing for a better NER classifier, the one we have right now proved to be the best; Stanza's CoNLL03.

Evaluation

	Train			Dev		
Features	Precision	Recall	F1	Precision	Recall	F1
Words	0.680	0.752	0.715	0.541	0.623	0.579
+Dependency Tree	0.731	0.725	0.728	0.608	0.585	0.596
+Entities Embedding	0.717	0.743	0.730	0.608	0.585	0.596
+Rules	0.736	0.743	**0.740**	0.608	0.585	**0.596**

Table 1: Results for Train and Dev set with entities in their entirety

We observe improvement with Dependency Tree feature. The path from one entity to another increases the precision by $\sim 5\%$ while decreases the recall by $\sim 3\%$, and overall improved the F1 score. Moreover, our rule based system improved the train precision as expected, without change in the dev set.

	Train			Dev		
Features	Precision	Recall	F1	Precision	Recall	F1
Words	0.811	0.862	0.836	0.598	0.698	0.644
+Dependency Tree	0.870	0.835	0.852	0.696	0.670	0.683
+Entities Embedding	0.850	0.853	0.851	0.696	0.670	0.683
+Rules	0.873	0.853	**0.863**	0.696	0.670	**0.683**

Table 2: Results for Train and Dev set with entities in soft method