**Assignment 4 – Relation Extraction (RE)**

Oved Nagar: 302824875, Karin Haim: 205788326.

we chose to focus on the **Work-For** relation.

Note: a trained model is attached, but to train a new model the folder of GloVe must contain the file: "glove.6B.50d.txt"

**Model Overview:**

we have a model that is a hybrid of deep learning and rule-based. the model consists of 4 main layers:

* First layer – extracting potential ordered pairs of (Person, Organization) from each sentence
* Second layer – give each pair a score using deep learning approach
* Third layer – give each pair a s score using rule-based approach
* Forth layer – combine score and tag each pair accordingly

**First Layer:**

this stage is crucial – missing pairs at this stage set a boundary for recall, while to many pairs will make deep learning stage less efficient.

* using NER tagger the model tags each word in the sentence
* two lists are created for each sentence:   
  - list of potential people  
  - list of potential organization

Failed attempts for first layer

we also tried using np-chunk from Spicy directly but it's recall (with respect to person, organization pairs detection) was smaller, moreover we tried to make our filtering for the pairs softer, e.g. include in potential people words that start with capital letter, it increased our potential recall for this stage, but eventually it gave a wide range of samples, resulting with lower recall from the deep model.

**Second Layer:**

We used Pytorch environment to create our model, our goal was to create a model that learns the structure of the sentence, represent each word with a vector and tag each (person, organization) using the relevant vectors

Sentence representation

each sentence is represented by:

* letter embeddings for each word (followed by LSTM to give word representation)
* GloVe word embeddings
* Embeddings for POS tag and NER tag
* An integer for each word representing the relative location of its parent in the semantic tree

Attention

We used attention to learn the structure of the sentence:

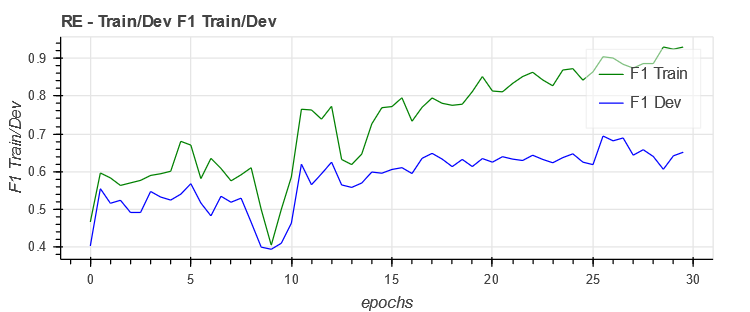
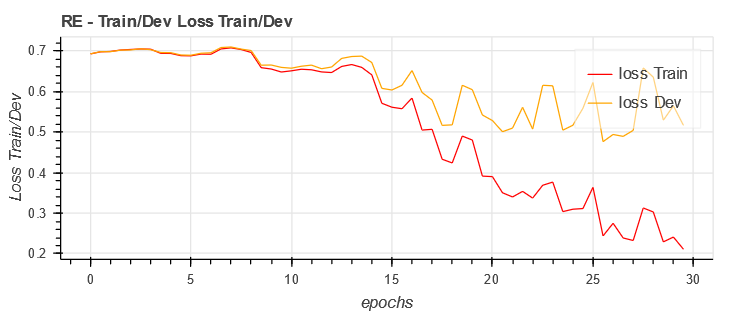
* 3 layered Bi-LSTM layer, with skip connections
* Concatenation of max-pool, avg-pool, and a Gate-Attention vector (calculated by the following article, Recurrent Neural Network-Based Sentence Encoder with Gated Attention)

Final Classification

* Another Bi-LSTM layer, this time the attention is concatenated to each word vectors that includes only character and word embeddings.
* We take the output ant use max-pool for the vectors representing the person, and for the vectors representing the organization.
* the concatenated vector for per-org is then goas through an MLP

because we had to work with small imbalanced data-set we used cross-validation for the learning stage, and for each batch we sampled examples according to the labeling distribution.

figures of F1, recall, precision by epoch:



Failed attempts for second layer

Our first attempt was without the attention layer, and the results were much worse.

**Third Layer:**

Because the data-set was small we tried to make the rules as generic as we can. We checked two things, semantic dependency, and semantic distance:

Semantic dependency

We follow the dependency tree from the person and from the organization up to an intersection (if exists) and collect the dependencies on the way up. An attribute is a pair (PER/ORG, dep), PER/ORG represents the source leaf node and the dep represents the dependency.

Then we calculate the following:

We also assume that the distance in the tree from person to the organization is normally distributed, so we calculate the mean and std of the distance to create the corresponding distribution.

**Forth Layer:**

At this stage we simply combine the results:

**Our results:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | recall | F1 |
| Work-For | 0.643564 | 0.601851 | 0.62200 |