MSiA414 SEC01
Text Analytics
Lab 7 - NER

Timo Wang

Northwestern University

October 19th, 2021

What is NER?

NER stands for **n**amed **e**ntity **r**ecognition. It is a method for extracting relevant entities from a large corpus and assigning them with a predefined category.

What is NER?

When Sebastian Thrun PERSON started at Google ORG in 2007 DATE , few people outside of the company took him seriously. "I can tell you very senior CEOs of major American NORP car companies would shake my hand and turn away because I wasn't worth talking to," said Thrun PERSON , now the co-founder and CEO of online higher education startup Udacity, in an interview with Recode ORG earlier this week DATE .

A little less than a decade later DATE , dozens of self-driving startups have cropped up while automakers around the world clamor, wallet in hand, to secure their place in the fast-moving world of fully automated transportation.

Figure: In this example, there are four different categories.

What makes NER tagging difficult?

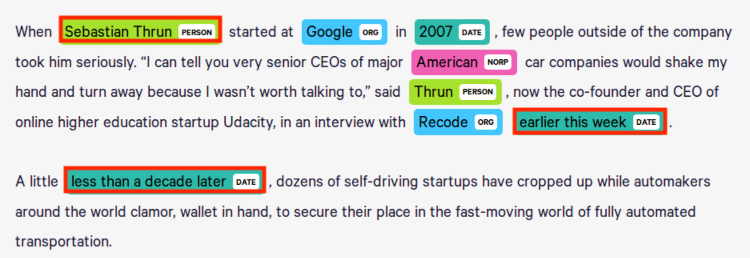

Figure: Notice that some entities comprise actually more than one word. We need explicitly the context information to determine the correct tag for a word.

Conditional Random Field (CRF)

For an input sequence $\mathbf{X}$, the probability of the output vector $\mathbf{y}$ is

$$p(\mathbf{y} \mid \mathbf{X}) \tag{1}$$

For a binary classification problem, we can reduce it to the following and through gradient decent update the parameters used in the linear transformation.

$$p(\mathbf{y} \mid \mathbf{X}) = \sigma(\mathrm{T}(\mathbf{X})) \tag{2}$$

Conditional Random Field (CRF)

However, since we want to utilize certain features, especially the context words, to make predictions, we need some model that let us explicitly specify that.

$$p(\mathbf{y} \,|\, \mathbf{X}) = \frac{1}{\mathrm{Z}(\mathbf{X})} \exp\left(\sum_{i=1}^{n} \sum_{j} \lambda_j \mathrm{f}_j(\mathbf{X}, i, \mathbf{y}_{i-1}) \right) \tag{3}$$

$$\mathrm{Z}(\mathbf{X}) = \sum_{\mathbf{y} \in \mathbf{Y}} \sum_{i=1}^{n} \sum_{j} \lambda_j \mathrm{f}_j(\mathbf{X}, i, \mathbf{y}_{i-1}) \tag{4}$$

Conditional Random Field (CRF)

$\mathrm{f}_j(\mathbf{X}, i, \mathbf{y}_{i-1}, \mathbf{y}_i)$ is a feature function which takes as input the set of input vectors $\mathbf{X}$, position of the data point we want to predict i, as well as the label of the data point at index $i-1$ $\mathbf{y}_{i-1}$.

λ_j is the weight for the j-th feature function and is learned through training (gradient descent).

What about "automatic" features?

One way to use CRF is to select our own sets of features. However, this requires very well planned feature engineering.

Question

How do we avoid feature engineering?

What about "automatic" features?

We rely solely on the data itself and deep neural networks to uncover those features for us.

This is essentially what bi-LSTM + CRF is based on.

What about "automatic" features?

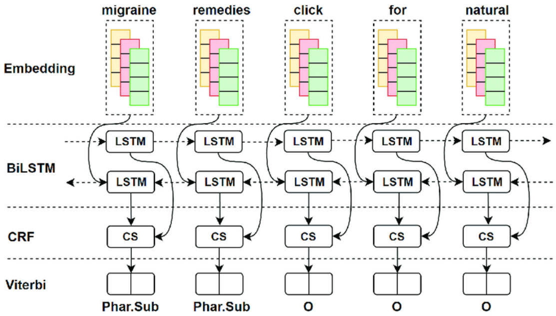

Figure: Notice here that the CRF layer takes as input the output from the LSTM states in both directions. The values in the output vectors serve as features here.