Name Entity Recognition

Business Insider Articles 2013~2014

James Jia, sjs8610, Northwestern IE308

Executive Summary

In this assignment, the goal is to be able to identify and extract various categories of name entities from the text files. The three categories of interests are CEO names, company names, and numbers involving percentages.

Various text processing methodologies are adopted in order to complete the tasks. Python **`nltk`** package is used to conduct text processing, such as tokenizing, stop word removal, and lemmatization. Python **`re`** regular expression package is used to explore and extract various text characteristics and snippets. Both positive samples and negative samples were labeled and marked according to features. Such datasets were then trained into a logistic regression model using the Python **`sklearn`** package. Finally, using the model, original text files were fed to the model and extract the desired entities. Python **`spacy`** package is also used to simplify the final output generation process.

Although we do not have the ground truth output (the correct answers), based on the logistic regression, CEO name extraction model obtained a 83.0% accuracy, company name model obtained a 72.9% accuracy, and the percentage value model performs well, at 99.7%.

Glossary:

Category	CEO names, company names, percentage values
Entity	An output of category. Example: Steve Jobs, Apple Inc., 10%
Entity suspect	A candidate for an entity. Example: Joe Biden, UN, 9
Snippets	Tokens around entity suspects. Example: XXXX Steve Jobs XXXX
First appearance	The first time an entity suspect appears in text, sorted by dates (ascending order)
Feature	Describe the text snippet as true (1) or false (0). Example: "He's a CEO!". Feature: whether snippet includes "CEO", feature = 1.

Overall Methodology

In this section, we will describe the process taken that is common to all three categories of extraction. This section will cover the preprocessing techniques, training and entity extraction methodology. The specific features for each category of entity in logistic regression will be covered in the next section.

Overall, in order to generate the entity outputs, we need to apply supervised learning. More specifically, we want to create 3 different logistic regression models. The input of the model **`X`** should be an "entity suspect" wrapped by a text snippet, output of the model **`Y`** should be the decision (whether **`X`** actually contains the correct entity). Above described is accomplished via the following:

1. ***Text pre-processing***
   1.1. All date-by-date text files are combined using `glob`. In order to locate the position of the labelled entity, searching through the entire text file might be more efficient.
   1.2. Garbage symbols such as "\", "," are first removed. The raw text file is then tokenized using `nltk.tokenize.word_tokenize`, with each "word" as an array element.
   1.3. Stop words such as "a" and "the" are removed using `nltk.corpus.stopwords`
   1.4. Tokens are then lemmatized using `nltk.stem.WordNetLemmatizer`. Normalization and Stemming are not used, because we might want to preserve the capitalization and conventional spelling for later usage.
2. ***Labels import***
   2.1. Labels (CEO names, company names, percentages) are imported in csv format to a dataframe. However, special processing was made for CEO names and percentage values. More details in the next section.
3. ***General method to build training sets:***
   3.1. For every positive label of an entity, only the location of its first ever appearance will be recorded. The assumption is that, when the entity (such as names) appears the first time, the description for that entity is the fullest. This is done by helper function `find_name_match()`.
   3.2. At every location, the text snippet is constructed by 7 tokens before the location, the token at the key location, and 7 tokens after the location. Examples seen below. This is done by helper function `create_snippet()`.

	name	snippet
0	Tom Horton	Airways CEO Doug Parker American 's CEO Tom H...
1	Patti Hart	laying course On February 1st IGT CEO Patti H...
2	Jamie Dimon	three come Wall Street The list start Jamie D...

Figure 1: text snippet in dataframe

   3.3. Negative features are generated using category-specific methods detailed in the next section.
   3.4. Text snippets for negative samples are generated similarly described in 3.1 and 3.2.
   3.5. Labels are appended to the training sets for both positive (1) and negative (0) samples.
4. ***Train logistic regression model:***
   4.1. For each category, different features are used as `X` variables in the logistic model. The features are detailed in the next section. Each feature is constructed using `regex`, and the feature output for each snippet is either 1 or 0. Example see below.

	snippet	label	f1	f2	f3	f4	f5	f6
0	Airways CEO Doug Parker American 's CEO Tom H...	1	1	0	0	0	0	1
1	laying course On February 1st IGT CEO Patti H...	1	1	0	0	0	0	0
2	three come Wall Street The list start Jamie D...	1	0	0	0	0	1	0
3	's chief investment office course move Facebo...	1	0	0	0	0	0	1

Figure 2: text snippet with features and labels in dataframe

   4.2. Standard logistic regression is applied to the training sets with 90%-10% training-testing split. The model is trained by `sklearn.linear_model.LogisticRegression`.

5. ***Final output:***

5.1. For each text file (by date), `spacy NER` package is first applied to extract the "entity suspect". This helps to significantly reduce the computational power needed to produce the output. For example, to find CEO names, we will only look at the "`PERSON`" labelled entities. Example see below.

Figure 3: spacy NER output labeled by displaCy visualizer.

5.2. Only the first ever appearance of an "entity suspect" will be recorded. For example, entity "James Jia" as a suspect for CEO name appears in 300 text files, only the 1st will be recorded.

5.3. Finally, after extracting a unique list of entity suspects and their text snippets (in first appearance), the logistic regression is applied. Only the list of entities with positive outcomes will be retained and output.

Category Specific Methodology

In this section, we will describe specific processing methodology that is different among categories. This will cover any label-cleaning assumptions, features, negative samples generation methods, and any other assumptions.

A) CEO names:

a) Assumptions:

i) During label pre-processing, 1-word names are not considered, because it is assumed that most people have at least 2 tokens for their names.

ii) It's assumed that the first appearance of any name in any article will provide the fullest context. For example, it is assumed that "Apple CEO Tim Cook" will be mentioned first, and only "Tim" or "Mr. Cook" will be mentioned in the following text.

iii) It is assumed that the logistic regression will run only snippets that already contain names. This is done by running the `spacy NER` module on the raw text first. Therefore, the task is to differentiate between CEO names and any other names.

b) Negative samples:

i) There are a lot of **politicians' names** in Business Insider. Hence, politicians' names (Senators, Representatives from 113th Congress, and world leaders' names) will be negative samples. This is assumed that those politicians are not CEOs (which might not be true occasionally, such as Mitt Romney, President of Bain Capital).

ii) There are also many economists' names, filtered by the token "Dr.".

iii) There are also a good number of editors' names, filtered by the token "edit".

c) Features:
   i) `'[Cc][Ee][Oo]'` → The snippet contains the word “CEO”
   ii) `'[Ee]xecutive'` → Contains the word “executive”
   iii) `'[Ll]ead'` → Contains the word “lead”
   iv) `'[Ss]aid'` → Contains the word “said”
   v) `'\$'` → Contains the “dollar $” symbol
   vi) `'\'s'` → Contains the possessive term “ ‘s “.

B) <u>Company names:</u>

a) Assumptions:
   i) It is assumed that the logistic regression will run only snippets that already contain names of an organization. This is done by running the `spacy NER` module on the raw text first. Therefore, the task is to differentiate between companies and any other organizations.

b) Negative samples:
   i) Negative samples include snippets that are clearly NGO, by containing tokens “council”, “foundation”, “association”.
   ii) Contain famous international organizations acronym: “WTO”, “UN”, “WHO”, “AIESEC”, “NATO”, “ASEAN”.

c) Features:
   i) `'[Cc][Ee][Oo]'` → contains the token “CEO”
   ii) `'Inc'` etc → contains the token “Inc”, “Corp”, “Company”, “Co”, “Group”, “Ltd”, “LLC”
   iii) `'Media'` etc → contains the token “Media”, “Entertainment”, “Management”, “Capital”, “Bank”
   iv) `'[0-9]+'` → contains numbers
   v) `'\$'` → contains money symbol
   vi) `'[A-Z][A-Z]+(?=[\ \t\n\?\!\;])'` → the number of all-caps (at least 2 letters) tokens

C) <u>Percentages:</u>

a) c
   i) It is assumed that the logistic regression will run only snippets that already contain like-numbers (“`PERCENTAGE`” and ‘`CARDINAL`”). This is done by running the `spacy NER` module on the raw text first. Therefore, the task is to distinguish out the percentages.

b) Negative samples:
   i) One negative sample includes snippets that were identified by `spacy NER`’s “`CARDINAL`” class. One confusing exception in that class is entities such as “one percent”, where the numerical digit is represented by text, and “percentage” is spelled out as text.

c) Features:
   i) `'^(?:[^ ]*\ ){9}([^ ]*)'` → the 9th token contains “percent”
   ii) `'^(?:[^ ]*\ ){9}([^ ]*)'` → the 9th token contains “%”
   iii) `'increase'` etc → contains token “increase”, “grew”, “hit”, “surge”, “climb”, “increase”
   iv) `'decrease'` etc → contains token “decrease”, “fell”, “collapse”, “slid”
   v) `'[0-9]+(?=[\ ])'` → Number of digit-based token

Performance

A) ***Accuracy:***

The percentage value's model performance is the best out of the three, achieving 99.7% accuracy. The company name model has the poorest performance, only achieving 72.9% accuracy. The CEO name model performs well, at around 83.0%.

B) ***Confusion Matrix:***

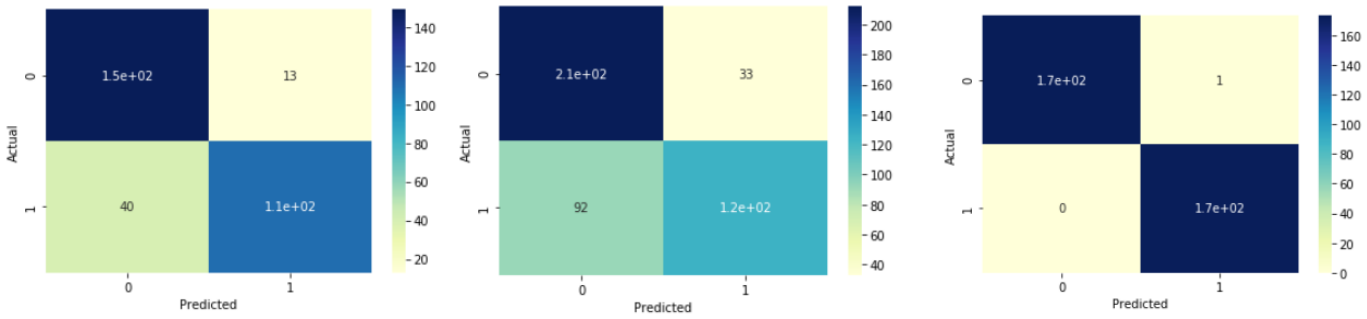

Figure 4 (left to right): training-testing confusion matrix for CEO names, companies, percentage values.

Following part (A), based on the testing results, it is clear that the accuracy for percentage values did comparably better than CEO names and company names. For CEO and company names, false negative is much more common than false positives -- meaning that the models have a hard time picking out true positive samples, but have little problem rejecting negative ones.

C) ***Feature coefficients:***

Investigating closely the logistic model, feature coefficients can provide good insights, although its p-value is not available for the `sklearn` package.

- In CEO model, the inclusion of words "CEO" is the most positively influential model. The inclusion of possessive terms has a reverse impact.
- In the company name model, again the inclusion of the word "CEO" becomes the most positive influential model. This is understandable as the company name and its CEO name often appears together.
- In the percentage value model, the inclusion of words "percent" and "%" right after numerical / number-based text has a highly positive influence.

D) ***Improvement:***

While the positive-negative training samples are well balanced, and the algorithm yielded decent results, many improvements can be made to enhance the model. For example, the elimination of some stop words such as "of" might bias against company names such as "Bank of America". Additionally, `spacy` and `nltk` packages seem to have different tokenization rules, resulting in further discrepancies. Finally, there should definitely be more investigation into feature-building, such as understanding feature p-value and iterating the model.