Table of contents

- Introduction
- Theory
- Implementation
- Parallelization
- Conclusion
NER Concept

- **Named-entity recognition** (NER) is a subtask of **Information Extraction** that seeks to locate and classify elements in text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc.

  [Wikipedia]

- **Why NER?**
  - Event Extraction
  - Question Answering
  - Text summarization
A man opposed to the joint South Korea-U.S. military drills attacked the American ambassador, Mark Lippert, in Seoul Thursday morning.
Table of contents

- Introduction
- **Theory**
- Implementation
- Parallelization
- Conclusion
Problem Definition

- Input: a word sequence
  \[ \text{word}_\text{sequence} = \langle X_1, X_2, X_3, X_4, \ldots, X_n \rangle \]

- Output: their Named-Entity tag sequence
  \[ \text{tag}_\text{sequence} = \langle Y_1, Y_2, Y_3, Y_4, \ldots, Y_n \rangle \]

- Items in \( \langle Y_1, Y_2, Y_3, Y_4, \ldots, Y_n \rangle \) might be person, location, organization, etc.
How to assign the NE tag for a particular word?

- Let us consider the following scenarios:
  - I love the city of New York. (New York is a location)
  - New York Times discloses the inside story. (New York is a news organization)
  - Jeremy Lin unexpectedly led a winning turnaround with New York in 2012. (New York is a sport organization)

- The context is a very important factor to assign the NE tag.
Linear-Chain CRF

- CRF: Conditional Random Field that is an undirected graph whose nodes correspond to YUX. This graph is parameterized in the same way as an Markov network, as a set of factors $\phi_1(D_1), ..., \phi_m(D_m)$.

\[
P(Y|X) = \frac{1}{Z(X)} \, P^\sim(Y,X) \quad \quad P^\sim(Y,X) = \prod_{t=1}^{m} \phi_t(D_t) \quad \quad Z(X) = \sum_{Y} P^\sim(Y,X)
\]

- Goal: Get the conditional distribution $P(Y | X)$, where $Y$ is a set of target variables and $X$ is a set of observed variables.
- Linear-Chain CRF:
  - Based on the basic CRF, and it has only two factors for each word:

  \[
  \phi_t^1(Y_t, Y_{t+1}) \quad \quad \phi_t^2(Y_t, X_1, ..., X_T)
  \]
Linear-Chain CRF

- Graphical Model of Linear-Chain CRF

- Two Factors:
  - \( \phi_1^t(Y_t, Y_{t+1}) \) represents the dependency between neighboring target variables. And \( \phi_2^t(Y_t, X_1, ..., X_T) \) represents the dependency between a target and its context in the word sequence.
  - Arbitrary features of the entire input word sequence.
  - Log-linear model, but not table factor

- Forward-backward Algorithm to compute the probability distribution

- Viterbi (Dynamic Programming) to choose the best tag sequence by maximizing the probability
Table of contents

- Introduction
- Theory
- Implementation
- Parallelization
- Conclusion
NER Architecture

NER System based on CRF

[Asif Ekbal]
Eyewitnesses have described the carnage and terror that ensued as gunmen forced their way into the office of the French satirical Charlie Hebdo magazine in Paris before shooting dead 12 people…

{LOCATION=France | Paris | …
ORGANIZATION=UK | European Union | …
PERSON=Charlie Hebdo | Michel Houellebecq | …
DATE=Thursday | January 2015}
Table of contents

- Introduction
- Theory
- Implementation
- Parallelization
- Conclusion
NER Parallelization on Hadoop

Key: Doc ID
Value: NE String

Worker node 1 → Mapper
Worker node 2 → NER
Worker node N → Reducer

mapper

Reducer

NER

Hadoop

WebpageNoise Reduction
TweetNoiseReduction schema

AVRO

Input

HDFS

AVRO

Output

WebpageNER schema
TweetNER schema

Key: Doc ID
Value: NE String
Map-Reduce Implementation

- Driver for Map-Reduce job
- Set configuration
- Read a document
- Perform NER
- Write NE string
- Parse NE string
- Write AVRO file

Public class **NERDriver** extends Configured implements Tool{
    public int run(String[] args) {}
    public static void main(String[] args) {}}

Public static class **AvroNERMapper** extends Mapper<AvroKey<WebpageNoiseReduction>, NullWritable, Text, Text> {
    protected void map(AvroKey<WebpageNoiseReduction> key, NullWritable value, Context context) {}}

Public static class **AvroNERReducer** extends Reducer<Text, Text, AvroKey<WebpageNER>, NullWritable> {
    protected void reduce(Text key, Iterable<Text> value, Context context) {}
Input & Output

AVRO Schema of Input File
(By Noise Reduction Team)

```json
{"type": "record", "namespace": "cs5604.tweet.NoiseReduction", "name": "TweetNoiseReduction", "fields":
  ...
}
```

AVRO Schema of Output File
(By Hadoop Team)

```json
{"namespace": "cs5604.tweet.NER", "type": "record", "name": "TweetNER", "fields": [
  {"name": "doc_id", "type": "string"},
  {"doc": "analysis", "name": "ner_people", "type": ["string", "null"],
  {"doc": "analysis", "name": "ner_locations", "type": ["string", "null"],
  {"doc": "analysis", "name": "ner_dates", "type": ["string", "null"],
  {"doc": "analysis", "name": "ner_organizations", "type": ["string", "null"]
}]
```
Input & Output (Cont.)

AVRO Output File with Named Entities

{\textit{ner\_dates}: \textit{December 08 | December 11}, \textit{ner\_locations}: None, \textit{doc\_id}: \textit{winter\_storm\_S--100052}, \textit{ner\_people}: None, \textit{ner\_organizations}: \textit{NWS}}

{\textit{ner\_dates}: None, \textit{ner\_locations}: None, \textit{doc\_id}: \textit{winter\_storm\_S--10025}, \textit{ner\_people}: \textit{Blaine Countys}, \textit{ner\_organizations}: None}

{\textit{ner\_dates}: None, \textit{ner\_locations}: None, \textit{doc\_id}: \textit{winter\_storm\_S--100229}, \textit{ner\_people}: None, \textit{ner\_organizations}: \textit{ALERT Winter Storm Watch}}

{\textit{ner\_dates}: None, \textit{ner\_locations}: None, \textit{doc\_id}: \textit{winter\_storm\_S--100364}, \textit{ner\_people}: None, \textit{ner\_organizations}: \textit{Heavy Snow Possible Winter Storm Watch | Northeast PA | Coal Region Endless Mtns}}

...

(From \textit{winter\_storm\_S Tweet collection})
# Statistics

<table>
<thead>
<tr>
<th>Collections</th>
<th>Size</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>winter_storm_S (Tweet)</td>
<td>166 MB</td>
<td>6 Min</td>
</tr>
<tr>
<td>storm_B (Tweet)</td>
<td>6.3 GB</td>
<td>10 Min</td>
</tr>
<tr>
<td>winter_storm_S (Webpage)</td>
<td>62 MB</td>
<td>4 Min</td>
</tr>
<tr>
<td>storm_B (Webpage)</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Table of contents

- Introduction
- Theory
- Implementation
- Parallelization
- Conclusion
Conclusion

- Investigate the theory of NER
- Implement NER prototype based on the Stanford NER tool
- Parallelize NER on Hadoop
- Export NEs to AVRO files
Acknowledgement

- This project is supported by the US National Science Foundation through grant IIS - 1319578.
- We express our deep appreciation to the Instructor Dr. Fox, GTA Sunshin Lee and GRA Mohammed for their help.
Thank You!

Q & A
NER Data/Back-Offs

- CoNLL-2002 and CoNLL-2003 (British newswire)
  - Multiple languages: Spanish, Dutch, English, German
  - 4 entities: Person, Location, Organization, Misc
- MUC-6 and MUC-7 (American newswire)
  - 7 entities: Person, Location, Organization, Time, Date, Percent, Money
- ACE
  - 5 entities: Location, Organization, Person, FAC, GPE
- BBN (Penn Treebank)
  - 22 entities: Animal, Cardinal, Date, Disease, …