FINET
Context-Aware Fine-Grained Named Entity Typing

Luciano Del Corro*, Abdalghani Abujabal*, Rainer Gemulla†, and Gerhard Weikum*

Max-Planck-Institute for Informatics*
University of Mannheim†
Named Entity Typing

The task of detecting type(s) of named entities in a given context with respect to a type system (e.g., WordNet)

“Page plays his guitar on the stage”

guitarist
FINET

A system

• for detecting *fine-grained types*
• in *short inputs* (e.g., sentences or tweets)
• in a given *context*
• with respect to *WordNet*
Context-Aware Typing

“Steinmeier, the German Foreign Minister, ..”
Context-Aware Typing

“Steinmeier, the German Foreign Minister, ..”

foreign minister explicit
Context-Aware Typing

“Steinmeier, the German Foreign Minister, ..”

foreign minister

“Messi plays soccer”
Context-Aware Typing

“Steinmeier, the German Foreign Minister, ..”

“Messi plays soccer”

foreign minister

soccer player

explicit

almost explicit
“Steinmeier, the German Foreign Minister, ..”

“Messi plays soccer”

“Pavano never even made it to the mound”
Context-Aware Typing

“Steinmeier, the German Foreign Minister, ..”

foreign minister

“Messi plays soccer”

soccer player

“Pavano never even made it to the mound”

baseball player
Applications

• KB Construction
  • find types for existing entities
Applications

• KB Construction
  • find types for existing entities
• Named Entity Disambiguation
  • “Page played amazingly on the stage”

Musician

Businessman
Applications

- KB Construction
  - find types for existing entities
- Named Entity Disambiguation
  - “Page played amazingly on the stage”

Musican

Businessman
Applications

- KB Construction
  - find types for existing entities
- Named Entity Disambiguation
  - “Page played amazingly on the stage”
- Semantic Search
  - Give me all documents talk about musicians
Supervised Approaches

• Manually labeled data is scarce
  • thousands of types, need sufficient training data for every type
Distantly Supervised Approaches

• Idea: automatically generated data via KB (e.g., Wikipedia)
Distantly Supervised Approaches

- Idea: automatically generated data via KB (e.g., Wikipedia)

  “Klitschko is the mayor of Kiev”

  “Klitschko is known for his powerful punches”
Distantly Supervised Approaches

• Idea: automatically generated data via KB (e.g., Wikipedia)

boxer  “Klitschko is the mayor of Kiev”

mayor  “Klitschko is known for his powerful punches”

politician
Distantly Supervised Approaches

• Idea: automatically generated data via KB (e.g., Wikipedia)

boxer
mayor
politician

“Klitschko is the mayor of Kiev”
“Klitschko is known for his powerful punches”

Problem: types are context-oblivious
FINET

- Unsupervised
  - Most extractors are unsupervised
FINET

• Unsupervised
  • Most extractors are unsupervised

• Context-aware
  • “Klitschko is the mayor of Kiev”

\textit{mayor, politician}
FINET

• Unsupervised
  • Most extractors are unsupervised

• Context-aware
  • “Klitschko is the mayor of Kiev”

• Super fine-grained
  • WordNet as typing system (16K types; per, loc, org)
FINET Overview

1. Preprocessing

2. Candidate Generation
   1. Pattern-based extractor [very explicit]
   2. Mention-based extractor [explicit]
   3. Verb-based extractor [almost explicit]
   4. Corpus-based extractor [implicit]

3. Type Selection (via WSD)
Stopping condition met?

Yes

Type Selection

No

Subsequent Extractor

Extractor
Preprocessing

“Albert Einstein discovered the law of photoelectric effect and he won the Nobel price in 1921”
Preprocessing

“Albert Einstein discovered the law of photoelectric effect and he won the Nobel price in 1921”

- Identify clauses
  - Some extractors operate on clause level (clauses capture local context)
Preprocessing

“Albert Einstein discovered the law of photoelectric effect and he won the Nobel price in 1921”

- Identify coarse-grained types [Stanford NER]
- FINET restricts its candidates to hyponyms
- Well studied task: high prec. and recall
- “Albert Einsten”: PER
Preprocessing

“Albert Einstein discovered the law of photoelectric effect and he won the Nobel price in 1921”

- Coreference resolution
  - (“Albert Einstein”, “he”)
FINET Overview

1. Preprocessing

2. Candidate Generation
   1. Pattern-based extractor [very explicit]
   2. Mention-based extractor [explicit]
   3. Verb-based extractor [almost explicit]
   4. Corpus-based extractor [implicit]

3. Type Selection (via WSD)
Pattern-based Extractor

[final patterns]

targets very explicit types

• “Barack Obama, the president of […]”

• [“Barack Obama”; president-1, president-2, ..]
Pattern-based Extractor
[final patterns]

targets very explicit types

• “Barack Obama, the president of […]”

• [“Barack Obama”; president-1, president-2, ..]

NAMED ENTITY , (modifier) NOUN (modifier)
Pattern-based Extractor

[final patterns]

targets very explicit types

• “Barack Obama, the president of […]”

• [“Barack Obama”; president-1, president-2, ..]

Stopping Condition: produce at least one type
Pattern-based Extractor
[non-final patterns]

• “Shakespeare’s productions”
  - production $\xrightarrow{\text{DER}}$ produce $\xrightarrow{\text{DER}}$ producer
  
  [“Shakespeare”; producer-1, producer-2, ..]

Poss. + transf.
Pattern-based Extractor
[non-final patterns]

• “Shakespeare’s productions”
  
  production $\xrightarrow{\text{DER}}$ produce $\xrightarrow{\text{DER}}$ producer

  [“Shakespeare”; $\text{producer-1, producer-2, ..}]

Poss. + transf.

Stopping Condition: KB lookup
Method Overview

1. Preprocessing

2. Candidate Generation
   1. Pattern-based extractor [very explicit]
   2. Mention-based extractor [explicit]
   3. Verb-based extractor [almost explicit]
   4. Corpus-based extractor [implicit]

3. Type Selection (via WSD)
Mention-based Extractor

- “Imperial College London”
- [“Imperial College London”; college-1, college-2, ..]
Mention-based Extractor

- “Imperial **College** London”
- [“Imperial College London”; **college-1**, **college-2**, ..]

**Stopping Condition**: KB lookup
Method Overview

1. Preprocessing

2. Candidate Generation
   1. Pattern-based extractor [very explicit]
   2. Mention-based extractor [explicit]
   3. Verb-based extractor [almost explicit]
   4. Corpus-based extractor [implicit]

3. Type Selection (via WSD)
Verb-based Extractor

Verb-argument semantic concordance

• Nominalization
  • “play” → “player”

  verb      deverbal noun
Example 1: Suffixes

- “Messi plays in Barcelona”
Example 1: Suffixes

• “Messi plays in Barcelona”
  play → “-er” → player
Example 1: Suffixes

- “Messi plays in Barcelona”
  play → “-er” → player

  play-1 $\xrightarrow{\text{DER}}$ player-1 ($player$)
  play-2 $\rightarrow$ player-2 ($musician$)
  play-3 $\rightarrow$ player-3 ($actor$)
  
  player-4 ($participant$)
Example 1: Suffixes

• “Messi plays in Barcelona”
  play “-er” player

  play-1 DER player-1 (player)
  play-2 player-2 (musician)
  play-3 player-3 (actor)
  . player-4 (participant)

[“Messi”; player, musician, actor, ..]
Example 1: Suffixes

• “Messi plays in Barcelona”
  play “-er” player

 play-1 DER player-1 (player)
 play-2 → player-2 (musician)
 play-3 → player-3 (actor)

player-4 (participant)

[“Messi”; player, musician, actor, ..]

Stopping Condition: KB lookup
Example 2: Synonyms

- “John committed a crime”
- \( \text{commit} \xrightarrow{\text{syn}} \text{perpetrate} \xrightarrow{\text{DER}} \text{perpetrator} \)
  
  [“John”; \textit{perpetrator-1}]

Stopping Condition: KB lookup
Method Overview

1. Preprocessing

2. Candidate Generation
   1. Pattern-based extractor [very explicit]
   2. Mention-based extractor [explicit]
   3. Verb-based extractor [almost explicit]
   4. Corpus-based extractor [implicit]

3. Type Selection (via WSD)
Corpus-based Extractor

Distributional hypothesis: similar entities tend to occur in similar context

- “Messi” & “Cristiano Ronaldo” occur in sport (soccer)
- Key idea: Collect types of similar entities via KB
Word2Vec

- Word vectors represent semantic contexts for a given phrase
- Given a set of phrases, return the $k$ most similar phrases with respect to context
“Maradona expects to win in South Africa”

query: {“Maradona”, “South Africa”}

<table>
<thead>
<tr>
<th>Mention</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Diego Maradona”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>“Parreira”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>“Carlos Alberto Parreira”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>“Dunga”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>…</td>
<td></td>
</tr>
</tbody>
</table>

“Parreira coached Brazil in South Africa”
“Dunga replaced Parreira after South Africa”
“Maradona expects to win in South Africa”

query: {“Maradona”, “South Africa”}

<table>
<thead>
<tr>
<th>Mention</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Diego Maradona”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>“Parreira”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>“Carlos Alberto Parreira”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>“Dunga”</td>
<td>&lt;coach-1&gt;, ..</td>
</tr>
<tr>
<td>…</td>
<td></td>
</tr>
</tbody>
</table>

“Parreira coached Brazil in South Africa”
“Dunga replaced Parreira after South Africa”

Stopping Condition: sufficient evidence for types
Method Overview

1. Preprocessing

2. Candidate Generation
   1. Pattern-based extractor [very explicit]
   2. Mention-based extractor [explicit]
   3. Verb-based extractor [almost explicit]
   4. Corpus-based extractor [implicit]

3. Type Selection (via WSD)
Type Selection via Word Sense Disambiguation

- Given an entity and a set of candidate types
  - [“Maradona”; soccer_player-1, football_player-1, coach-1, …]
- Select the best types according to context
Entity Context for WSD

- Entity-oblivious context
  - all words in an input sentence
- Entity-specific context via lexical expansions
  - entity-related words from word vectors
Type Selection via WSD

Naive Bayes trained with word features on WN glosses and labeled data (if available) [ExtendedLesk].

“Maradona expects to win in South Africa”

Entity-oblivious context:
“expects”, “win”, “South Africa”

Entity-specific context:
“coach”, “cup”, “striker”, “mid-fielder”, and “captain”
Experiments

- Datasets
  - 500 random sentences from NYT year 2007
  - 500 random sentences from CoNLL
  - 100 random tweets
Type Granularity

- CG: (artifact, event, person, location, organization)
- FG: ~200 prominent WN types
- SFG: all remaining WN types
<table>
<thead>
<tr>
<th>System</th>
<th>Type System</th>
<th>Total Types</th>
<th>Top Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>FINET</td>
<td>WN</td>
<td>16K+</td>
<td>pers, org, loc</td>
</tr>
<tr>
<td>HYENA</td>
<td>WN</td>
<td>505</td>
<td>all</td>
</tr>
<tr>
<td>System</td>
<td>CG</td>
<td>FG</td>
<td>SFG</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>----------</td>
<td>-----------</td>
</tr>
<tr>
<td></td>
<td>P</td>
<td>Correct Types</td>
<td>P</td>
</tr>
<tr>
<td>FINET</td>
<td>87.90</td>
<td>872</td>
<td>72.42</td>
</tr>
<tr>
<td>FINET (w/o l.)</td>
<td>87.90</td>
<td>872</td>
<td>71.13</td>
</tr>
<tr>
<td>HYENA</td>
<td>72.40</td>
<td>779</td>
<td>28.26</td>
</tr>
</tbody>
</table>

Results on NYT dataset
<table>
<thead>
<tr>
<th>System</th>
<th>CG</th>
<th>FG</th>
<th>SFG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>Correct Types</td>
<td>P</td>
</tr>
<tr>
<td>FINET</td>
<td>87.90</td>
<td>872</td>
<td>72.42</td>
</tr>
<tr>
<td>FINET (w/o l.)</td>
<td>87.90</td>
<td>872</td>
<td>71.13</td>
</tr>
<tr>
<td>HYENA</td>
<td>72.40</td>
<td>779</td>
<td>28.26</td>
</tr>
</tbody>
</table>

Results on NYT dataset
<table>
<thead>
<tr>
<th>System</th>
<th>CG</th>
<th>FG</th>
<th>SFG</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FINET</td>
<td>87.90</td>
<td>72.42</td>
<td>70.82</td>
<td>233</td>
</tr>
<tr>
<td>FINET (w/o l.)</td>
<td>87.90</td>
<td>71.13</td>
<td>67.11</td>
<td>204</td>
</tr>
<tr>
<td>HYENA</td>
<td>72.40</td>
<td>28.26</td>
<td>20.65</td>
<td>160</td>
</tr>
</tbody>
</table>

Results on NYT dataset
Conclusion

• FINET
  • A system for detecting types of named entities
  • Context-aware
  • Unsupervised (mostly)
  • Very fine-grained typing system
Mapping CG types to WN

- persons all descendants of
  - person-1, imaginary, being-1, characterization-3, and operator-2 (10584 in total);

- locations all descendants of
  - location-1, way-1, and landmass-1 (3681 in total);

- organizations all descendants of
  - organization-1 and social group-1 (1968 in total).
Verb-based Extractor

• “Messi plays soccer”
  • “Messi” is a subject
  • “soccer” is direct object
  • Add “soccer” as a noun modifier to the deverbal noun
Verb-based Extractor

- Utilize a corpus of frequent (verb, type) pairs
- “Messi was treated in the hospital”
  - [“Messi”; patient-1]
Corpus-based Extractor

• Retrieve 100 most related phrases along with similarity scores
Corpus-based Extractor

- Retrieve 100 most related phrases along with similarity scores
- Filter out non-entity phrases and entities not compatible with CG type
Corpus-based Extractor

- Retrieve 100 most related phrases along with similarity scores
- Filter out non-entity phrases and entities not compatible with CG type
- Traverse the result list until we collect 50% of the total score
Corpus-based Extractor

- Retrieve 100 most related phrases along with similarity scores
- Filter out non-entity phrases and entities not compatible with CG type
- Traverse the result list until we collect 50% of the total score
- If no more than 10 different types were added → add types as candidates