AFET: Automatic Fine-Grained Entity Typing by Hierarchical Partial-Label Embedding

Xiang Ren* (UIUC), Wenqi He* (UIUC), Meng Qu (UIUC), Lifu Huang (RPI), Heng Ji (RPI), Jiawei Han (RPI)
Fine-grained Entity Typing

- Assigning type labels to mentions of entities in text
- Type labels for a mention constitutes a “type-path” in a given type hierarchy (NOT necessarily ending in a leaf node)

<table>
<thead>
<tr>
<th>ID</th>
<th>Sentence</th>
<th>Type-paths</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Republican presidential candidate Donald Trump spoke during a campaign event in Rock Hill.</td>
<td>Person $\rightarrow$ politician</td>
</tr>
<tr>
<td>S2</td>
<td>Donald Trump’s company has threatened to withhold up to $1$ billion of investment if the U.K. government decides to ban his entry into the country.</td>
<td>Person $\rightarrow$ businessman</td>
</tr>
<tr>
<td>S3</td>
<td>In Trump’s TV reality show, “The Apprentice”, 16 people competed for a job.</td>
<td>Person $\rightarrow$ artist $\rightarrow$ actor</td>
</tr>
</tbody>
</table>

Text corpus with mentions of entities (bold)

A 3-level Type hierarchy

- root
  - product
  - person
  - location
  - organization
    - politician
    - artist
    - businessman
      - author
      - actor
      - singer
Fine-grained Entity Typing: Applications

- Critical task for bringing structures to unstructured text data
  - Information network construction

- A primitive step in many NLP tasks
  - Coreference resolution, relation extraction, slot filling, ...

- Knowledge base completion

- Question answering

- etc.
Fine-Grained Typing with Distant Supervision

- Manually annotating training corpora with **100+ types**
- Expensive & Error-prone

**Current practice:** using distant supervision to automatically label training corpora

<table>
<thead>
<tr>
<th>ID</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Republican presidential candidate <em>Donald Trump</em> spoke during a campaign event in Rock Hill.</td>
</tr>
</tbody>
</table>

**Automatically typed mention in training corpus**

1. **Mention:** “*Donald Trump*”; **Context:** S1; **Candidate Types:** {**person**, **politician**, **businessman**, **artist**, **actor**}
Problem Definition

- **Input:**
  1. Automatically labeled training corpus $D$
  2. Target type hierarchy $T$
  3. Knowledge base (KB), and entity-type facts $\rightarrow$ (entity, type) tuples
  4. Test data $D_T \rightarrow$ (mention, context) tuples
     - short text inputs: sentences, tweets, user queries, etc.

- **Output:**
  - Estimate *a single type-path* in $T$ for each test mention in $D_T$ based on the *mention itself* and its *context*

- **Non-goals:** Entity mention detection; Entity linking; Type hierarchy creation
Existing Studies

Fine-Grained Entity Typing
- Context-independent methods → not our focus
- Context-dependent methods:
  1. Multi-label multi-class classification [FIGER-AAAI’12, etc.]
  2. Hierarchical classification [HYENA-COLING’12]
  3. Joint mention-label embedding [WSABIE-ACL’15]
  4. Label propagation [ClusType-KDD’15, etc.]

Limitations
1. Except for 2, none of them capture semantic similarity between entity types
2. All of them assume that automatically generated training labels are all “correct”
Existing Studies (cont.)

- **Partial Label Learning**
  - Assumes that “only one candidate label is correct” (1 among K is correct)
  - **Strategy 1**: Assumes equal contribution of each candidate label and average the outputs from all candidate labels for prediction [CLPL-JMLR’11]
  - **Strategy 2**: Progressively estimate true labels (as latent variables) using a EM-kind framework [PL-SVM-KDD’08]

- **Limitations**
  - Both cast the task as multi-label learning problem --- ignore the semantic similarity between entity types
  - Difficulty in handling high-dimensional feature space --- often the case in entity typing (0.4M+ features)
Fine-Grained Entity Typing: Challenges

Are we using the label information in training data in a correct and comprehensive way?
Challenge I: Noisy Type Labels

<table>
<thead>
<tr>
<th>ID</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Republican presidential candidate Donald Trump spoke during a campaign event in Rock Hill.</td>
</tr>
<tr>
<td>S2</td>
<td>Donald Trump's company has threatened to withhold up to $1 billion of investment if the U.K. government decides to ban his entry into the country.</td>
</tr>
<tr>
<td>S3</td>
<td>In Trump's TV show, “The Apprentice”, 16 people competed for a job.</td>
</tr>
</tbody>
</table>

Donald Trump is mentioned in sentences S1-S3.

- **Distant supervision**
- Assign *same* types (blue region) to all the mentions
- Does not consider *local contexts* when assigning type labels
- Introduce *label noise* to the mentions

The types assigned to entity Trump include *person, artist, actor, politician, businessman*, while only *{person, politician}* are correct types for the mention “Trump” in S1.
Challenge I: Noisy Type Labels (cont.)

- Current typing systems either **ignore this issue**
  - assume all the given type labels are “correct” labels

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Wiki</th>
<th>OntoNotes</th>
<th>BBN</th>
<th>NYT</th>
</tr>
</thead>
<tbody>
<tr>
<td># of target types</td>
<td>113</td>
<td>89</td>
<td>47</td>
<td>446</td>
</tr>
<tr>
<td>(1) noisy mentions (%)</td>
<td>27.99</td>
<td>25.94</td>
<td>22.32</td>
<td>51.81</td>
</tr>
<tr>
<td>(2a) sibling pruning (%)</td>
<td>23.92</td>
<td>16.09</td>
<td>22.32</td>
<td>39.26</td>
</tr>
<tr>
<td>(2b) min. pruning (%)</td>
<td>28.22</td>
<td>8.09</td>
<td>3.27</td>
<td>32.75</td>
</tr>
<tr>
<td>(2c) all pruning (%)</td>
<td>45.99</td>
<td>23.45</td>
<td>25.33</td>
<td>61.12</td>
</tr>
</tbody>
</table>

- Or use **simple pruning heuristics** to **delete** mentions with conflicting types
  - aggressive deletion of mentions $\rightarrow$ significant loss of training data

**The larger the target type set, the more severe the loss!**
Challenge II: Type Correlation

- Type correlation can be derived from the given type hierarchy or entity-type facts in KB
  - e.g., sim(singer, actor) > sim(politician, actor)

- However, such information are ignored in existing methods → “every type label is treated equally”

- if actor is a true label for “Trump”, both singer and politician (as the false labels) will receive equal penalty

- In fact, penalty on politician should be larger than that on singer

- → mentions of actor entities should be more similar to singer than to politician (instead of equally similar)
Our Solution

- Automatic Fine-Grained Entity Typing (AFET) Framework:
  - Jointly embed **entity mentions** (represented by text feature vectors) and **type labels** (in a given type hierarchy) into a low-dimensional space.
  - In that space, **objects with similar types should be close to each other**.
  - Essentially, we are learning embedding for each text feature and each type, and we can use them to infer types for unseen mentions in test data.
The AFET Framework: Basic Idea I

- To handle noisy type labels

Model “clean” & “noisy” mentions using different objectives

- **Clean mentions** \((m)\): \(\text{rank}(\text{true type} | m) < \text{rank}(\text{false type} | m)\)
- **Noisy mentions** \((m')\): \(\text{rank}(\text{“best” candidate type} | m') < \text{rank}(\text{false type} | m')\)

- e.g., \(\text{rank(} \text{politician} | “Bernie Sanders”) < \text{rank(} \text{businessman} | “Bernie Sanders”\))
- “Democratic presidential candidate **Bernie Sanders** will make a campaign stop Friday at the University of Wisconsin”

- e.g., \(\text{rank(} \{\text{actor, artist, politician, businessman, ...}\} | “Trump”) < \text{rank(} \text{singer} | “Trump”\))
- “In **Trump**’s TV reality show, The Apprentice, 16 people ...”
The AFET Framework: Basic Idea I

- To handle noisy type labels

Model “clean” & “noisy” mentions using different objectives

- **Clean mentions** \((m)\): \(\text{rank}(\text{true type} | m) < \text{rank}(\text{false type} | m)\)
- **Noisy mentions** \((m')\): \(\text{rank}(“best” \text{ candidate type} | m') < \text{rank}(\text{false type} | m')\)

Progressively estimate the “best” candidate types for noisy mentions

- In each iteration, candidate types are scored based on *currently estimated mention embeddings & type embeddings* ➔ highest score as the “best”
- Text features used as side signal when learning mention embeddings
The AFET Framework: Basic Idea II

- To incorporate type correlation

  Adaptive margin between mention’s true type and false type, depending on the correlation between them

  - **Hierarchy-based correlation**: “close in the hierarchy” $\rightarrow$ more correlated
  - **KB-based correlation**: “sharing more entities in KB” $\rightarrow$ more correlated

- e.g., $\text{sim}(\text{singer, actor}) > \text{sim}(\text{politician, actor})$

- **Type distance**: $\gamma(X, Y) = 1 + 1 / [a + \text{sim}(X, Y)]$

  - $a$ is a given smoothing parameter
The AFET Framework: Basic Idea II

- To incorporate type correlation

Adaptive margin between mention’s true type and false type, depending on the correlation between them

- Require a large margin if the false type is less correlated to the true type (i.e., larger type distance between them)

- e.g., sim(“Bernie Sanders”, politician) - sim (“Bernie Sanders”, businessman)
  > γ(politician, businessman) = 1

- e.g., sim(“Bernie Sanders”, politician) - sim (“Bernie Sanders”, singer)
  > γ(politician, singer) = 10

- “Democratic presidential candidate Bernie Sanders will make a campaign stop ...”
The AFET Framework: Basic Idea II

- To incorporate type correlation

  Adaptive margin between mention’s true type and false type, depending on their correlation (distance)

  - Require a large margin if the false type is less correlated to the true type (i.e., larger type distance between them)

For a clean mention, its fine-grained true type should be ranked higher than its coarse-grained true types

- e.g., rank(politician | “Bernie Sanders”) < rank(person | “Bernie Sanders”)
The AFET Framework: Overview

1. Use distant supervision to automatically label training corpus $D$

2. Extract text features for each mention in $D$; partition training mentions $M$ into clean set $M_c$ and noisy set $M_n$, based on the given type hierarchy $T$

3. Perform Hierarchical Partial-Label Embedding to learn mention embeddings and type embeddings jointly, by solving a proposed optimization problem

4. Predict type-path for each mention in test data $D_T$ following a top-down inference procedure
Text Features for Entity Mentions

- Features are extracted from:
  - (1) mention’s name string: e.g., head token, POS tags, Brown Cluster of head token
  - (2) mention’s context in the sentence: e.g., n-grams, dependency roles

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head Token</td>
<td>Syntactic head token of the mention</td>
<td>“HEAD_Turing”</td>
</tr>
<tr>
<td>POS</td>
<td>Tokens in the mention</td>
<td>“Turing”, “Machine”</td>
</tr>
<tr>
<td>Character</td>
<td>Part-of-Speech tag of tokens in the mention</td>
<td>“NN”</td>
</tr>
<tr>
<td>Word Shape</td>
<td>All character trigrams in the head of the mention</td>
<td>“:tu”, “tur”, ..., “ng:”</td>
</tr>
<tr>
<td>Length</td>
<td>Number of tokens in the mention</td>
<td>“Aa” for “Turing”</td>
</tr>
<tr>
<td>Context</td>
<td>Unigrams/bigrams before and after the mention</td>
<td>“2”</td>
</tr>
<tr>
<td>Brown Cluster</td>
<td>Brown cluster ID for the head token (learned using D)</td>
<td>“CXT_B:Maserati ,”, “CXT_A:and the”</td>
</tr>
<tr>
<td>Dependency</td>
<td>Stanford syntactic dependency [16] associated with the head token</td>
<td>“4_1100”, “8_110111111111”, “12_111011111111”</td>
</tr>
</tbody>
</table>

- “Turing Machine” is used as an example mention from the sentence:
  - “The band’s former drummer Jerry Fuchs—who was also a member of Maserati, Turing Machine and The Juan MacLean—died after falling down an elevator shaft.”.
The Joint Embedding Model
Modeling Hierarchy-Induced Errors

- For a mention, the margin between each of its true type labels and each of its false type labels is required to be at least the hierarchy-based/KB-based distance between them.

- For a mention, a false type is ranked higher than its true type if and only if the margin between them is smaller than the hierarchy-based/KB-based distance between them.
Deriving Type Correlation and Distance
Modeling Clean Mentions

- For a clean mention, its “true types” should be ranked higher than all its “false types”
Modeling Noisy Mentions

- For a clean mention, its “best candidate type” should be ranked higher than all its “false types” (non-candidate types)
The Optimization Problem for AFET
Model Learning

- Block-wise Coordinate Gradient Descent
- Normalize the mapping matrices after each iteration
- Easy to parallelize
Type Inference

- Perform top-down search in the candidate type sub-tree to estimate the correct type-path

```
Algorithm 2: Type Inference

Input: candidate type sub-tree \( \mathcal{V}' \), mention embeddings \( \{u_i\} \),
type embeddings \( \{v_k\} \), threshold \( \eta \)
Output: estimated type-path \( \mathcal{V}_{i}^{*} \) for \( m_i \in M \)
1 for \( m_i \in M \) do
   2 Initialize: \( \mathcal{V}_{i}^{*} \) as \( \emptyset \), \( r \) as the root of \( \mathcal{V} \)
   3 while \( C_i(r) \neq \emptyset \) do
      4 \( r \leftarrow \arg\max_{y_k \in C_i(r)} s(u_i, v_k) \)
      5 if \( s(u_i, v_r) > \eta \) then
         6 Update the type-path: \( \mathcal{V}_{i}^{*} \leftarrow \mathcal{V}_{i}^{*} \cup \{r\} \)
      7 else
         8 return \( \mathcal{V}_{i}^{*} \) as the estimated type-path for \( m_i \)
   9 end
10 end
```

Diagram: Heterogeneous Partial-label Embedding

- Label embedding framework
- Framework Overview
- Sources, types, we further derive type correlation from two different text features for each mention to assist in modeling its true
- Carefully model mention-type associations, and extract a set
- Use both corpus-level statistics and KB facts to derive the relatedness between mentions and their candidate
- This problem as a
- Based on the incorrect type labels [7]. Our solution casts the
- Type sets in the training corpus contain "false" types,

3. LABEL NOISE REDUCTION

This section lays out the framework. As the candidate
- Non-goals.

2
- Perform joint embedding of the constructed graph
- Generate text features for each entity mention
- Perform top-down search in the candidate type sub-tree to estimate the correct type-path
## Experiment Setting

### Datasets:

1. **Wiki**: 1.5M sentences sampled from ~780k Wikipedia articles
2. **OntoNotes**: 13,109 news
3. **BBN**: 2,311 Wall Street Journal articles

<table>
<thead>
<tr>
<th>Data sets</th>
<th>Wiki</th>
<th>OntoNotes</th>
<th>BBN</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Types</td>
<td>113</td>
<td>89</td>
<td>47</td>
</tr>
<tr>
<td>#Documents</td>
<td>780,549</td>
<td>13,109</td>
<td>2,311</td>
</tr>
<tr>
<td>#Sentences</td>
<td>1.51M</td>
<td>143,709</td>
<td>48,899</td>
</tr>
<tr>
<td>#Training mentions</td>
<td>2.69M</td>
<td>223,342</td>
<td>109,090</td>
</tr>
<tr>
<td>#Ground-truth mentions</td>
<td>563</td>
<td>9,604</td>
<td>121,001</td>
</tr>
<tr>
<td>#Features</td>
<td>644,860</td>
<td>215,642</td>
<td>125,637</td>
</tr>
<tr>
<td>#Edges in graph</td>
<td>87M</td>
<td>5.9M</td>
<td>2.9M</td>
</tr>
</tbody>
</table>
Compared Methods

- **(1) Sib**: removes siblings types; **(2) Min**: removes types that appear only once in the document; **(3) All**: first performs Sib pruning then Min pruning;
- These three pruning strategies are combined with all baselines in the training stage.
- **(1) FIGER**: it adopts multi-label perceptron. **(2) HYENA**: it adopts hierarchical SVM. 
  - **(3) DeepWalk**: embedding a homogeneous graph with binary edges; **(4) LINE**: second-order LINE for feature-label bipartite graph; **(5) WSABIE**: adopts WARP loss for feature-label bipartite graph; **(6) PTE**: applied PTE joint training algorithm on subgraphs $G_{mf}$ and $G_{my}$. **(7) CNN**: applied convolutional neural network on mention’s local context. **(8) PL-SVM**: uses a margin-based loss to handle label noise. **(9) CLPL**: uses a linear model to encourage large average scores for candidate types.

- For AFET, we compare **(1)AFET**: adopts KB-based type correlation; **(2)AFET-CoH**: adopts hierarchy-based type correlation; **(3)AFET-NoCo**: does not consider type correlation.
## Performance Comparison on Fine-Grained Typing

<table>
<thead>
<tr>
<th>Typing Method</th>
<th>Wiki</th>
<th>OntoNotes</th>
<th>BBN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc</td>
<td>Ma-F1</td>
<td>Mi-F1</td>
</tr>
<tr>
<td>CLPL (Cour et al., 2011)</td>
<td>0.162</td>
<td>0.431</td>
<td>0.411</td>
</tr>
<tr>
<td>PL-SVM (Nguyen and Caruana, 2008)</td>
<td>0.428</td>
<td>0.613</td>
<td>0.571</td>
</tr>
<tr>
<td>FIGER (Ling and Weld, 2012)</td>
<td>0.474</td>
<td>0.692</td>
<td>0.655</td>
</tr>
<tr>
<td>FIGER-Min (Gillick et al., 2014)</td>
<td>0.453</td>
<td>0.691</td>
<td>0.631</td>
</tr>
<tr>
<td>HYENA (Yosef et al., 2012)</td>
<td>0.288</td>
<td>0.528</td>
<td>0.506</td>
</tr>
<tr>
<td>HYENA-Min</td>
<td>0.325</td>
<td>0.566</td>
<td>0.536</td>
</tr>
<tr>
<td>DeepWalk (Perozzi et al., 2014)</td>
<td>0.414</td>
<td>0.563</td>
<td>0.511</td>
</tr>
<tr>
<td>WSABIE (Yogatama et al., 2015)</td>
<td>0.373</td>
<td>0.565</td>
<td>0.521</td>
</tr>
<tr>
<td>LINE (Tang et al., 2015b)</td>
<td>0.181</td>
<td>0.480</td>
<td>0.499</td>
</tr>
<tr>
<td>PTE (Tang et al., 2015a)</td>
<td>0.405</td>
<td>0.575</td>
<td>0.526</td>
</tr>
<tr>
<td>AFET-noPartial</td>
<td>0.476</td>
<td>0.627</td>
<td>0.597</td>
</tr>
<tr>
<td>AFET-CorrH</td>
<td>0.433</td>
<td>0.583</td>
<td>0.551</td>
</tr>
<tr>
<td>AFET-EditDist</td>
<td>0.520</td>
<td>0.714</td>
<td>0.682</td>
</tr>
<tr>
<td>AFET</td>
<td>0.503</td>
<td>0.689</td>
<td>0.655</td>
</tr>
</tbody>
</table>
Case Study I
Case Study II
Conclusion
Thank You!

Questions?