USING RECURRENT NEURAL NETWORKS FOR DUPLICATE DETECTION AND ENTITY LINKING

BRUNO MARTINS, RUI SANTOS, RICARDO CUSTÓDIO
SEPTEMBER 20TH, 2016

GOLOCAL WORKSHOP
WHAT THIS TALK IS ABOUT...

Approximate string matching, duplicate detection, entity resolution, ...

• Given two names \( A \) and \( B \) (i.e., two sequences of characters \( A=<a_1,\ldots, a_n> \) and \( B=<b_1,\ldots, b_m> \)), decide if they describe the same real-world entity

• Extensive literature on the problem – see Cohen et al. (2003)

Used in many applications:

• Integrating data from multiple sources
• Matching queries in map-based services against a reference DB
• Part of systems for linking entity mentions in text to KB entries
  
  …
CLASSIC METHODS

Traditional approaches leverage string similarity methods plus a threshold for making decisions

- Character-based approaches
  - variations of edit distance, Jaro-Winkler similarity, …
- Vector space approaches
  - cosine similarity leveraging n-grams or skip-grams
- Hybrid approaches
  - 2-level scheme from Monge-Elkan

- Performance depends on the matching task
- Tuning similarity thresholds can be hard
METHODS BASED ON SUPERVISED LEARNING

1. Learn weights for edit-distance operations
2. Combine multiple similarity metrics
   • Labeled dataset with pairs of names
   • Each pair represented through a set of similarity scores
   • Train classifier to decide if a pair matches or not

   • Better performance than individual metrics
   • Hard to capture transliterations (e.g., different scripts)
   • Hard to capture phonetic substitutions
     • Existing phonetic coding methods (e.g., metaphone) are language dependent
PROPOSAL: LEVERAGE RECURRENT NEURAL NETS

- Similar approaches nowadays commonly used for inferring entailment between textual sentences
  - See previous work leveraging the Stanford Natural Language Inference Corpus

- Multi-layered architecture trained end-to-end
  - Each unicode string initially represented as a sequence of bytes
    - normalize unicode strings, then use a sequence of binary vectors (one-hot)
  - RNNs used for building compact representations of the strings
    - use Gated Recurrent Units (GRUs) in a bi-directional arrangement
  - Compact representations are combined
    - concatenation, vector difference, and element-wise product
  - MLP classifier leverages the combined representations
Gated Recurrent Units (GRUs) in a Bi-Directional Arrangement
(parameters are shared between the nodes that process the left and the right strings)

Second Layer of Gated Recurrent Units (GRUs) in a Bi-Directional Arrangement
(parameters are shared between the nodes that process the left and the right strings)

Combination of the representations produced through the GRUs:
- Concatenation
- Difference
- Element-wise product

Three feed-forward layers with Rectified Linear Units (RLUs)

Final output layer with a sigmoid activation

training with back-propagation and ADAI optimization procedure, minimizing the cross-entropy in the binary classification task
INITIAL EXPERIMENTS

• Matching place names
  • Used 5 million pairs of place names collected from geonames.org
  • Places from different countries, different types, in different scripts, …
  • Balanced dataset, 2-fold cross validation
  • Model details: dropout between each layer, Adam used for optimization, …

• Matching names for persons and organizations
  • Leveraging JRC-Names dataset with names for persons and organizations collected from news resources – Europe Media Monitor initiative
  • Same methodology, experiments still ongoing

• The names Jean-Claude Juncker, Jean Cloud Junker, Jean-Claude Juencker, Жан-Клод Юнкер, 詹 клод жюнкер, Zαν Κλοντ Γιούνκερ, 让-克洛德·容克, and many others have all been identified as referring to the current President of the European Commission
## Example: Results for Place Name Matching

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Damerau-Levenstein</td>
<td>0.77</td>
<td>0.43</td>
<td>0.55</td>
<td>0.65</td>
</tr>
<tr>
<td>Jaccard Similarity</td>
<td>0.75</td>
<td>0.36</td>
<td>0.49</td>
<td>0.62</td>
</tr>
<tr>
<td>Davis and De Salles (2007)</td>
<td>0.72</td>
<td>0.43</td>
<td>0.54</td>
<td>0.63</td>
</tr>
<tr>
<td>SVM</td>
<td>0.66</td>
<td>0.75</td>
<td>0.70</td>
<td>0.68</td>
</tr>
<tr>
<td>Random Forest</td>
<td>0.78</td>
<td>0.84</td>
<td>0.81</td>
<td>0.80</td>
</tr>
<tr>
<td>Gradient Boosting</td>
<td>0.73</td>
<td>0.79</td>
<td>0.76</td>
<td>0.75</td>
</tr>
<tr>
<td>Approach leveraging RNNs</td>
<td><strong>0.82</strong></td>
<td><strong>0.86</strong></td>
<td><strong>0.84</strong></td>
<td><strong>0.85</strong></td>
</tr>
</tbody>
</table>

- The RNN method takes approx. 4 days to train, but matching is fast...
  - Random Forest classifier takes 36 minutes per 50K records (*13 similarity metrics*)
  - Damerau-Levenshtein takes approx. 0.5 minutes per 50K records
  - RNN-based method takes approx. 1 minute per 50K records
WHAT NOW?

• Different model architectures and tuning results
  • E.g., LSTMs versus GRUs, consider attention mechanisms, ...
  • MSc student Rui Santos is working on this

• What if we consider modeling multiple attributes simultaneously (i.e., matching records):
  • Matching entries in different POI datasets
    • e.g., match data collected from FourSquare, Zomato, ...
    • Pre-train some of parts of the model (e.g., with JRC-Names)

• What if we consider modeling occurrence context:
  • Apply to toponym resolution (Ricardo Custódio is working on this)
  • Apply to (cross-lingual) entity linking in general
EXAMPLE: TOPONYM RESOLUTION IN TEXT

- Inputs correspond to pairs of sequences + latitude/longitude coordinates
  - Place name modeled as a sequence of bytes/characters
  - Surrounding context modeled as a sequence of words
  - Training data collected from Wikipedia
  - Pre-trained word embeddings

- RNNs for processing the sequences
- Concatenate representations

- Output layer predicts lat./lon.
- Optimize “great circle” distance
QUESTIONS?

BRUNO MARTINS
SEPTEMBER 20TH, 2016