Project Status Report, 10-707

Attribute & Relationship Extraction for Co-reference Chains

Bo Lin, Kevin Dela Rosa, Rushin Shah

Overview

As stated in our proposal, we plan to implement attribute and relation extraction modules that take chains produced by a within-document co-reference (WDC) system as input and output attributes for each chain and relationship labels for pairs of chains. We propose to use these modules to augment a cross-document co-reference (CDC) system and a cross-document visualization tool that we have built as part of our prior research.

In particular we are looking at performing the following tasks:

1. For each co-reference chain, extract as many attributes as possible, from a predefined list of attributes (Gender, Nationality, Occupation, Date-of-Birth, Birth Place)
2. For each pair of co-reference chains, extract a relationship label wherever possible.
3. Use attributes of chains as features in the CDC system's SVM classifier
4. Use relationship labels to augment cross-document visualization tool

For tasks 1 and 2, we plan to implement some standard attribute and relationship extraction algorithms. With respect attribute extraction, we plan to implement the methods described in [Garera & Yarowsky, 2009]. With respect to relationship extraction, we plan to implement a subsequence-based kernel, with the template described in [Bunescu & Mooney].

Once our attribute and relationship extraction algorithms have been implemented, we plan to augment our cross-document system in the ways implied by tasks 3 and 4. Due to a lack of data, we may not get evaluation numbers for tasks 3 and 4, but will do our best ensure that they are added to system by the end of the semester.

Data Sets

For training and attribute extraction modules, are using a set of NNDB.com seed pairs and corresponding Wikipedia pages.

For relationship extraction, we are using ACE Phase 2 top level relations.

For our CDC system, we are using the John Smith corpus, and WePS corpora.

Attribute Extraction Approach & Progress

We decided to implement the attribute extraction models described in [Garera & Yarowsky]. In short, those models are:

- **Contextual pattern-based model**: Take seed examples of (entity, attribute) and learn extraction patterns
- **Document position-based model**: Use typical document positions of attributes
- **Transitivity-based model**: Use attributes of neighboring entities
- **Latent wide-document-context model**: Use document-level topic models to infer attributes

We have collected an attribute extraction data set, making use of labels from NNDB.com for a set of people, and Wikipedia pages corresponding to those people. Currently our data set has attribute values for occupation, date-of-birth, and date-of-death, and we plan on collecting values for birthplace, gender, and nationality. We also wrote some code that processes the HTML from Wikipedia pages to extract the body text, and patterns to find candidate attribute values (for occupation and dates) in the text.

With respect to the different algorithms, we have implemented the first two mentioned, mostly from scratch and in Java.

Relationship Extraction Approach & Progress

We considered using different kinds of Kernels (subsequence, dependency trees, etc). From the literature, we came to the conclusion that subsequence kernels achieve quite good performance, and don't require significant NLP resources such as parsers, etc. Consequently, they are not only faster to calculate but also more robust to ill-formed documents. We therefore chose to implement a subsequence-based kernel for relation extraction, and as a template we chose to follow the paper by Bunescu & Mooney, presented by Dr. Cohen in class.

We realized that although the basic idea presented in the paper is quite easy to understand, implementing a kernel for relation extraction is quite a complex task. We present a brief summary of our major steps forward for this task:

- We decided to use LibSVM for our Support Vector Machine implementation. Bunescu has made a custom version that allows the use of custom kernel functions, but it is based on an older release of the package. Meanwhile, the current version supports many optimizations and *precomputed* kernels. We created a custom version of LibSVM based on the current version, but with the addition of Bunescu's changes. In addition, although his changes allow the use of custom functions, we still needed to write additional code to allow for richer representation of instances than the default vectors of numbers. (in our case, sentences with named entities).

- We spent the bulk of our time on this part of the project working on feature extraction. We started with XML files from the ACE dataset, and extracted sentences with named entity and

relationship labels. We added POS tags to the words (as an additional feature dimension), and we produced instances for the SVM, where each instance consists of:
    - The 2 Named Entities of interest
    - Sentence fragments before, between and after NEs
- If a sentence consists of more than 2 NEs, make ${}_{N}C_{2}$ copies of the sentence and create an instance from each copy (With the appropriate NEs and relationship labels).

We then implemented general subsequence Kernel function. We are currently working on adapting this for the specific case of *relationship Kernels* (as explained in Bunescu & Mooney).

Future Plans

With respect to attribute extraction, we need to finish implementing the remaining models from [Garera & Yarowsky]. We also need to add a few more attributes to our data set, currently we have ground truth attribute values for occupation, date-of-birth, and date-of-death; we'd like to have gender, nationality, and birthplace as well. Once these algorithms have been implemented we need to run experiments to compare them against some baseline. Then to address task 3 we will need to extend the code to work with chains, and use attributes extracted from code base as features in the CDC system's SVM classifier to do better co-reference

With respect to relationship extraction, once we have implemented the relationship kernel successfully, we will extend this code (which is currently designed for sentences), to *chains*, so it can be plugged into our CDC system.