An End-to-end Supervised Target-Word Sense Disambiguation System

Mahesh Joshi[1] **Serguei Pakhomov**[2] **Ted Pedersen**[1] **Richard Maclin**[1] **Christopher Chute**[2]

[1]Department of Computer Science
University of Minnesota
Duluth, MN, USA 55812
{joshi031, tpederse, rmaclin}@d.umn.edu

[2]Division of Biomedical Informatics
Mayo College of Medicine
Rochester, MN, USA 55905
{Pakhomov.Serguei, chute}@mayo.edu

Abstract

We present an extensible supervised Target-Word Sense Disambiguation system that leverages upon GATE (General Architecture for Text Engineering), NSP (Ngram Statistics Package) and WEKA (Waikato Environment for Knowledge Analysis) to present an end-to-end solution that integrates feature identification, feature extraction, preprocessing and classification.

Introduction

Word Sense Disambiguation (WSD) is the task of automatically deciding the sense of an ambiguous word based on its surrounding context. The correct sense is usually chosen from a predefined set of senses, known as the *sense inventory*. In *target-word sense disambiguation* the scope is limited to assigning meaning to occurrences of a few predefined *target* words in the given corpus of text.

Most popular approaches to WSD use supervised machine learning methods to train a classifier using a set of labeled instances of the ambiguous word and create a statistical model. This model is then applied to unlabeled instances of the ambiguous word to decide their correct sense. In such approaches, the ability to run several experiments based on the choice of (i) features; and (ii) the classifier along with its parameters, is the key factor in determining the configuration that yields the best accuracy for the task under consideration. This is exactly what our system facilitates - an end-to-end interface for running several WSD experiments, with the choice of features using many existing and **one new** GATE (Cunningham *et al.* 2002) component and the choice of classifiers from WEKA (Witten & Frank 2005).

Background and Related Systems

The GATE framework for text engineering and analysis provides several components that identify and annotate features such as Part-of-Speech tags, named entities and syntactic features including noun phrases and verb phrases. In addition, it also provides components for using machine learning methods for text analysis and for integrating WEKA classifiers in GATE. However, being a general purpose system by design, there are certain limitations in the GATE framework that prevent it from being used as an off-the-shelf end-to-end WSD system. First, we found the feature identification method provided by the machine learning component in GATE to be restrictive in two ways: (i) In a target-word scenario, it does not allow extraction of features from annotations that do not surround the target word; (ii) it does not include extraction of *floating* features – where one might be interested in extracting a certain feature or sets of features which are at a *variable distance* from the target word. Second, from the machine learning perspective - the framework does not *integrate* abilities such as: (i) generation of nominal features from string-valued features, (ii) cross-validation experiments and (iii) automation of train-test experiments. Our system builds upon the GATE framework by addressing these limitations.

Other publicly available supervised target-word sense disambiguation systems include SenseTools[1] (Pedersen 2001), SyntaLex[2] (Mohammad & Pedersen 2004) and WSD Shell[3]. SenseTools and WSD Shell (which is built on top of SenseTools) are a set of modular Perl programs that integrate feature identification using the Ngram Statistics Package (NSP) (Banerjee & Pedersen 2003) and machine learning using WEKA. SyntaLex adds Part-of-Speech and syntactic features to the SenseTools system. The only limitation that these systems have is the lack of off-the-shelf components for: (i) integration with other knowledge sources such as WordNet and (ii) using features such as named entities, coreference resolutions and morphological forms. Our system benefits from the extensive set of components that are already available in GATE.

System Description

Figure 1 shows the architecture of our system. The system provides a graphical user interface (GUI) through GATE for running individual experiments and a command line interface (CLI) (using a Perl script and Java wrapper class) for running experiments in batch mode. The round-cornered rectangles in the diagram represent the components of our system and others are the input/output files of these components. The components and files enclosed in the dotted line

[1]http://www.d.umn.edu/∼tpederse/sensetools.html
[2]http://www.d.umn.edu/∼tpederse/syntalex.html
[3]http://www.d.umn.edu/∼tpederse/wsdshell.html

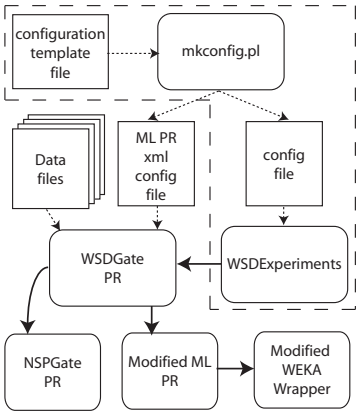

Figure 1: System Architecture

are used only in the CLI mode. Dotted arrows indicate the data flow and solid arrows show the control flow. Details regarding each component are as follows:

WSDGate PR: A text processing component in GATE is a *Processing Resource* (PR). At the heart of our system is the WSDGate PR which coordinates the control and data flow between NSPGate PR, a modified version of the Machine Learning PR (ML PR) from GATE, a modified version of the WEKA Wrapper from GATE and the Tokenizer, Sentence-Splitter and Part-of-Speech Tagger PRs (not shown) from the ANNIE sub-system in GATE.

NSPGate PR: This is a wrapper for NSP, which we have developed as a PR in GATE. It provides the functionality of invoking NSP with the required options and annotating identified ngrams in the input document.

Modified ML PR: This is a modified version of the Machine Learning PR that exists in GATE. Its role is to identify and extract features from the input document. Our modifications to this PR overcome all of the GATE limitations for end-to-end WSD that we have discussed earlier.

Modified WEKA Wrapper: This is a modified version of the WEKA Wrapper component in GATE, which is a helper component of the ML PR and handles operations and integration specific to WEKA. Our modifications to this involve adapting it to the changes in ML PR and changes to support the scenario where we need just dataset generation in the WEKA ARFF data format, without having to necessarily run some WEKA classifier.

WSDExperiments: This is a Java class which facilitates command line usage of WSDGate PR, using a configuration file to specify the parameters required by WSDGate PR.

mkconfig.pl: This is a Perl script which facilitates setting up batch mode experiments for multiple ambiguous words, with desired features and WEKA classifiers. One of its important functions is automatic generation of XML feature configuration files that are required by the ML PR component. These XML feature configuration files are also required during the GUI usage and mkconfig.pl can be used to generate them for the GUI as well as for CLI usage.

Advantages and Future Improvements

There are three advantages to our system. The first advantage is the integration of an existing robust NLP framework in the from GATE, feature selection tool in the form of NSP and the widely used WEKA data mining suite. GATE provides extreme flexibility in input data format and choice of a wide range of features, NSP adds sophisticated feature selection criteria for ngrams and WEKA offers several data preprocessing options and machine learning algorithms. The second advantage is a streamlined CLI to set up experiments in batch mode. The final advantage is that although we have used the system only for the tasks of WSD and automatic expansion of ambiguous acronyms, nothing in its architecture prevents it from being used for any task that can be cast as a supervised classification problem, such as document classification or semantic role labeling.

We are currently working on three improvements to our system. First, as of now, all NSP features are not fully supported by the NSPGate PR. Specifically, we would like NSPGate to support non-contiguous ngrams (with one or more intermediate words) as opposed to only contiguous ngrams. Second, although GATE provides several feature identification components, WSDGate directly utilizes only the ANNIE system for Part-of-Speech tags. We would like to enable the user to choose any GATE PR for feature identification using the WSDGate interface itself. Third, we believe we can further improve the ease-of-use and automation in the system by providing a web interface for it.

Note: WSDGate and NSPGate are available as SourceForge projects at http://sourceforge.net/projects/wsdgate and http://sourceforge.net/projects/nspgate respectively and have been used extensively for experimentation.

Acknowledgments

We would like to thank Julien Nioche from the GATE team for his technical advice on the ML PR modifications.

References

Banerjee, S., and Pedersen, T. 2003. The Design, Implementation and Use of the Ngram Statistics Package. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*.

Cunningham, H.; Maynard, D.; Bontcheva, K.; and Tablan, V. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, 168–175.

Mohammad, S., and Pedersen, T. 2004. Complementarity of Lexical and Simple Syntactic Features: The SyntaLex Approach to SENSEVAL-3. In *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (*SENSEVAL-3*)*.

Pedersen, T. 2001. Machine Learning with Lexical Features: The Duluth Approach to SENSEVAL-2. In *Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (*SENSEVAL-2*)*.

Witten, I., and Frank, E. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.