HebrewNER-MWE

# Purpose

The purpose of the system is to classify lexical tokens in Hebrew sentences according to their named-entity: "יוסי" is a person, "אינטל" is an organization, "חיפה" is a location, etc.

# Input-Output

The input is a document written in Hebrew. The output is a list of (word, entity) pairs, matching the original sentence. To this end, the system is using information given from a Part-of-Speech tagger.

# Method

The system is using three independent classifying algorithms:

1. Baseline – a simple pattern-based classifier.
2. HMM – An anlysis using Hidden Markov Model and Viterbi algorithm.
3. MaxEnt model (Multinomial logistic regression).

The results from these methods are merged according to some heuristic rules.

# Origins

This system is an adoption of a system developed by Naama Ben-Mordechai for her master's degree thesis. The original system is accessible at <http://www.cs.bgu.ac.il/~nlpproj/naama/>. The main difference is the adoption of the MWE system as the Part-of-Speech tagger <http://www.mila.cs.technion.ac.il/mila/eng/index.html>

Other changes include

* Porting to Java 8: Better type safety, better resource management, and cleaner syntax
* Massive redesign of class structure
* Using a newer version of the open-nlp library
* Massive refactoring. In particular, decoupling the POS-tagger from the named-entity tagger
* A basic framework for evaluating performance
* Intergration with Maven

# Terminology:

* **Entity**: A named Entity. One of **PERS**, **LOC**, **ORG**, **MONEY**, **TIME**, **PERCENT**, **DATE**, **O**.
  + **O**  means "Not an entity"
* **Surface**: the word as it appears in the raw text. Example: הביתה
* **Lemma** (or **base**): The base form of the surface. Example: בית
* **POS**: Part Of Speech. The possible values depends on the pos-tagger. Examples can be found in the appropriate pos\_list.txt file.
* **Tagged** or **POS-Tagged** means "Part of Speech tagged"
* **MWE:** Technion's POS tagger
* **Construct** means סמיכות. The opposite of a construct called **absolute.**
* **Model**: A method for finding entities. One of HMM, Maxent, Baseline

# Important Classes:

* **Token**: The basic information unit about a surface.
* **Sentence**: A list of tokens.
* **EntityFinder**: The main interface for entity recognition. Implemented by **HmmEntityFinder**, **BaselineEntityFinder**, **MaxentEntityFinder**, and **MergeEntityFinder** which is the default finder.
* **ModelTrainer**: An interface for training the above NameFinders.

# Usage:

# Directory structure:

There are 3 corpora for testing and evaluation purposes:

* Original: 3 test corpuses ("500", "nrg-3", "test") from the original project.
* Haaretz: A corpus based on MWE's corpus for Haaretz. The Entities in this corpus are almost a direct interpretation of the original MWE entity recognition, and thus have little value for evalutaion purposes.
* Gold: 4 hand-tagged documents from ynet.
* Default: Original training corpora
* Cross-Validation: a 1/5 of the original training corpus.

**docs/**: Documentation

* **README.docx**, **README.pdf**: Original documentations for HebrewNER
* **Evaluations/**: Excel files documenting some statistical evaluations.

Under **resources/**:

* **lists/**: dictionaries for Dictionary class.
  + **README.txt** : explanation for each dictionary file
* **models/** : .model files
  + **baseline**.model
  + **hmm**.model
  + **maxent**.model
* **trainset/** :
  + **txt/** : .txt files
    - A subfolder for each corpus with a plaintext version or the corpus.
  + **entities/**: .entity files
  + **tagged/**: .tagged files
  + **full/**: .full files. Combination of entity and tagged
  + **xml/**: .xml version, straight from the tagger
  + **pos\_list.txt**: a list of all possible Part-Of-Speech. It is updated from inside **class Sentence**.

## File formats:

* **pos\_list.txt**: line per value
* **.txt**: raw documents
* **.entity**: tab-seperated <Surface, Entity> pairs, grouped in newline-seperated sentences.
* **.tagged**: tab-seperated values extracted from the xml. Matching class: **TokenEntity**.
* **.full**: combination of **.entity** and **.tagged**.
* **.model**: A model-specific representation file. These files are the result of **ModelTrainer.train()** for each subclass.