Computational Linguistics Group
Department of Computer Science
University of Haifa

בלשנות חישובית
החוג למדעי המחשב
אוניברסיטת חיפה

Laboratory in Natural Language Processing

Shuly Wintner

Semester B, 2009: Wednesday, 16:00–19:00

`http://cs.haifa.ac.il/~shuly/teaching/09/lab/`

1 Objectives

The Lab offers a number of practical projects in Natural Language Processing (NLP), focusing on (but not limited to) processing of Hebrew. Some projects require previous knowledge of computational linguistics but some assume no previous background. All projects (except one) involve programming: the end result is a relatively large-scale, well-documented and efficient software package. Some of the projects may involve also some research (e.g., reading a research paper and implementing its ideas).

2 Administration

Projects are to be implemented by groups of at most two students. All systems will be presented at the end of the semester for a final demo. A coordination meeting is planned for Wednesday, June 24th; all work must be completed by Monday, August 31st.

The programming language must be portable enough to be usable on a variety of platforms; Java and Python are recommended, C++ or Perl will be tolerated, if you have a different language in mind discuss it with the instructor. Most projects will have to be executed in a Linux environment due to dependencies on external packages.

Grading will be based on comprehension of the problem, quality of the implementation and quality of the documentation. In particular, the final grade will be based on: Comprehension of the problem (and the accompanying paper(s), where applicable); Full implementation of a working solution; Presentation of a final working system; Comprehensive documentation.

3 List of projects

3.1 Collecting a Hebrew-English bilingual corpus

No background in NLP is required

Text corpora are among the most important resources for a variety of NLP applications. They are used to provide word frequency counts for statistical NLP and information retrieval applications such as part-of-speech taggers, shallow parsers, categorization and summarization, to list just a few. Collecting corpora, representing and maintaining them are non-trivial tasks. The objective of this project is to build a parallel corpus of Hebrew and English documents by crawling the web. The documents in the corpus will then be sentence- and word-aligned. The parallel corpus will be used to train a statistical machine translation system.

You will develop software for collecting Hebrew-English corpora. The main technique is web-crawling: a program which crawls the web and searches for relevant documents. Search will be focused on a number of dynamic web sites which are known to have similar documents in the two languages (e.g., some newspapers), but in principle the technology should work for the entire Hebrew web. The main task is then determining whether two documents are indeed possible translations, and you will use some of the techniques reported in the literature (Resnik and Smith, 2003). You will experiment with a number of different methods for determining whether two given documents are indeed translations of each other. Finally, you will have to develop a storage solution for the collected corpora, such that new documents can be easily added at any time.

Specific tasks in this project include:

1. Implementing a web-crawler and defining the domains to search for documents. You can use publicly-available packages for this sub-task.

2. "Cleaning" web pages and producing text from the HTML. You can use publicly-available packages for this sub-task.

3. Determining whether two documents are translations. You will likely want to normalize the words in the documents. This can be achieved by *stemming* on the English side, and full morphological analysis and disambiguation on the Hebrew side. You will use available tools for these sub-tasks. You will also be able to use an available Hebrew-English translation dictionary. You will, however, have to design and implement a scoring function that determines the distance between documents in a reliable way.

4. Storing the retrieved documents in an organized way.

5. Evaluation of the performance of your system.

In order to be useful, your crawler must be designed in a way that will easily facilitate future changes, in case the format of the HTML documents in the web sites you crawl changes.

3.2 Collecting a Hebrew-Arabic bilingual corpus

No background in NLP is required

Same as above, for Hebrew-Arabic.

3.3 Sentence-alignment of a Hebrew-English bilingual corpus

No background in NLP is required

Given a parallel corpus, it is useful to align it such that each sentence in L1 corresponds to zero or more sentences in L2 which represent its translation. In this project you will implement the sentence-level alignment algorithm of Gale and Church (1993) and apply it to the texts in the corpus. The expected outcome of the project is a tool for inducing a bilingual Hebrew-English dictionary, as well as a phrase-translation table that is an invaluable resource for training statistical machine translation systems.

The algorithm is fairly straight-forward, and you will be required to implement it and evaluate its performance. Then, you will be required to introduce improvements that utilize an existing word translation dictionary, and evaluate the revised algorithm. Evaluation will be run on the texts of `opensubtitles.org`, as well as on additional (sentence aligned) data that will be provided to you. Eventually, you will have to verify that the system you develop can reliably sentence-align the corpora developed by the previous projects.

3.4 Building a Hebrew to English statistical machine translation system

Introduction to Computational Linguistics recommended but not mandatory.

The goal of this project is to develop a default, low-resources statistical machine translation system for Hebrew to English. Such a system can be constructed by combining two probabilistic models: a *translation model* which determines the probability of an English "phrase" (sequence of one or more words) given a Hebrew "phrase"; and a *target language model* which determines the probability of English sentences (Brown et al., 1990). The parameters of translation models are estimated from sentence- and word-aligned parallel corpora, and the parameters of language models are estimated from large monolingual corpora. In this project you will use the sentence-aligned parallel corpora developed by projects 3.1 and 3.3 for training the translation model (initially, you will use `opensubtitles.org` as a parallel corpus). An English language model will be provided to you.

For word alignment, you will compare the performance of three available packages: GIZA++ (Och and Ney, 2003) (`http://www.fjoch.com/GIZA++.html`), Berkeley Word Aligner (Liang et al., 2006) (`http://nlp.cs.berkeley.edu/Main.html#WordAligner`) and PostCAT (Ganchev et al., 2008) (`http://www.seas.upenn.edu/~strctlrn/CAT/CAT.html`). The idea is to create an experimental setup in which the alignments produced by each of these packages can be plugged into an MT system, so that the quality of the MT system reflects the quality of the alignments.

In addition, you will compare the quality of alignments based on surface forms (the tokens of the raw data) with those obtained after morphological processing. Specifically, the Hebrew side of the corpus will be morphologically analyzed and disambiguated, and the alignments will link English words to Hebrew lemmas (base forms).

The translation tables that will be extracted from the word-aligned corpora will be used by a generic *decoder* to produce a basic Hebrew-English MT system. You will experiment with two decoders: Stat-XFER (Lavie, 2008) and the more ubiquitous, publicly-available Moses (Koehn et al., 2007) (`http://www.statmt.org/moses/`). A small set of Hebrew sentences, along with reference translations to English, will be provided to you and will be used for evaluation of the quality of the translations produced by your system.

3.5 A generic transliteration system

Introduction to Computational Linguistics recommended but not mandatory.

When texts are translated from one language to another, some words are not translated; rather, they are *transliterated*: rendered in the writing system of the target language in a way that retains or approximate the original pronunciation of the word. Transliterated words are frequently proper names or loan words. For example, when the Hebrew sentence **0: 3** ספרד הביסה את ברזיל is translated to English, the proper name ספרד is translated to *Spain*, but the proper name ברזיל is transliterated as *Brazil*.

You will develop a generic system for transliterating words in a large number of languages to English, following the ideas of Kirschenbaum and Wintner (2009). Transliteration will be based on statistical machine translation (Brown et al., 1990), in which the translation model maps characters in the source language to characters in English, and the language model is a unigram English word model (viewed as a character n-gram model). The language model will be provided to you. The translation model will be extracted from multilingual titles of Wikipedia documents.

In order to create a translation model for a given source language, you will have to extract from Wikipedia all the titles of the articles that occur both in the source language and in English, and to determine whether these titles are translations or transliterations. This can be done by comparing the characters in the title terms, given some possible mappings of characters from the source to English. For example, the Hebrew-English mapping will include the pairs ב–*b*, ב–*v*, פ–*p*, פ–*f*, ס–*s*, ר–*r*, ד–*d*, ז–*z*, ל–*l*. Based on such mapping, you will be able to determine that ברזיל–*Brazil* is a transliterated pair, whereas ספרד–*Spain* is not. You will have to prepare such character mapping tables for a few languages.

Variant: a more generic system will allow transliteration to *any* language. Two additional resources will be required:

- a monolingual (target) language model: you will use the monolingual projection of Wikipedia on the target language to create such a language model.
- a mapping of characters between the source and target languages: you will have to provide such mappings for a few language pairs.

3.6 Translation of numeric, time and date expressions

Introduction to Computational Linguistics required.

Numeric expressions, date and time expressions are ubiquitous in natural language texts and typically have an idiosyncratic, irregular grammar. However, since they are constructed with few and well-defined lexical items and constructions, they can be relatively easily identified. In this project you will design and implement a system for identifying such expressions in Hebrew texts and translating them to English.

Your main tasks will be:

1. Define a small representative corpus of Hebrew texts to serve as the development set.
2. Identify all the numeric, date and time expressions in the corpus and annotate them accordingly.
3. Use insights from the development set to define a regular grammar of such expressions.
4. Implement the grammar using some (extended) regular-expression language, such that each expression is mapped to a unique representation. Implement a generator that converts this representation to English.
5. Create a stand-alone system whose input is Hebrew text (represented in XML) and whose output is the same text, in the same format, where numeric, date and time expressions are annotated.

The dictionary and generator that you will develop for this project must be external to the code, so that they can be easily replaced by modules for some other language (e.g., Hebrew-Arabic).

3.7 A classifier for Translationese

Introduction to Computational Linguistics recommended but not mandatory.

Translated texts are known to have linguistic properties that set them apart from text written originally in the target language. Given the same domain and genre, translated texts tend to have shorter sentences, lower type/token ratio (i.e., less rich language), more limited syntactic constructions, etc. In this project you will use machine learning techniques to construct a classifier that can distinguish between translated and original texts in English, following Baroni and Bernardini (2006).

You will be provided with a training corpus consisting of newspaper articles in a single domain in English. The articles will be tagged as either translated (from Hebrew) or original. Your main task will be to define a set of distinctive features and implement the feature extractor. Features may include superficial characteristics, such as the average length of sentences or the type/token ratio in a document; n-gram features, such as unigrams of function words, or specific bigrams or trigrams; or more linguistically-informed features, such as n-grams of part-of-speech tags, ratio of active to passive verbs, complexity of syntactic structures, etc. You will be able to use off-the-shelf

tools for processing the corpus, and publicly-available machine learning packages (such as SVM, SNoW or Memory-based Learning).

Once the feature extractor is implemented, you will train a classifier on the training material and conduct a robust evaluation of the results. The result of this project will be used in a research on selecting the best language models for machine translation.

3.8 Morphology-aware search engine for Hebrew

Introduction to Computational Linguistics recommended but not mandatory.

Existing search engines are text-oriented: they index documents by *tokens*, which are usually defined as space-delimited strings. However, natural language texts are not just collections of tokens, since words in natural languages are formed through certain well-understood *morphological* processes, and are hence related to each other in ways which can contribute to better search results.

In this project you will create morphologically-aware search capabilities: given query terms, you will generate all their possible inflections and submit those as a disjunctive query. In addition, you will use a bilingual dictionary to translate query terms from Hebrew to English and retrieve also relevant documents in English. For example, if the user entered the query term מקלדת, the term will be expanded and the inflected forms המקלדת, מקלדות, המקלדות etc. will be generated. All those terms will be presented to the search engine as a disjunction. The user will also have the ability to specify that translated terms should be generated. In this case, the term *keyboard* will be generated and presented to the search engine.

The morphological generator (which produces all the inflected forms of a given lexeme) and the bilingual dictionary will be provided to you (Itai and Wintner, 2008), although you may have to adapt them for this task. Your main task will be to implement a web-based interface to (at least two) search engines that will interface with these existing resources to provide the additional capability; and to evaluate the contribution of your work.

3.9 Morphological analysis of dotted Hebrew

Introduction to Computational Linguistics recommended but not mandatory. As you will be revising an existing Java code, knowledge of Java is mandatory.

Morphological analysis is the process of determining the base (also known as *lexeme*, or *lemma*) of a word, along with its morphological attributes. An example of the morphological analysis of a simple Hebrew sentence is depicted in Figure 1.

Hebrew has a complex morphology and hence the design of a morphological analyzer for the language is a complex task. We currently have a large-scale and relatively accurate morphological system for Hebrew (Yona and Wintner, 2008) which works for *undotted* texts. In this project you will create a variant of the morphological system for the *dotted* script.

The main task here is to understand the morphological rules that apply to words, as stipulated for the undotted case, and then revise and refine them for the dotted case. The greatest benefit of such a system is that it will facilitate, in conjunction with a morphological disambiguation system which is currently under development, an automatic vocalization of undotted texts.

```
הרכבת   [+noun][+id]18182[+undotted]הרכבה[+transliterated]hrkbh[+gender]+feminine
        [+number]+singular[+script]+formal[+construct]+true
הרכבת   [+verb][+id]19729[+undotted]הרכיב[+transliterated]hrkib[+root]רכב[+binyan]+Hif'il
        [+person/gender/number]+2p/M/Sg[+script]+formal[+tense]+past
הרכבת   [+verb][+id]19729[+undotted]הרכיב[+transliterated]hrkib[+root]רכב[+binyan]+Hif'il
        [+person/gender/number]+2p/F/Sg[+script]+formal[+tense]+past
הרכבת   [+defArt]ה[+noun][+id]18975[+undotted]רכבת[+transliterated]rkbt[+gender]+feminine
        [+number]+singular[+script]+formal[+construct]+false

שבתה    [+noun][+id]17280[+undotted]שבת[+transliterated]ebt[+gender]+feminine
        [+number]+singular[+script]+formal[+construct]+false[+possessiveSuffix]+3p/F/Sg
שבתה    [+verb][+id]9430[+undotted]שבת[+transliterated]ebt[+root]שבת[+binyan]+Pa'al
        [+person/gender/number]+3p/F/Sg[+script]+formal[+tense]+past
שבתה    [+verb][+id]1541[+undotted]שבה[+transliterated]ebh[+root]שבה[+binyan]+Pa'al
        [+person/gender/number]+3p/F/Sg[+script]+formal[+tense]+past
שבתה    [+subord]ש[+preposition]ב[+noun][+id]19804[+undotted]תה[+transliterated]th
        [+gender]+masculine[+number]+singular[+script]+formal[+construct]+true
שבתה    [+subord]ש[+preposition]ב[+noun][+id]19804[+undotted]תה[+transliterated]th
        [+gender]+masculine[+number]+singular[+script]+formal[+construct]+false
שבתה    [+subord]ש[+preposition]ב[+defArt][+noun][+id]19804[+undotted]תה[+transliterated]th
        [+gender]+masculine[+number]+singular[+script]+formal[+construct]+false
שבתה    [+subord]ש[+noun][+id]19130[+undotted]בתה[+transliterated]bth[+gender]+feminine
        [+number]+singular[+script]+formal[+construct]+false
שבתה    [+subord]ש[+noun][+id]1379[+undotted]בת[+transliterated]bt[+gender]+feminine
        [+number]+singular[+script]+formal[+construct]+false[+possessiveSuffix]+3p/F/Sg

אתמול   [+adverb][+id]12448[+undotted]אתמול[+transliterated]atmwl
```

Figure 1: Example morphological analysis

3.10 Unification Grammars

Introduction to Computational Linguistics required.

This is a very different kind of project. You will be required to read and fully understand a textbook on Unification Grammars. Your main task will be to fully solve numerous exercises scattered throughout the text. Some of the exercises are technical and easy, some require more thought.

References

Marco Baroni and Silvia Bernardini. A new approach to the study of Translationese: Machine-learning the difference between original and translated text. *Literary and Linguistic Computing*, 21(3):259–274, September 2006.

P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16 (2):79–85, 1990.

William A. Gale and Kenneth W. Church. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102, 1993. ISSN 0891-2017.

Kuzman Ganchev, Joâo de Almeida Varelas Graça, and Ben Taskar. Better alignments = better translations? In *Proceedings of ACL-08: HLT*, pages 986–993. Association for Computational Linguistics, June 2008.

Alon Itai and Shuly Wintner. Language resources for Hebrew. *Language Resources and Evaluation*, 42:75–98, March 2008.

Amit Kirschenbaum and Shuly Wintner. Minimally supervised transliteration for machine translation. In *Proceedings of The 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL-09)*, April 2009.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. The Association for Computational Linguistics, June 2007.

Alon Lavie. Stat-xfer: A general search-based syntax-driven framework for machine translation. In Alexander F. Gelbukh, editor, *CICLing*, volume 4919 of *Lecture Notes in Computer Science*, pages 362–375. Springer, 2008. ISBN 978-3-540-78134-9.

Percy Liang, Ben Taskar, and Dan Klein. Alignment by agreement. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 104–111, New York City, USA, June 2006. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N/N06/N06-1014`.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Philip Resnik and Noah A. Smith. The web as a parallel corpus. *Comput. Linguist.*, 29(3):349–380, 2003. ISSN 0891-2017. doi: http://dx.doi.org/10.1162/089120103322711578.

Shlomo Yona and Shuly Wintner. A finite-state morphological grammar of Hebrew. *Natural Language Engineering*, 14(2):173–190, April 2008.