*Computational Linguistics Group*
*Department of Computer Science*
*University of Haifa*

*בלשנות חישובית*
*החוג למדעי המחשב*
*אוניברסיטת חיפה*

# Laboratory in Natural Language Processing

Shuly Wintner, shuly@cs.haifa.ac.il

Semester A, 2012-13: Monday, 13:00–16:00

## 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 and NLP but some assume no previous background. All projects 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). In fact, some projects may evolve into graduate-level research.

## 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 Monday, 17.12.12; all work must be completed by the final (presentation) meeting which will be held on Monday, 4.3.13.

The programming language must be portable enough to be usable on a variety of platforms; Python is highly recommended, C++, Perl or Java will be tolerated, if you have a different language in mind discuss it with the instructor. Some projects may 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 Word frequencies in natural texts and Zipf's law

**Introduction to NLP highly recommended.**

The distribution of word frequencies in human generated texts is characterized by what has become to be known as Zipf's law. The main observation is that any given text consists of few highly frequent words, a certain amount of words with medium frequencies and quite a few with extremely low frequencies. Formally, let $w$ be a word, $f(w)$ the frequency of $w$ in some corpus and $r(w)$ the rank of $w$, that is, the place of $w$ in a list of words ordered by frequency (such that the most frequent word is ranked 1st). Then

$$f(w) = c/r(w)^a$$

where $c$ and $a$ are constants, free parameters whose values depend on the text. The constant $a$ is said to be typically around 1, and therefore $c$ approximately equals the frequency of the highest ranked word. Also, this law seems to apply for many distributions characterized by a few "giants" and many "dwarfs", such as population size in cities within a country, or incomes within a population. For a general review, see Baroni (2008).

Several open questions revolve around this phenomenon/law:

1. How much does it depend on the text size?
2. Does it apply in the same way to different writing systems?
3. Does the same function hold for all the ranges of the curve?

In this project you will be asked to conduct several basic tasks of text processing, in an attempt to address some of these questions. You will be provided with several texts (corpora) in various languages, written in a variety of writing systems. You will develop a software package that will:

1. Tokenize the texts. At a minimum, you will have to develop tokenization solutions for three different writing systems: alphabet (e.g., English), Abjad (e.g., Hebrew, dotted and non-dotted), and logography (e.g., Mandarin Chinese).
2. Perform morphological analysis and disambiguation. Where available, you will incorporate morphological processing in your code, in order to reduce words to their base forms.
3. Produce for each tokenized, morphologically-processed text a list of the words, their ranks and frequencies.
4. Plot the curve of a word frequency as a function of its rank.
5. Run a regression line to check how the function fits the data.

Ideally, the project should be written in Python. The use of NLTK can significantly reduce your programming load.

This work is part of a research in progress, and — if successful — may lead to a graduate thesis involving issues in graph algorithms and information theory.

## 3.2 Cross-classification of translationese

**Introduction to NLP recommended but not mandatory.**

Translated texts are known to have linguistic properties that set them apart from texts 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. Several works use these differences as features that inform classifiers, which can then distinguish between original and translated texts (Baroni and Bernardini, 2006; van Halteren, 2008; Ilisei et al., 2010; Koppel and Ordan, 2011; Volansky et al., 2012). The problem with such classifiers, however, is that they tend to be highly dependent on the specific corpus they are trained on.

In this project you will explore the features that can *robustly* distinguish between original and translated texts, even across domains, genres and datasets. You will be provided with two corpora: a training corpus consisting of newspaper articles in a single domain in English; and a test corpus consisting of the European Parliament proceedings. The texts will be tagged as either translated (from several different languages) 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 for implementing the classifier. In particular, you will be expected to implement (at least some of the better) features introduced by Volansky et al. (2012).

Once the feature extractor is implemented, you will train a classifier on the training material and test it on the test corpus, conducting a robust evaluation of the results.

## 3.3 Distinguishing between human and machine translation

**Introduction to NLP recommended but not mandatory.**

Consider the following texts:

> Britain has amended a law that allowed for issuing arrest warrants against Israeli politicians who visit the country, British Ambassador Matthew Gould announced Thursday. Gould called opposition leader Tzipi Livni, against whom an arrest warrant was issued in 2009, and told her the Queen has signed the amendment "to ensure that the UK's justice system can no longer be abused for political reasons."

> British queen has signed today (Thursday) on an amendment to reform the police and social responsibility, to prevent submission of arrest warrants against senior Israeli officials in Britain. Ends legislative amendment process that began following the arrest order was issued against the opposition chairwoman, Tzipi Livni.

Both of them were translated from Hebrew to English; can you tell which one was translated by a human and which one by machine translation?

You will develop a classifier that can distinguish human from machine translated texts. 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 human translated or machine translated. 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 for implementing the classifier. You will be able to base your work on a similar effort whose goal was to distinguish between translated texts and original ones (Volansky et al., 2012).

Once the feature extractor is implemented, you will train a classifier on the training material and conduct a robust evaluation of the results. A specific goal would be to accurately distinguish between machine translation outputs and other types of texts given very small samples (e.g., a few sentences only).

## 3.4 The features of translation

**Introduction to NLP recommended but not mandatory.**

Translated texts are known to have linguistic properties that set them apart from texts 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. Several works use these differences as features that inform classifiers, which can then distinguish between original and translated texts (Baroni and Bernardini, 2006; van Halteren, 2008; Ilisei et al., 2010; Koppel and Ordan, 2011; Volansky et al., 2012). This last work, in particular, uses text classification as a computational methodology, but its goal is to better understand the features of translationese.

The goal of this project is similar, but it uses the text classification methodology differently. You will be given a corpus consisting of *three* different types of texts (all in English): originals (O); human translations (T); and machine translation output (MT). 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 for implementing the classifier. You will be able to base your work on a similar effort whose goal was to distinguish between translated texts and original ones (Volansky et al., 2012).

Once the feature extractor is implemented, you will train classifiers to distinguish between O and MT, and use them to classify T. The expectation is that in certain features, T would be closer to O (both types of texts are produced by humans), whereas in other features, T would be closer to MT (both are translations).

## 3.5 A generic transliteration system

**Introduction to NLP highly recommended.**

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 approximates 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 methodology of Kirschenbaum and Wintner (2009, 2010). 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.

In order to evaluate the quality of your solution, you will have to prepare an evaluation corpus. This should consist of some 1000 hand-transliterated term-pairs (from various sources). You will evaluate the accuracy of your system on these held-out data.

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 Simplification of Hebrew sentences

**Introduction to NLP recommended but not mandatory.** Real-world sentences can be long and complex. Such complexity is achieved by two main linguistic mechanisms: coordination and subordination. The former allows the conjunction of two simpler sentences, as in: הפלסטינים מבינים שהם בבעיה ואנחנו מנסים להוריד אותם מהעץ יחד עם האמריקאים The latter combines two simpler sentences in an asymmetric way, where one sentence is said to be subordinated to the other: יהיה כמעט בלתי אפשרי למצוא נוסחה שתהיה מקובלת גם על נתניהו וגם על אבו מאזן.

A coordinated structure can in principle be repharsed as two sentences. For example, הפלסטינים מבינים שהם בבעיה ואנחנו מנסים להוריד אותם מהעץ יחד עם האמריקאים can be rephrased as הפלסטינים מבינים שהם בבעיה. אנחנו מנסים להוריד אותם מהעץ יחד עם האמריקאים A subordinated clause can also be simplified by splitting the sentence in two, but this may not be straight-forward. For example, the sentence יהיה כמעט בלתי אפשרי למצוא נוסחה שתהיה מקובלת גם על נתניהו וגם על אבו מאזן can be rephrased as יהיה כמעט בלתי אפשרי למצוא נוסחה. הנוסחה תהיה מקובלת גם על נתניהו וגם על אבו מאזן.

A third type of complexity, frequently observed in journalistic texts, involved quoting. For example, the sentence "אשטון העלתה כמה הצעות ורצינו לשמוע מה הפלסטינים חושבים על זה", אמרו גורמים בלשכת ראש הממשלה can be easily split into "אשטון העלתה כמה הצעות ורצינו לשמוע מה הפלסטינים חושבים על זה". כך אמרו גורמים בלשכת ראש הממשלה

The benefits of sentence simplification are many: such techniques can generate texts that may be easier to understand, for example for language learners. The main motivation of this project, however, is to investigate whether sentence simplification can be useful for improving the quality of an automatic Hebrew to English machine translation system.

You will have to identify linguistic constructions that naturally lend themselves to simplification; stipulate the rules that facilitate splitting one sentence into two or more shorter sentences; and implement a system that used the rules in order to simply arbitrary Hebrew texts.

To evaluate the quality of the system, you will experiment with an existing Hebrew to English MT system, with and without simplification, and compare the results.

## 3.7 Conversion of transcribed Hebrew to the standard script

**Introduction to NLP recommended but not mandatory.**

CHILDES (MacWhinney, 2000) is an on-line repository of hundreds of corpora recording spoken interactions between children and adults. The Hebrew section of CHILDES contains two large corpora. Both were manually transcribed, and the current transcription reflects both the pronunciation of the words and the specific consonant distinctions of the standard Hebrew orthography. Figure 1 depicts an example; observe that all vowels are reflected, as well as the main stress (as a horizontal bar over the stressed vowel); observe also that the transcription distinguishes between א and ע, and between ס and ש.

```
*MOT:   bo? nexapēṣ ?et ha- cvaʕīm .
%mor:   v|ba?&root:bw?&ptn:qal&form:imp&pers:2&gen:masc&num:sg=come
        ne#v|xipēṣ&root:xpṣ&ptn:piel&tense:fut&pers:1&gen:unsp&num:pl=search/look_for
        acc|?et det|ha-=the ?|cvaʕīm .
```

Figure 1: Hebrew CHILDES transcription example

We are currently developing a morphological analyzer for the Hebrew CHILDES section, whose output can be seen in Figure 1. One way to evaluate the accuracy of the analyzer is to

compare its analyses to the ones produced by the MILA analyzer of *written* Hebrew (Itai and Wintner, 2008). To this end, the transcription has to be converted to the standard Hebrew script.

You will develop software that converts the Hebrew CHILDES transcription to (undotted) Hebrew. You will also develop tools that run the MILA analyzer on the output of your program, and compares the results of the MILA analysis to the morphological annotation available in CHILDES. You will have to develop a set of conversion rules for the two types of analysis. You will also have to overcome difficulties caused by the fact that the CHILDES transcription reflects the vowels, whereas the MILA analyzer assume standard undotted Hebrew. The results of this project will be instrumental for us in improving the CHILDES morphological analyzer.

### 3.8 A web-based user interface for KWIC in Hebrew

**No prior knowledge is required.** Understanding of SQL databases and XML is recommended.

Key Word In Context (KWIC) is an algorithm which, given a text and a keyword, presents all the occurrences of the word in the text, allowing a few context words on both sides of the keyword to be displayed. Such a tool is very useful for linguistic research.

You will develop a KWIC system with a web-based graphical user interface which will allow users to present queries referring not just to words, but also to their morphological features. This tool will be similar to an existing GUI for Arabic (Dror et al., 2004), but will be specific to Hebrew corpora. The underlying corpora will be XML documents of morphologically analyzed Hebrew texts. The GUI will enable users to specify a corpus to work with, and then search the corpus for combinations of words and/or their properties. To this end, the corpora will have to be stored in an efficient database; you will be able to use an existing infrastructure for storing corpora, such as The Corpus Workbench. The GUI should be accessible on the Web, and hence will have to be developed in a Web-supporting environment, e.g., JSP or PHP.

A detailed requirements specification will be available in a separate document.

## 4 Available resources

You may freely use any available resources that you find useful for your project (respecting copyright and licensing agreements, of course). Specifically, you may find the following handy:

- Wikipedia as a source of multilingual texts, in particular in order to extract transliterated term-pairs
- Weka, a toolbox of various general-purpose machine learning tools, in particular in order to implement classifiers
- Open NLP, a set of tools for natural language processing, in particular in order to pre-process English texts
- NLTK, a natural language processing toolkit in Python
- The MILA resources for processing Hebrew.

# References

Marco Baroni. Distributions in text. In Anke Lüdeling and Merja Kytö, editors, *Corpus Linguistics: An International Handbook*, volume 2, chapter 37, pages 803–822. Mouton de Gruyter, Berlin, 2008. URL `http://sslmit.unibo.it/~baroni/publications/hsk_39_dist_rev2.pdf`.

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. URL `http://llc.oxfordjournals.org/cgi/content/short/21/3/259?rss=1`.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990. ISSN 0891-2017.

Yehudit Dror, Dudu Shaharabani, Rafi Talmon, and Shuly Wintner. Morphological analysis of the Qur'an. *Literary and linguistic computing*, 19(4):431–452, 2004.

Iustina Ilisei, Diana Inkpen, Gloria Corpas Pastor, and Ruslan Mitkov. Identification of translationese: A machine learning approach. In Alexander F. Gelbukh, editor, *Proceedings of CICLing-2010: 11th International Conference on Computational Linguistics and Intelligent Text Processing*, volume 6008 of *Lecture Notes in Computer Science*, pages 503–511. Springer, 2010. ISBN 978-3-642-12115-9. URL `http://dx.doi.org/10.1007/978-3-642-12116-6`.

Alon Itai and Shuly Wintner. Language resources for Hebrew. *Language Resources and Evaluation*, 42(1):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.

Amit Kirschenbaum and Shuly Wintner. A general method for creating a bilingual transliteration dictionary. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 273–276, Valletta, Malta, May 2010. European Language Resources Association (ELRA). ISBN 2-9517408-6-7.

Moshe Koppel and Noam Ordan. Translationese and its dialects. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1318–1326, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P11-1132`.

Brian MacWhinney. *The CHILDES Project: Tools for Analyzing Talk*. Lawrence Erlbaum Associates, Mahwah, NJ, third edition, 2000.

*Computational Linguistics Group*
*Department of Computer Science*
*University of Haifa*

*בלשנות חישובית*
*החוג למדעי המחשב*
*אוניברסיטת חיפה*

---

Hans van Halteren. Source language markers in EUROPARL translations. In Donia Scott and Hans Uszkoreit, editors, *COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK*, pages 937–944, 2008. ISBN 978-1-905593-44-6. URL `http://www.aclweb.org/anthology/C08-1118`.

Vered Volansky, Noam Ordan, and Shuly Wintner. On the features of translationese, 2012. To Appear.