Language and Computers
LTS: Grammatical Error Detection

Based on slides from Ross Israel
Indiana University
Fall 2013
What Is Grammatical Error Detection?

We will be talking about errors made by learners in a second language acquisition context.

Language Learners often make non-native-like mistakes when constructing sentences:

- We arrived to the station.
- There is the garden in my house.
- I eat rice, nikujaga and salada yesterday.

Grammatical error detection entails trying to find these mistakes automatically.
Where Is This Useful?

- Automatic grading
  - Language teachers
  - Standardized testing
- Analysis and annotation of learner data for research
- Language learning software (ICALL)
Where Error Detection Fits In
(a bit of a review)

- **CALL: Computer Assisted Language Learning**
  - Using computers and media in language learning and teaching
  - e.g., Rosetta Stone, eLanguage
  - Exercises are typically very simple in design, and offer little feedback

- **ICALL: Intelligent Computer Assisted Language Learning**
  - Utilize computational linguistics tools, such as POS tagging & parsing along with statistical language modeling strategies (e.g., n-grams)
    - These tools often need to be altered to expect and diagnose errors, or at least handle learner data better
    - We can also build software for specific kinds of errors. (**today’s discussion**)
  - Focus on precision; we don’t want to tell a learner that they’ve made a mistake when they haven’t!
Learner Errors

Learners typically make different kinds of mistakes than native speakers.

- **Content Word Choice**
  - We need to deliver the merchandise on a daily *base/basis.*

- **Preposition Error**
  - Our society is developing *in/at high* speed.

- **Determiner Error**
  - There is *the/a garden in my house.*
Motivation

Some common areas of research in English error detection are articles, prepositions, and collocations. We’ll look a little more in depth about prepositions.

- Because prepositions make up a large portion of errors commonly made by learners, there has been a good deal of research on how to find and diagnose preposition errors.
- Also, prepositions are a closed set, so it’s a problem that’s easier to define than a more open error type like use of the wrong content word.
- Prepositions can be treated as a confusion set where we know that one is being substituted for another.
Choosing the correct preposition can be a tough task even for native speakers.

There has been concern about syncing phone contacts with Facebook. "As long as you are aware of who is in the group it can be a great privacy tool. If it gets out of hand it could give you a sense of false security." The roll out of new products comes with reports that a syncing feature lets Facebook access contact data and share it with the site. "It's very possible that your private phone numbers - and those lots - are on the site," said Charles Arthur of the Guardian newspaper.
Choosing the correct preposition can be a tough task even for native speakers.

There has been concern over syncing phone contacts with Facebook. "As long as you are aware of who is in the group it can be a great privacy tool. If it gets out of hand it could give you a sense of false security." The roll out of new products comes amid reports that a syncing feature on the iPhone lets Facebook access contact data and share it on the site. "It’s very possible that your private phone numbers - and those of lots of your and their friends - are on the site," said Charles Arthur of the Guardian newspaper.
Commonly Used Techniques

- Language Model - Gamon et al. (2008)
  - Build n-grams of POS and/or parsing labels from native text and check if learner data n-grams align with the model we build

- Web-based methods - Gamon and Leacock (2010)
  - Take a few words of context on either side of a preposition to generate a web query
  - Replace the preposition with neighbors from a confusion set and search those queries
  - The search with the greatest number of hits is selected as the right answer

  - Write linguistic rules designed to find errors in learner data

- Statistical - Tetreault and Chodorow (2008)
  - Statistical methods means building a classifier
  - So, what is a classifier?
Understanding Classifiers

Machine learning is not as scary as it sounds!

▶ There are a number of algorithms for classification that we could talk about
  ▶ Maximum Entropy, Support Vector Machines, Memory Based Learning
  ▶ Each method requires different representations of information
  ▶ These slides are indicative of Memory Based Learning
▶ TiMBL http://ilk.uvt.nl/timbl/ - Daelemans et al. (2007)
  ▶ Easy to install
  ▶ Easy to use
  ▶ Works well with language data
Running TiMBL

- We will need two sets of data:
  - Training Set - needs to be big
  - Testing Set - usually smaller

- The data sets are full of events (*instances*) that contain *features* that describe the circumstances of the event and a *class* that is the answer we are trying to guess

- With a little bit of a language like python, you can extract features pretty easily to train and run TiMBL
  - Open a file (e.g. a POS tagged file)
  - Extract bits of text (features) that you deem useful
  - Print those bits of text on a single line for each instance
  - The real trick is selecting appropriate features
Selecting Features

Let’s consider a real-world example:

- The Task: We want to classify the weather as either good or bad.
- We would want features like
  - temperature
  - sunny?
  - cloudy?
  - windy?
  - humidity level
  - rain/snow/none
Running the Classifier

- Then, we would build **vectors** for every measurement we take and *label* them to build training data:
  - 75, yes, no, no, 70%, none, good
  - 35, no, no, yes, 50%, none, bad
  - 105, yes, no, no, 98%, rain, bad
  - 68, yes, yes, no, 75%, none, good

- Now, when we give the classifier an unknown feature vector, we hope that it makes a wise decision
  - 85, yes, no, no, 65%, none - classifier’s guess = **good** yay!
  - 15, no, no, yes, 70%, snow - classifier’s guess = **bad** yay!
  - 75, yes, no, yes, 70%, none - classifier’s guess = **bad** oops!
Tetreault and Chodorow used a maximum entropy classifier to try to find preposition confusions and extraneous uses.

They extracted 25 features including:

- words/POS tags in a 2 word window(+/−) around preposition
- the head verb and noun of the preceding VP and NP
- the head noun of the following NP

John went to the store this morning.

- word+POS bigrams: went_vbd, the_det
- head of previous VP = went
- head of previous NP = John

Their system achieved 84% precision and 19% recall.

This might sound low, but keep in mind, we want to get the best possible precision, even if it means losing recall.
Korean language learning

- This is based on joint work with Ross Israel & Sun-Hee Lee - Dickinson et al. (2010)
- Ultimate goal: develop computational tools to assist learners of Korean
- Korean has a number of features uncommon to Western languages:
  - agglutinative morphology, rich system of case marking, relatively free word order
  - One prominent area of difficulty for learners is that of post-positional particles

We want to build a machine learner to detect errors in particles
Background: Korean particles

Similar to English prepositions, but wider range of functions:

- Case marker/Semantic role markers:

  (1) Sumi-ka John-eykey chayk-ul ilhke-yo
      Sumi-SBJ John-to book-OBJ read-polite
      ‘Sumi reads a book to John.’

- Modifiers (cf. prepositions): indicate specific lexical, syntactic, & semantic information between verb & noun

  (2) Sumi-ka John-uy cip-eyse ku-lul
      Sumi-SBJ John-GEN house-LOC he-OBJ
twu sikan-ul kitaly-ess-ta.
two hours-OBJ wait-PAST-END
      ‘Sumi waited for John for (the whole) two hours in his house.’

Korean particles: expected errors

Learners of Korean often misuse particles

(3) a. Sumi-*nun* chayk-*i* philyohay-*yo*
   Sumi-TOP book-SBJ need-polite
   ‘Sumi needs a book.’

   b. *Sumi-nun* chayk-*ul* philyohay-*yo*
      Sumi-TOP book-OBJ need-polite
      ‘Sumi needs a book.’

particle errors by learners of Korean can be categorized into 6 types:

- omission, replacement, addition, malformation, paraphrasing, and spacing
Parallel errors made by ESL learners for prepositions

- We can base our system on the work by Tetreault and Chodorow, but we need to consider the differences between English and Korean

Some major differences in Korean:

- Particles are post-positional - they show up after a word instead of before it like in English
- Base word order is SOV
  - Need to look at following verb & following noun
- Morphological composition of words is different
  - Agglutinative: stem+suffixes
Our Features

▶ For this experiment, we used TiMBL
▶ The feature vector is built on a five word window that includes the target word and two words (+/-) for context.
▶ Each word is broken down into four features:
  ▶ Stem, affixes, stem_POS, affixes_POS
    ▶ Use trigram+rule based morphological tagger for Korean
  ▶ Include features for preceding and following nouns & verbs (roots only)
▶ If the target word affixes contains a particle, it is removed and used as the class; otherwise the class is NONE
  ▶ In this study, we only predicted a particle’s presence (Y/N)
An Example Instance

(4) a. Mikwuk-eysa sal-myense Yenge-man-ul
    America-in live-while English-only-OBJ
cip-eysa ss-ess-eyo.
home-at use-Past-Decl
‘While living in America, (I/she/he) used only English at home.’

b. Mikwuk NPR NONE NONE
   sal VV myense ECS
   Yenge NPR NONE NONE
cip NNC NONE NONE
   ss VV ess+eyo EPF+EFN
   sal Mikwuk ss cip
   YES
Results

- Our system was able to achieve around 84% precision and around 81% recall for this task.
- The recall is higher here because the task is simpler than guessing *which* particle is best, we just try to guess *if* there should be a particle.
- We have continued this work by adding the decision of which particle to use (too much for today ...)
References


