Heteronymous Ambiguity Resolution for Text-to-Speech Synthesis

Paloma Casteleiro Costa, Desmond Caulley, Sonali Govindaluri, Jinsol Lee, Alex Parisi
Georgia Institute of Technology, Atlanta, GA

Problem

The focus of this project is to determine a statistical decision method for heteronymous ambiguity resolution to accomplish proper pronunciation when attempting text-to-speech synthesis. A homograph is a word that has the same written form as another, without sharing the same meaning. When these two words also have a different pronunciation, they are called heteronyms.

Most text-to-speech synthesizers are unable to resolve obvious heteronyms, and therefore may cause confusion to the listener. This problem can be exacerbated if the listener is visually impaired, and cannot resolve the ambiguity themselves by sight. Mispronouncing a word can completely change the meaning of a sentence or phrase, which we would like to avoid.

Introduction

Current methods use N-Gram taggers, Bayesian classifiers, and decision trees, each coming with their advantages and disadvantages:

N-Gram Tagger Assigns a "part of speech" to each word in a sentence, such as noun, verb, or adjective.

Most heteronyms can be resolved using their parts of speech, but not all can - "bass" can mean the fish or the instrument, and both words are nouns.

Bayesian Classifier Takes the words surrounding the heteronym, builds a "bag of words" model, and then classifies based on word association.

Handles various heteronyms well, but many depend on word placement as well as word association – "close" can be resolved depending on if "to" comes directly before or after.

Decision Trees Performs well, and can resolve most conditional dependencies, but may struggle with high-dimensional data.

Our project focuses on the implementation of each classification scheme, how they compare to one another, and the designing of an algorithm that combines the advantages of each into a coherent, yet robust classifier.

Implementation

Dataset We used the "Reuters-21578, Distribution 1.0" text corpus as our dataset. It is a diverse collection of articles on the Reuters newswire in 1987. This data is written in Standard Generalized Markup (SGM) language, and therefore required some cleanup to reduce it to raw text form.

Vectorizer Most classification algorithms will not accept raw text documents with variable lengths; therefore, we must use a vectorizer to assign an integer ID to each unique token word. Because our corpus is so large, we only vectorize the 20 words that surround the target heteronym.

N-Gram Tagger Since the N-gram tagger will assign a part of speech to a target heteronym in a sentence using context, we do not use the corpus to train. The library we used to perform the tagging was already trained, and only required the sentence containing the target heteronym as an input. Since our heteronyms were separable by their parts of speech, if we assign a part of speech to each label, the N-gram tagger should be able to properly resolve the ambiguity.

Bayesian Classifier After cleaning up the corpus, we found each instance of the target heteronym, and then vectorized the 20 surrounding words into three lists – the words before the heteronym, the words after, and a combination of both. Each list of vectorized words is treated as a "bag of words" and processed by a Naïve Bayes classifier.

The list of words that come before the heteronym are prioritized as these give us the most accurate estimation. The results from each classifier are combined together to give us an accurate prediction of the final pronunciation.

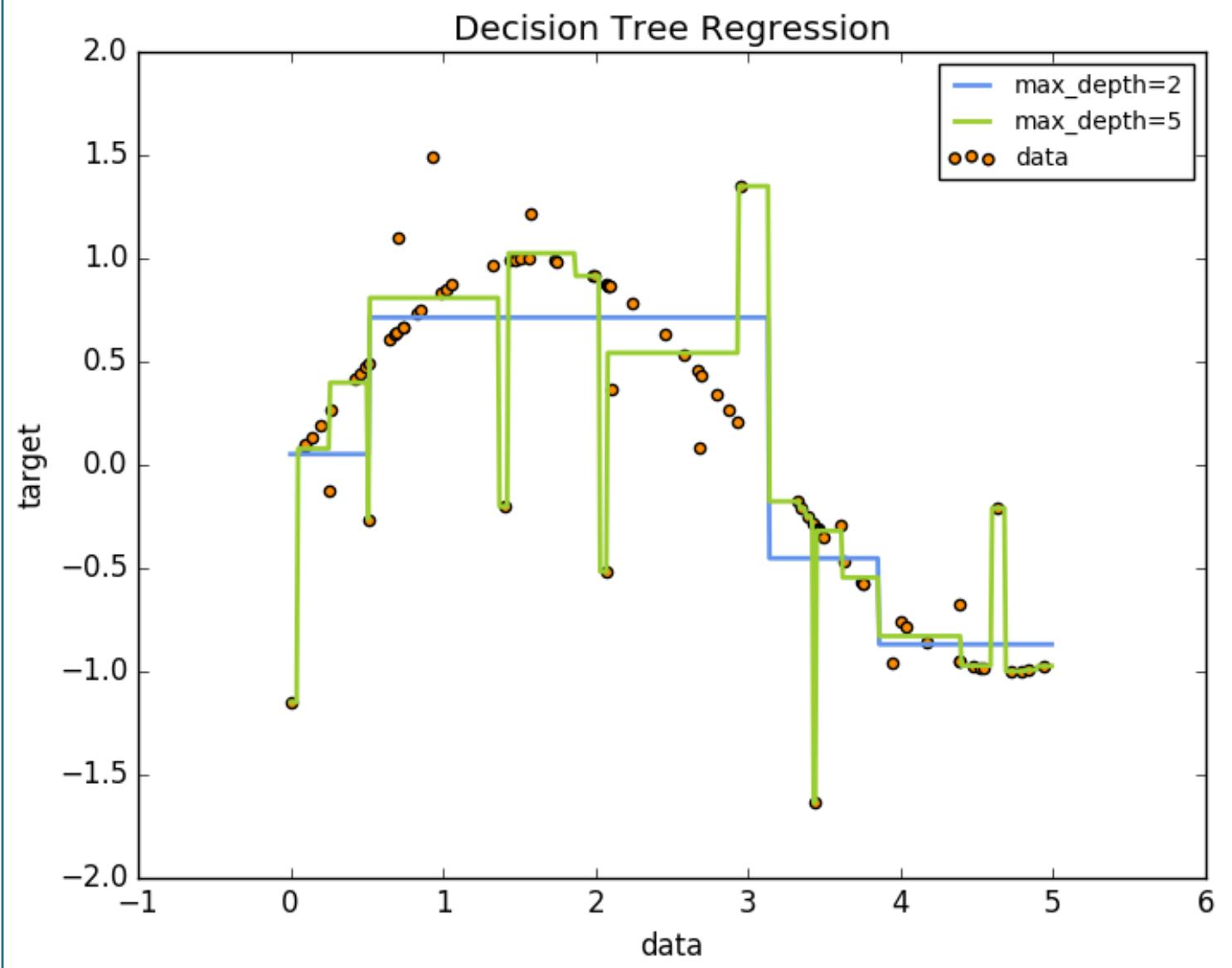
Decision Trees Similarly to the Bayesian classifier, we are using the same three lists of the vectorized words. Each of these lists are treated as a vector of features, and sent into a decision tree classifier along with the labels.

The tree finds the "best" attribute and sets it as the root node. The node is then split into the next two best features for each class, which continues down until all features have been exhausted. A new sentence can now be sorted by vectorizing it and passing it through the constructed tree.

Libraries "regular expression operations (re)", "scikit-learn", "openpyxl", "Natural Language Toolkit (nltk)"

Results

Classifier	Heteronym	Accuracy
N-Gram Tagger	"Object"	52.5%
	"Minute"	87.5%
	"Close"	90.0%
	"Use"	97.5%
Naïve Bayesian	"Object	80.0%
	"Minute"	97.5%
	"Close"	32.5%
	"Use"	82.5%
Decision Trees	"Object"	80.0%
	"Minute"	97.5%
	"Close"	12.5%
	"Use"	87.5%


Decision Tree Regression

Heteronym	Label	Part of Speech
Object	"OBject" – 0	Noun
	"obJECT" – 1	Verb
Minute	"MINute" – 0	Noun
	"minUTE" – 1	Adjective
Close	"cloZe" – 0	Verb/Noun
	"cloSe" – 1	Adjective/Adverb
Use	"uZe" – 0	Verb
	"uSe" – 1	Noun

Drawbacks

- N-Gram taggers only work well when the target heteronyms are separable by their parts of speech, and can only accept grammatically correct English.
  - Some heteronyms may also have fringe cases where their parts of speech may be hard to define - e.g. the verb "to close" can be a noun in certain rare cases, while retaining the same pronunciation.
- The Bayesian classifier also performs well, but does not account for word placement.
  - Many heteronyms can be resolved solely by the specificities of its surrounding words, which we are ignoring in this case – e.g. "close" can almost always be resolved depending on if "to" comes before or after.
- Decision trees are unstable, and tiny variations in the feature set can drastically alter the structure of the tree.
  - They also may not generalize the data well – to avoid this, we may opt to implement random forests instead.

Contact

Paloma Casteleiro Costa
Georgia Tech
Email: casteleiro@gatech.edu

Desmond Caulley
Georgia Tech
Email: Caulley@gatech.edu

Sonali Govindaluri
Georgia Tech
Email: sonaligovindaluri@gatech.edu

Jinsol Lee
Georgia Tech
Email: jinsol.lee@gatech.edu

Alex Parisi
Georgia Tech
Email: alex.parisi@gatech.edu

References

1. Hearst, Marti A., and Xerox Palo Alto Research Center. "Noun Homograph Disambiguation Using Local Context in Large Text Corpora." (n.d.): 1-15. Web.
2. Rivest, Ronald L. &quot;Learning Decision Lists.&quot; (2001): 1-18. Web.
3. Weber, Heinz J. "The Automatically Built Up Homograph Dictionary - A Component of a Dynamic Lexical System." (n.d.): 1-11. Web.
4. Yarowsky, David. Homographic Disambiguation in Text-to-Speech Synthesis. N.p.: n.p., n.d. Print.