An Algorithm that Learns What's in a Name

Daniel M. Bikel
Richard Schwartz
Ralph Weischedel

Overview

- Written in 1999, IdentiFinder is a hidden Markov model (HMM) designed to recognize names, dates, times, and numerical quantities.
- The IdentiFinder model was evaluated on data from the 6th and 7th Message Understanding Conferences (MUC) as well as the first Multilingual Entity Task (MET).
- Both Spanish and English data was analyzed.

The NER Task

- The named entity recognition (NER) task is to identify all named locations, named persons, named organizations, dates, times, monetary amounts, and percentages in text.
- This sounds simple, but there are issues that can complicate basic rule sets.

The NER Task

The delegation, which included the commander of the U.N. troops in Bosnia, Lt. Gen. Sir Michael Rose, went to the Serb stronghold of Pale, near Sarajevo, for talks with Bosnian Serb leader Radovan Karadzic.

Este ha sido el primer comentario publico del presidente Clinton respecto a la crisis de Oriente Medio desde que el secretario de Estado, Warren Christopher, decidiera regresar precipitadamente a Washington para impedir la ruptura del proceso de paz tras la violencia desatada en el sur de Libano.

1. Locations
2. Persons
3. Organizations

Evaluation Metric

A computer program is used to evaluate performance based on:

- Precision
- Recall
- F Measure

Evaluation Metric

Precision:

P = number of correct responses / number of responses

- A "response" is "an answer delivered by a name finder."

Evaluation Metric

Recall:

R = number of correct responses / number correct in key

- The key is "an annotated file containing the correct answers."

Evaluation Metric

F measure:

F = RP / ½(R+P)

Evaluation Metric

What counts as correct?

- Bikel et al. use MUC and MET standards.
  - Correct boundaries.
  - Correct labels.
- Answers can be partially right if only some conditions are met.

Evaluation Metric

MUC and MET Label Types

- Entity (ENAMEX): person, organization, location
- Time expression (TIMEX): date, time
- Numeric expression (NUMEX): money, percent

Why NER?

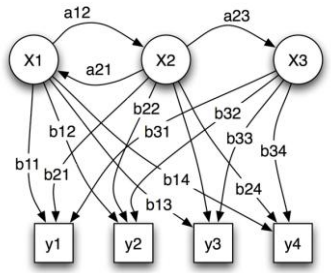
- "It seems to be both useful and solvable."
- The NER problem is fairly easy in mixed case English text, but becomes an interesting problem when dealing with other languages where case information is not available, or non-text modalities (like speech).
- Representative of a general challenge for learning.

Why NER?

Why a learning algorithm?

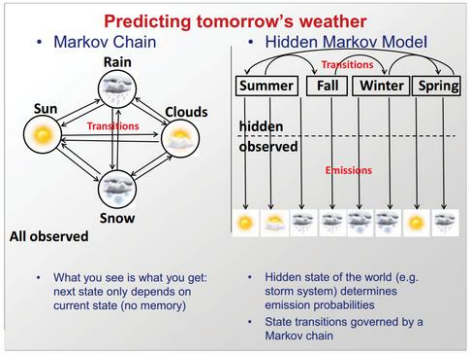
- Learning algorithms are more generalizable than hand crafted rules.
  - They work better for non-text modalities.
- A learning algorithm reduces the need for human input.
- Each new source of text for a rule based system requires large scale tweaking of the rule set.

What's an HMM?



http://homepages.inf.ed.ac.uk/group/sli_archive/slip0809_c/s0562005/theory.html

What's an HMM?



http://www.quora.com/Hidden-Markov-Models/What-is-a-simple-explanation-of-the-Hidden-Markov-Model-algorithm

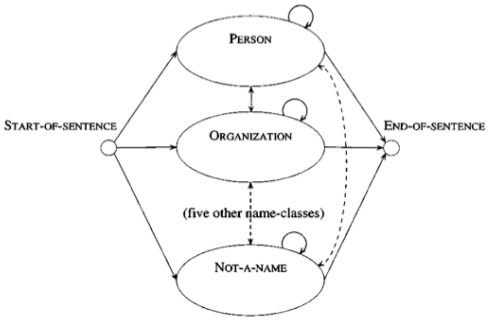
The Hidden Markov Model

Overview

NER can be viewed as a classification problem in which:

- Only one label can be assigned to a word.
- Every word is either part of some name or not part of any name.

The Hidden Markov Model



The Hidden Markov Model

Overview

- Bikel et al.'s model assigns to every word either one of the desired classes or the label NOT-A-NAME.
- The HMM has a model for each of the desired classes, as well as the rest of the training text.
- An arbitrary number of classes can be added to the system at run time.

The Hidden Markov Model

Overview

- In addition to all this, there are two special states:
  - START-OF-SENTENCE
  - END-OF-SENTENCE

The Hidden Markov Model

- Within each of the established regions, a model for computing the likelihood of words occurring within that region (name-class) is used.
  - This model is a bigram language model.
  - The likelihood of a given word is based solely on the previous word.
- Every word is represented by a state. There is a probability associated with every transition to the next word.
- Given this, the likelihood of a sequence of words W1 through Wn is found using:

$$\Pi^{n}_{i=1}\, p(w_i \mid w_{i-1})$$

- A special +begin+ word is used to compute the likelihood of W_1.

The Hidden Markov Model

- In addition to word sequences, the most likely sequence of classes must also be found.
  - That is to say: Max Pr (NC | W)
  - NC: name-class
  - W: sequence of words
- This paper assumes a generative model where the HMM generates the sequence of words and labels using Bayes Rule:
  - Pr(NC | W) = Pr(W, NC) / Pr (W)

The Hidden Markov Model

The Generation of Words and Word Classes

1. Select a name-class *NC,* conditioning on the previous name-class and the previous word.
2. Generate the first word inside that name-class, conditioning on the current and previous name classes.
3. Generate all subsequent words inside the current name-class, where each subsequent word is conditioned on its immediate predecessor (as per a standard bigram model).

The Hidden Markov Model

- These three steps are then repeated until the entire observed word sequence is generated.
- The entire space of all possible name-class assignments is searched, maximizing the numerator of the previous Baye's rule equation.

The Hidden Markov Model

- Constructing the model in this way means that each type of "name" should be viewed as its own language, with separate bigram probabilities for generating its words.
- This affects the the intuitions regarding the model in the following ways:
  - There is generally predictive internal evidence regarding the class of a desired entity.
  - Logical external evidence often suggests the boundaries and class of one of the desired expressions.

Word Features

- This part of the language model is language-dependent.
- Fortunately, though, the implementation is only roughly twenty lines of code long.
- Word features are conceptualized as ordered pairs (or two-element vectors) composed of a word and its word feature.
  - <w, f>
- The word feature is a deterministic computation performed on each word as it is added to or looked up in the dictionary.

Word Features

Word feature	Example text	Intuition
twoDigitNum	90	Two-digit year
fourDigitNum	1990	Four-digit year
containsDigitAndAlpha	A8956-67	Product code
containsDigitAndDash	09-96	Date
containsDigitAndSlash	11/9/89	Date
containsDigitAndComma	23,000.00	Monetary amount
containsDigitAndPeriod	1.00	Monetary amount, percentage
otherNum	456789	Other number
allCaps	BBN	Organization
capPeriod	M.	Person name initial
firstWord	*first word of sentence*	No useful capitalization information
initCap	Sally	Capitalized word
lowerCase	can	Uncapitalized word
other	,	Punctuation marks, all other words

Formal Model

Top Level Model

1. A model to generate a name-class.
2. A model to generate the first word in a name-class.
3. A model to generate all subsequent words in a name class.

Formal Model

- In order to generate the first word, a transition must be made from one name-class to another, as well as calculating the likelihood of that word.
- This works because words preceding a name class can be hugely helpful in determining the class (words like Mr.). Words following a name class can help determine the following class, as well.

Formal Model

- Generating the first word of the name-class:
  - $PR(NC \mid NC_{-1}, w_{-1}) \cdot PR(\langle w, f \rangle_{first} \mid NC, NC_{-1})$
- Generating all but the first word:
  - $Pr(\langle w, f \rangle \mid \langle w, f \rangle_{-1}, NC)$
- Generating the final word (+end+ is a special word allowing any word to be the final word in its class):
  - $Pr(\langle +end+, other \rangle \mid \langle w, f \rangle_{final} \mid NC)$

Formal Model

- It would be useless to have the first word of a new name-class be generated on the +end+ word of a previous class.
- This is overcome by conditioning the new class on the last real word in the previous class.
  - It's still allowed to be +end+ if the previous class is START-OF-SENTENCE. Otherwise it's the last observed word.

Dealing with Unknown Words

- Ideally, the training data would contain every instance that is observed in the data. This, unfortunately, is rarely the case.
- All unknown words are mapped to the token _UNK_.
- Some training data is withheld in order to train an unknown word model.
  - This wins the authors an idea of how often _UNK_ appears in their training data.
  - 50% of data is held, and an unknown word model is trained on that set (the vocabulary was built on the first 50%).
  - The counts in that model are stored in a data file.
  - Then the other 50% is held out, and the bigram counts of this file are concatenated with the counts in the first unknown training file.
- This allows the likelihood of unknown data to be calculated using all of the data.

Back-off Strategy

- Whether a bigram model contains an unknown word or not, it's possible that a given bigram may still be unknown (never seen in the training data).
- The model gives a weight to the likelihood that a back-off is necessary.
- A back up model is built to decrease specificity of a probability when necessary.
  - $Pr(NC \mid NC_{-1}, w_{-1})$
  - $Pr(NC \mid NC_{-1})$
  - $Pr(NC)$
  - 1 / number of name-classes

Results

- Slightly worse performance than rule based NER on mixed case data.
  - The performance is close enough that the learning approach is likely still more useful, ultimately.
- Performance on the Spanish data was worse than the English data.
- Out performed all previous approaches when mixed case data was not available.
- Required no labor to handle upper case or speech format.
  - Only required a few machine cycles to convert mixed case training data to other forms, and retrain.

Conclusions

- None of the formalisms in this paper were new. But applying them to the NER task, as well as the model itself was novel.
- This paper produced an efficient learning algorithm that is largely language independent, and that performs near human levels.
- To the authors knowledge, this model produced a higher f-measure than any other learned NE system at the time.

Our Project

- Currently, the SHARP grant uses an NER system to pre-annotate clinical records.
- It's not very good.
- James is going to make a better one.
- I will analyze the data output of his system and identify patterns in the errors. He'll use this to improve the system.