STANFORD UNIVERSITY

CS 224N: Natural Language Processing

Final Project Report

Sander Parawira

6/5/2010

In this final project we built a Part of Speech Tagger using Hidden Markov Model. We determined the most likely sequence of tags of a sentence by applying Viterbi Algorithm to the sequence of words of that sentence.

Hidden Markov Model and Viterbi Algorithm

Hidden Markov Model

Hidden Markov Model is a stochastic model in which the system being modeled is assumed to be a Markov Process with unobservable states but observable outputs. Hidden Markov Model consists of three components:

1. $P_S(S_i)$: Probability of the system starting in state S_i
2. $P_T(S_j|S_i)$: Probability of the system transitioning from state S_i to state S_j
3. $P_E(X_j|S_i)$: Probability of the system emitting output X_j in state S_i

In the specific case of our Part of Speech Tagger, the tags are assumed to be the states and the words are assumed to be the outputs. Hence, our Part of Speech Tagger consists of:

1. $P_S(T_i)$: Probability of the sequence starting in tag T_i
2. $P_T(T_j|T_i)$: Probability of the sequence transitioning from tag T_i to tag T_j
3. $P_E(W_j|T_i)$: Probability of the sequence emitting word W_j on tag T_i

Given a sequence of words, our Part of Speech Tagger is interested in finding the most likely sequence of tags that generates that sequence of words. In order to accomplish this, our Part of Speech Tagger makes two simplifying assumptions:

1. The probability of a word depends only on its tag. It is independent of other words and other tags.
2. The probability of a tag depends only on its previous tag. It is independent of next tags and tags before the previous tag.

Thus, given a sequence of n words $W_1W_2 \dots W_n$, the most likely sequence of tags $T_1T_2 \dots T_n$ is

$$T_1T_2 \dots T_n =$$

$$\underset{T_1T_2 \dots T_n}{argmax} P(T_1T_2 \dots T_n|W_1W_2 \dots W_n) =$$

$$\underset{T_1T_2 \dots T_n}{argmax} P(W_1W_2 \dots W_n|T_1T_2 \dots T_n) \frac{P(T_1T_2 \dots T_n)}{P(W_1W_2 \dots W_n)} =$$

$$\underset{T_1T_2 \dots T_n}{argmax} P(W_1W_2 \dots W_n|T_1T_2 \dots T_n) P(T_1T_2 \dots T_n) =$$

$$\underset{T_1T_2 \dots T_n}{argmax} \prod_{i=1}^{n} P(W_i|T_i) \prod_{i=2}^{n} P(T_i|T_{i-1})\, P(T_1)$$

Suppose that our corpus is a k-tag Treebank with tags $t_1, t_2, \dots, t_k$ and m words $w_1, w_2, \dots, w_m$ in the dictionary. If we compute the most likely sequence of n tags by enumerating all possible sequence of tags, then the running time of our algorithm is $O(k^n)$. This is clearly very inefficient and obviously unfeasible. Therefore, we calculate the most likely sequence of tags by using the Viterbi Algorithm.

Viterbi Algorithm

Suppose that our corpus is a k-tag Treebank with tags $t_1, t_2, \dots, t_k$ and m words $w_1, w_2, \dots, w_m$ in the dictionary. Let $P[r, s]$ for $1 \le r \le n, 1 \le s \le k$ be the greatest probability among all probabilities of sequence of tags $T_1 T_2 \dots T_r$ with $T_r = t_s$. Let $L[r, s]$ for $1 \le r \le n, 1 \le s \le k$ be the sequence of tags $T_1 T_2 \dots T_r$ with $T_r = t_s$ corresponding to that probability. Then, the Viterbi Algorithm for our Part of Speech Tagger can be described as follows:

1. Set $P[1, s] = P(W_1 = w_i | T_1 = t_s) P(T_1 = t_s)$ for $1 \le s \le k$

2. Set $L[1, s] = \{t_s\}$ for $1 \le s \le k$

3. Set $P[r, s] = \underset{1 \le j \le k}{max} P[r-1, j] P(W_r = w_i | T_r = t_s) P\left(T_r = t_s | T_{r-1} = t_j\right)$ for $2 \le r \le n$ and $1 \le s \le k$

4. Set $L[r, s] = \{L[r-1, \underset{1 \le j \le k}{argmax} P[r-1, j] P(W_r = w_i | T_r = t_s) P\left(T_r = t_s | T_{r-1} = t_j\right)], t_l\}$ for $2 \le r \le n$ and $1 \le s \le k$

5. Then the most likely sequence of tags is given by $L[n, \underset{1 \le j \le k}{argmax} P[n, j]]$

It is easy to see that the running time of the Viterbi Algorithm for our Part of Speech Tagger is $O(nk^2)$ which is much more efficient and consequently, feasible.

Implementations and Experiments

We implemented four Hidden Markov Models. The first model is Laplace Smoothed Hidden Markov Model which uses Laplace smoothed probability densities. The second model is Absolute Discounting Hidden Markov Model which uses absolute discounting probability densities. The third model is Interpolation Hidden Markov Model which interpolates higher order and lower order probability densities. The last model is Extended Hidden Markov Model which looks at two previous tags instead of just the previous tag. In all of our models, we assume that that our corpus is a k-tag Treebank with tags $t_1, t_2, \dots, t_k$ and m words $w_1, w_2, \dots, w_m$ in the dictionary.

We experimented on two sets of data. The first set of data is the 6-tag Treebank Mini corpus which is taken from http://reason.cs.uiuc.edu. It has 900 tagged sentences for training and 100 tagged sentences for testing. The second set of data is the 87-tag Treebank Brown corpus which is taken from http://www.stanford.edu/dept/linguistics/corpora. It has 56617 tagged sentences. We split it into 56517 tagged sentences for training and 100 tagged sentences for testing.

Laplace Smoothed Hidden Markov Model

Overview

We define the Laplace smoothed probability of the sequence starting in tag t_i for $1 \leq i \leq k$ as

$$P_S(t_i) = \frac{C(t_i) + 1}{\sum_{j=1}^{k} C(t_j) + k}$$

Observe that $\frac{C(t_i)+1}{\sum_{j=1}^{k} C(t_j)+k} > 0$ and $\sum_{i=1}^{k} \frac{C(t_i)+1}{\sum_{j=1}^{k} C(t_j)+k} = \frac{\sum_{i=1}^{k} C(t_i)+k}{\sum_{j=1}^{k} C(t_j)+k} = 1$. So, $P_S(t_i)$ is a valid probability density.

Now, we define the Laplace smoothed probability of the sequence transitioning from tag t_i to tag t_j for $1 \leq j \leq k$ as

$$P_T(t_j|t_i) = \frac{C(t_i t_j) + 1}{C(t_i) + k}$$

Observe that $\frac{C(t_i t_j)+1}{C(t_i)+k} > 0$ and $\sum_{j=1}^{k} \frac{C(t_i t_j)+1}{C(t_i)+k} = \frac{C(t_i)+k}{C(t_i)+k} = 1$. So, $P_T(t_j|t_i)$ is a valid probability density.

Finally, we define the Laplace smoothed probability of the sequence emitting word w_j on tag t_i for $1 \leq j \leq m$ as

$$P_E(w_j|t_i) = \frac{C(t_i w_j) + 1}{C(t_i) + m}$$

Observe that $\frac{C(t_i w_j)+1}{C(t_i)+m} > 0$ and $\sum_{j=1}^{m} \frac{C(t_i w_j)+1}{C(t_i)+m} = \frac{C(t_i)+m}{C(t_i)+m} = 1$. So, $P_E(w_j|t_i)$ is a valid probability density.

Simulation and Error Analysis

We trained our Part of Speech Tagger on 900 tagged sentences from the Mini data set training sentences. Then, we tested it on 100 tagged sentences from the Mini data set testing sentences. This resulted in an accuracy of **90.03%**. The confusion matrix for the errors is as follows:

		True Tag					
		NOUN	VERB	FUNCT	PUNCT	CONJ	OTHER
Most Likely Tag	NOUN	-	20	26	0	0	6
	VERB	27	-	0	0	0	10
	FUNCT	3	0	-	0	0	2
	PUNCT	0	0	0	-	0	0
	CONJ	0	0	0	0	-	0
	OTHER	27	6	22	0	0	-

From the confusion matrix, we see that the five most common mistakes are classifying NOUN as VERB, classifying NOUN as OTHER, classifying FUNCT as NOUN, classifying FUNCT as OTHER, and classifying VERB as NOUN. We also see that PUNCT and CONJ are always correctly classified.

An example of a perfectly tagged sentence: `we_noun_noun are_verb_verb not_other_other concerned_verb_verb here_other_other with_funct_funct a_funct_funct law_noun_noun of_funct_funct nature_noun_noun ._punct_punct`.

Note that the format is the word followed by the true tag and the most likely tag.

An example of a poorly tagged sentence: `however_other_other ,_punct_punct this_funct_funct factory_noun_noun increased_verb_verb its_noun_noun` **`profits_noun_verb`** `by_funct_funct` **`83_noun_funct`** `%_noun_noun in_funct_funct 2002_noun_noun ,_punct_punct compared_verb_verb with_funct_funct 2001_noun_noun ,_punct_punct and_conj_conj` **`received_verb_noun`** `a_funct_funct fat_other_other` **`subsidy_noun_verb`** `from_funct_funct the_funct_funct greek_other_other government_noun_noun ._punct_punct`.

Similarly, we trained our Part of Speech Tagger on 56517 tagged sentences from the Brown data set training sentences. Then, we tested it on 100 tagged sentences from the Brown data set testing sentences. This resulted in an accuracy of **88.16%**.

The five most common errors are classifying NP as NN, classifying NN as NP, classifying VB as VBD, classifying JJ as NP, and classifying NN as NNS. We noticed that there is almost no perfectly tagged sentence. The Viterbi Algorithm usually makes one or two mistakes per sentence.

For example: `the_at_at operator_nn_nn asked_vbd_vbd` **`pityingly_rb_ppo`** `._._.`.

And another example: `and_cc_cc` **`how_wrb_ql right_jj_rb`** `she_pps_pps was_bedz_bedz ._._.`

Laplace smoothed probabilities do not work well for N-Gram language models. So it is possible that Laplace smoothed probabilities also do not work well for our Part of Speech Tagger. For that reason, we decided to implement Absolute Discounting Hidden Markov Model.

Absolute Discounting Hidden Markov Model

Overview

We define the absolute discounting probability of the sequence starting in tag t_i for $1 \leq i \leq k$ as

$$P_S(t_i) = \left[\frac{C(t_i) - D_S}{\sum_{j=1}^{k} C(t_j)}\right]^+ + \frac{D_S \sum_{j=1}^{k} 1\{C(t_j) > 0\}}{k \sum_{j=1}^{k} C(t_j)}$$

Observe that $\left[\frac{C(t_i)-D_S}{\sum_{j=1}^{k} C(t_j)}\right]^+ + \frac{D_S \sum_{j=1}^{k} 1\{C(t_j)>0\}}{k \sum_{j=1}^{k} C(t_j)} > 0$ and $\sum_{i=1}^{k} \left[\frac{C(t_i)-D_S}{\sum_{j=1}^{k} C(t_j)}\right]^+ + \frac{D_S \sum_{j=1}^{k} 1\{C(t_j)>0\}}{k \sum_{j=1}^{k} C(t_j)} = \frac{\sum_{i=1}^{k} C(t_i) - D_S \sum_{i=1}^{k} 1\{C(t_i)>0\}}{\sum_{j=1}^{k} C(t_j)} + \frac{D_S \sum_{j=1}^{k} 1\{C(t_j)>0\}}{\sum_{j=1}^{k} C(t_j)} = 1$. So, $P_S(t_i)$ is a valid probability density.

Now, we define the absolute discounting probability of the sequence transitioning from tag t_i to tag t_j for $1 \leq j \leq k$ as

$$P_T(t_j|t_i) = \left[\frac{C(t_i t_j) - D_T}{C(t_i)}\right]^+ + \frac{D_T \sum_{h=1}^{k} 1\{C(t_i t_h) > 0\}}{kC(t_i)}$$

Observe that $\left[\frac{C(t_i t_j)-D_T}{C(t_i)}\right]^+ + \frac{D_T \sum_{h=1}^{k} 1\{C(t_i t_h)>0\}}{kC(t_i)} > 0$ and $\sum_{j=1}^{k} \left[\frac{C(t_i t_j)-D_T}{C(t_i)}\right]^+ + \frac{D_T \sum_{h=1}^{k} 1\{C(t_i t_h)>0\}}{kC(t_i)} = \frac{C(t_i) - D_T \sum_{j=1}^{k} 1\{C(t_i t_j)>0\}}{C(t_i)} + \frac{D_T \sum_{h=1}^{k} 1\{C(t_i t_h)>0\}}{C(t_i)} = 1$. So, $P_T(t_j|t_i)$ is a valid probability density.

Finally, we define the absolute discounting probability of the sequence emitting word w_j on tag t_i for $1 \leq j \leq m$ as

$$P_E(w_j|t_i) = \left[\frac{C(t_i w_j) - D_E}{C(t_i)}\right]^+ + \frac{D_E \sum_{h=1}^{m} 1\{C(t_i w_h) > 0\}}{mC(t_i)}$$

Observe that $\left[\frac{C(t_i w_j)-D_E}{C(t_i)}\right]^+ + \frac{D_E \sum_{h=1}^{m} 1\{C(t_i w_h)>0\}}{mC(t_i)} > 0$ and $\sum_{j=1}^{m} \left[\frac{C(t_i w_j)-D_E}{C(t_i)}\right]^+ + \frac{D_E \sum_{h=1}^{m} 1\{C(t_i w_h)>0\}}{mC(t_i)} = \frac{C(t_i) - D_E \sum_{j=1}^{m} 1\{C(t_i w_j)>0\}}{C(t_i)} + \frac{D_E \sum_{h=1}^{m} 1\{C(t_i w_h)>0\}}{C(t_i)} = 1$. So, $P_E(w_j|t_i)$ is a valid probability density.

Simulation and Error Analysis

We trained our Part of Speech Tagger on 900 tagged sentences from the Mini data set training sentences. Then, we tested it on 100 tagged sentences from the Mini data set testing sentences. From our experiments, $D_S = 0.50, D_T = 0.50$, and $D_E = 0.50$ yielded the highest accuracy which was **92.73%**. The confusion matrix for the errors is as follows:

		True Tag					
		NOUN	VERB	FUNCT	PUNCT	CONJ	OTHER
Most Likely Tag	NOUN	-	24	0	0	0	10
	VERB	37	-	0	0	0	7
	FUNCT	0	0	-	0	0	3
	PUNCT	0	0	0	-	0	0
	CONJ	0	0	0	0	-	0
	OTHER	41	6	2	0	0	-

From the confusion matrix, we see that the three most common mistakes are classifying NOUN as OTHER, classifying NOUN as VERB, and classifying VERB as NOUN. Furthermore, we see that classification for FUNCT is improved significantly compared to Laplace Smoothed Hidden Markov Model. We also see that PUNCT and CONJ are always correctly classified as before.

An example of a perfectly tagged sentence: we_noun_noun would_other_other do_verb_verb better_other_other to_funct_funct put_verb_verb this_funct_funct into_funct_funct the_funct_funct explanations_noun_noun and_conj_conj notes_noun_noun ._punct_punct.

An example of a poorly tagged sentence: the_funct_funct european_other_other institutions_noun_noun are_verb_verb not_other_other state_noun_noun organisations_noun_noun but_conj_conj **supernational_other_noun** authorities_noun_noun to_funct_funct **whom_funct_noun** a_funct_funct limited_other_other number_noun_noun of_funct_funct powers_noun_noun are_verb_verb **delegated_verb_noun** ._punct_punct.

Similarly, we trained our Part of Speech Tagger on 56517 tagged sentences from the Brown data set training sentences. Then, we tested it on 100 tagged sentences from the Brown data set testing sentences. From our experiments, $D_S = 0.50, D_T = 0.50$, and $D_E = 0.50$ yielded the highest accuracy which was **92.79%**.

The four most common errors are classifying NN as NP, classifying JJ as NN, classifying NP as NN, and classifying VB as VBD. We noticed that there is almost no perfectly tagged sentence. The Viterbi Algorithm usually makes one or two mistakes per sentence.

For example: `his_pp$_pp$ hubris_nn_nn ,_,_, deficiency_nn_nn of_in_in taste_nn_nn ,_,_, and_cc_cc sadism_nn_nn carried_vbd_vbd him_ppo_ppo` **`straightaway_rb_nn`** `to_in_in the_at_at top_nn_nn ._._..`

And another example: `not_*_*` **`long_jj_rb`** `ago_rb_rb ,_,_, i_ppss_ppss rode_vbd_vbd down_rp_rp with_in_in him_ppo_ppo in_in_in an_at_at elevator_nn_nn in_in_in radio_nn_nn city_nn_nn ;_._..`

Absolute discounting probabilities do not have means to interpolate with lower order models. It may be the case that interpolating with lower order models can improve our Part of Speech Tagger. For that reason, we decided to implement Interpolation Hidden Markov Model.

Interpolation Hidden Markov Model

Overview

We define the interpolation probability of the sequence starting in tag t_i for $1 \le i \le k$ as

$$P_S(t_i) = \left[\frac{C(t_i) - D_S}{\sum_{j=1}^{k} C(t_j)}\right]^+ + \frac{D_S \sum_{j=1}^{k} 1\{C(t_j) > 0\}}{k \sum_{j=1}^{k} C(t_j)}$$

$P_S(t_i)$ is a valid probability density as we showed earlier.

Now, we define the interpolation probability of the sequence transitioning from tag t_i to tag t_j for $1 \le j \le k$ as

$$P_T(t_j|t_i) = \lambda_{T2} P_{T2}(t_j|t_i) + \lambda_{T1} P_{T1}(t_j)$$

where

$$\lambda_{T2} + \lambda_{T1} = 1, \lambda_{T2} > 0, \lambda_{T1} > 0$$

,

$$P_{T2}(t_j|t_i) = \left[\frac{C(t_i t_j) - D_{T2}}{C(t_i)}\right]^+ + \frac{D_{T2} \sum_{h=1}^{k} 1\{C(t_i t_h) > 0\}}{k C(t_i)}$$

, and

$$P_{T1}(t_j) = \left[\frac{C(t_j) - D_{T1}}{\sum_{i=1}^{k} C(t_i)}\right]^+ + \frac{D_{T1} \sum_{i=1}^{k} 1\{C(t_i) > 0\}}{k \sum_{i=1}^{k} C(t_i)}$$

$P_{T2}(t_j|t_i)$ and $P_{T1}(t_j)$ are valid probability densities as we proved earlier. So, $P_T(t_j|t_i)$ is also a valid probability density.

We computed the optimal values for λ_{T2} and λ_{T1} using the Deleted Interpolation Algorithm. The Deleted Interpolation Algorithm can be described as follows:
1. Set $\lambda_{T2} = 0, \lambda_{T1} = 0$
2. For each tag t_i and tag t_j such that $C(t_i t_j) > 0$:
   Depending on the maximum of:
      Case $\frac{C(t_i t_j)-1}{C(t_i)-1}$: increment λ_{T2} by $C(t_i t_j)$
      Case $\frac{C(t_j)-1}{\sum_{h=1}^{k} C(t_h)-1}$: increment λ_{T1} by $C(t_i t_j)$

Finally, we define the interpolation probability of the sequence emitting word w_j on tag t_i for $1 \le j \le m$ as

$$P_E(w_j|t_i) = \lambda_{E2} P_{E2}(w_j|t_i) + \lambda_{E1} P_{E1}(w_j)$$

where

$$\lambda_{E2} + \lambda_{E1} = 1, \lambda_{E2} > 0, \lambda_{E1} > 0$$

,

$$P_{E2}(w_j|t_i) = \left[\frac{C(t_i w_j) - D_{E2}}{C(t_i)}\right]^+ + \frac{D_{E2} \sum_{h=1}^{m} 1\{C(t_i w_h) > 0\}}{mC(t_i)}$$

, and

$$P_{E1}(w_j) = \left[\frac{C(w_j) - D_{E1}}{\sum_{i=1}^{m} C(w_i)}\right]^+ + \frac{D_{E1} \sum_{i=1}^{m} 1\{C(w_i) > 0\}}{m \sum_{i=1}^{m} C(w_i)}$$

Observe that $\left[\frac{C(w_j)-D_{E1}}{\sum_{i=1}^{m} C(w_i)}\right]^+ + \frac{D_{E1} \sum_{i=1}^{m} 1\{C(w_i)>0\}}{m \sum_{i=1}^{m} C(w_i)} > 0$ and $\sum_{j=1}^{m} \left[\frac{C(w_j)-D_{E1}}{\sum_{i=1}^{m} C(w_i)}\right]^+ + \frac{D_{E1} \sum_{i=1}^{m} 1\{C(w_i)>0\}}{m \sum_{i=1}^{m} C(w_i)} = \frac{\sum_{j=1}^{m} C(w_j) - D_{E1} \sum_{j=1}^{m} 1\{C(w_j)>0\}}{\sum_{i=1}^{m} C(w_i)} + \frac{D_{E1} \sum_{i=1}^{m} 1\{C(w_i)>0\}}{\sum_{i=1}^{m} C(w_i)} = 1$. So, $P_{E1}(w_j|t_i)$ is a valid probability density. On the other hand, $P_{E2}(w_j|t_i)$ is a valid probability density as showed earlier. Hence, $P_E(w_j|t_i)$ is also a valid probability density.

Similarly, we computed the optimal values for λ_{E2} and λ_{E1} using the Deleted Interpolation Algorithm.

Simulation and Error Analysis

We trained our Part of Speech Tagger on 900 tagged sentences from the Mini data set training sentences. Then, we tested it on 100 tagged sentences from the Mini data set testing sentences. From our experiments, $D_S = 1.00, D_{T2} = 0.25, D_{T1} = 0.75, D_{E2} = 0.25$, and $D_{E1} = 0.50$ yielded the highest accuracy which was **93.01%**. The confusion matrix for the errors is as follows:

		True Tag					
		NOUN	VERB	FUNCT	PUNCT	CONJ	OTHER
Most Likely Tag	NOUN	-	10	1	0	0	3
	VERB	52	-	0	0	0	5
	FUNCT	2	0	-	0	0	2
	PUNCT	0	0	0	-	0	0
	CONJ	0	0	0	0	-	0
	OTHER	47	2	3	0	0	-

From the confusion matrix, we see that the two most common mistakes are classifying NOUN as VERB and classifying NOUN as OTHER. Furthermore, we see that classification for VERB is improved compared to Absolute Discounting Hidden Markov Model. We also see that PUNCT and CONJ are always correctly classified as before.

An example of a perfectly tagged sentence: liability_noun_noun will_other_other ensure_verb_verb that_funct_funct producers_noun_noun are_verb_verb careful_other_other about_funct_funct how_funct_funct they_noun_noun produce_verb_verb ._punct_punct.

An example of a poorly tagged sentence: if_funct_funct these_funct_funct proposals_noun_noun are_verb_verb accepted_verb_verb as_funct_funct they_noun_noun **stand_verb_noun** ,_punct_punct europe_noun_noun will_other_other be_verb_verb **committing_verb_noun** a_funct_funct serious_other_other **strategic_other_noun** error_noun_noun by_funct_funct reducing_verb_verb these_funct_funct payments_noun_noun for_funct_funct the_funct_funct major_other_other crops_noun_noun ._punct_punct

Similarly, we trained our Part of Speech Tagger on 56517 tagged sentences from the Brown data set training sentences. Then, we tested it on 100 tagged sentences from the Brown data set testing sentences. From our experiments, $D_S = 1.00, D_{T2} = 0.25, D_{T1} = 0.75, D_{E2} = 0.25$, and $D_{E1} = 0.50$ yielded the highest accuracy which was **93.00%**.

The four most common errors are classifying NN as NP, classifying JJ as NN, classifying NP as NN, and classifying VBN as VBD. We noticed that there are several perfectly tagged sentences.

For example: `his_pp$_pp$ energy_nn_nn was_bedz_bedz prodigious_jj_jj ;_._.`

And another example: `he's_pps+bez_pps+bez really_rb_rb asking_vbg_vbg for_in_in it_ppo_ppo ._._..`

Interpolation probabilities only look at the current tag and the previous tag. It is likely that looking at the two previous tags can improve our Part of Speech Tagger. For that reason, we decided to implement Extended Hidden Markov Model.

Extended Hidden Markov Model

Overview

We define the interpolation probability of the sequence starting in tag t_i for $1 \leq i \leq k$ as

$$P_S(t_i) = \left[\frac{C(t_i) - D_S}{\sum_{j=1}^{k} C(t_j)}\right]^+ + \frac{D_S \sum_{j=1}^{k} 1\{C(t_j) > 0\}}{k \sum_{j=1}^{k} C(t_j)}$$

$P_S(t_i)$ is a valid probability density as we proved earlier.

Now, we define the interpolation probability of the sequence transitioning from tag $t_h t_i$ to tag t_j for $1 \leq j \leq k$ as

$$P_T(t_j|t_h t_i) = \lambda_{T3} P_{T3}(t_j|\, t_h t_i) + \lambda_{T2} P_{T2}(t_j|t_i) + \lambda_{T1} P_{T1}(t_j)$$

where

$$\lambda_{T3} + \lambda_{T2} + \lambda_{T1} = 1, \lambda_{T3} > 0, \lambda_{T2} > 0, \lambda_{T1} > 0$$

,

$$P_{T3}(t_j|t_h t_i) = \left[\frac{C(t_h t_i t_j) - D_{T3}}{C(t_h t_i)}\right]^+ + \frac{D_{T3} \sum_{g=1}^{k} 1\{C(t_h t_i t_g) > 0\}}{k C(t_h t_i)}$$

,

$$P_{T2}(t_j|t_i) = \left[\frac{C(t_i t_j) - D_{T2}}{C(t_i)}\right]^+ + \frac{D_{T2} \sum_{h=1}^{k} 1\{C(t_i t_h) > 0\}}{k C(t_i)}$$

, and

$$P_{T1}(t_j) = \left[\frac{C(t_j) - D_{T1}}{\sum_{i=1}^{k} C(t_i)}\right]^+ + \frac{D_{T1}\sum_{i=1}^{k} 1\{C(t_i) > 0\}}{k\sum_{i=1}^{k} C(t_i)}$$

Observe that $\left[\frac{C(t_h t_i t_j) - D_{T3}}{C(t_h t_i)}\right]^+ + \frac{D_{T3}\sum_{g=1}^{k} 1\{C(t_h t_i t_g) > 0\}}{kC(t_h t_i)} > 0$ and $\sum_{j=1}^{k}\left[\frac{C(t_h t_i t_j) - D_{T3}}{C(t_h t_i)}\right]^+ + \frac{D_{T3}\sum_{g=1}^{k} 1\{C(t_h t_i t_g) > 0\}}{kC(t_h t_i)} = \frac{C(t_h t_i) - D_{T3}\sum_{j=1}^{k} 1\{C(t_h t_i t_j) > 0\}}{C(t_h t_i)} + \frac{D_{T3}\sum_{g=1}^{k} 1\{C(t_h t_i t_g) > 0\}}{C(t_h t_i)} = 1$. So, $P_{T3}(t_j|t_h t_i)$ is a valid probability density. On the other hand, $P_{T2}(t_j|t_i)$ and $P_{T1}(t_j)$ are valid probability densities as we showed earlier. Thus, $P_T(t_j|t_i)$ is also a valid probability density.

We computed the optimal values for $\lambda_{T3}, \lambda_{T2}$, and λ_{T1} using the Deleted Interpolation Algorithm.

Finally, we define the interpolation probability of the sequence emitting word w_j on tag t_i for $1 \leq j \leq m$ as

$$P_E(w_j|t_i) = \lambda_{E2} P_{E2}(w_j|t_i) + \lambda_{E1} P_{E1}(w_j)$$

where

$$\lambda_{E2} + \lambda_{E1} = 1, \lambda_{E2} > 0, \lambda_{E1} > 0$$

,

$$P_{E2}(w_j|t_i) = \left[\frac{C(t_i w_j) - D_{E2}}{C(t_i)}\right]^+ + \frac{D_{E2}\sum_{h=1}^{m} 1\{C(t_i w_h) > 0\}}{mC(t_i)}$$

, and

$$P_{E1}(w_j) = \left[\frac{C(w_j) - D_{E1}}{\sum_{i=1}^{m} C(w_i)}\right]^+ + \frac{D_{E1}\sum_{i=1}^{m} 1\{C(w_i) > 0\}}{m\sum_{i=1}^{m} C(w_i)}$$

$P_E(w_j|t_i)$ is a valid probability density as we proved earlier.

Similarly, we computed the optimal values for λ_{E2} and λ_{E1} using the Deleted Interpolation Algorithm.

Simulation and Error Analysis

We trained our Part of Speech Tagger on 900 tagged sentences from the Mini data set training sentences. Then, we tested it on 100 tagged sentences from the Mini data set testing sentences. From our experiments, $D_S = 1.0, D_{T3} = 0.25, D_{T2} = 0.50, D_{T1} = 0.75, D_{E2} = 0.25,$

and $D_{E1} = 0.75$ yielded the highest accuracy which was **93.01%**. The confusion matrix for the errors is as follows:

		True Tag					
		NOUN	VERB	FUNCT	PUNCT	CONJ	OTHER
Most Likely Tag	NOUN	-	10	1	0	0	7
	VERB	49	-	0	0	0	5
	FUNCT	2	0	-	0	0	3
	PUNCT	0	0	0	-	0	0
	CONJ	0	0	0	0	-	0
	OTHER	46	1	3	0	0	-

From the confusion matrix, we see that the two most common mistakes are classifying NOUN as VERB and classifying NOUN as OTHER. We also see that PUNCT and CONJ are always correctly classified as before. We do not see any improvements compared to the Interpolation Hidden Markov Model.

An example of a perfectly tagged sentence: if_funct_funct the_funct_funct percentage_noun_noun is_verb_verb 90_noun_noun %_noun_noun ,_punct_punct so_other_other be_verb_verb it_noun_noun ._punct_punct.

An example of a poorly tagged sentence: as_funct_funct you_noun_noun **hear_verb_noun** ,_punct_punct mr_noun_noun solana_noun_noun and_conj_conj mr_noun_noun patten_noun_noun ,_punct_punct we_noun_noun **all_funct_other** feel_verb_verb **incredibly_other_noun** **powerless_other_noun** ,_punct_punct **disgusted_other_noun** and_conj_conj **frustrated_other_noun** ._punct_punct

Similarly, we trained our Part of Speech Tagger on 56517 tagged sentences from the Brown data set training sentences. Then, we tested it on 100 tagged sentences from the Brown data set testing sentences. From our experiments, $D_S = 1.0, D_{T3} = 0.25, D_{T2} = 0.50, D_{T1} = 0.75, D_{E2} = 0.25$, and $D_{E1} = 0.75$ yielded the highest accuracy which was **92.87%**.

The five most common errors are classifying NN as NP, classifying JJ as NN, classifying NP as NN, classifying NN as NNS, and classifying VBN as VBD. We noticed that there are several perfectly tagged sentences. For example: i_ppss_ppss wouldn't_md*_md* be_be_be in_in_in his_pp$_pp$ shoes_nns_nns for_in_in all_abn_abn the_at_at rice_nn_nn in_in_in china_np_np ._._..

And another example: in_in_in this_dt_dt work_nn_nn ,_,_, his_pp$_pp$ use_nn_nn of_in_in non-color_nn_nn is_bez_bez startling_jj_jj and_cc_cc skillful_jj_jj ._._..

Overall, we noticed that the performance of Extended Hidden Markov Model was equal or slightly worse compared to Interpolation Hidden Markov Model.

Conclusion and Future Work

Out of the four Hidden Markov Models we built, Laplace Smoothed Hidden Markov Model has the lowest accuracy (90.03% for the Mini corpus data set and 88.16% for the Brown corpus data set) since Laplace smoothed probabilities do not work well for our Part of Speech Tagger. Conversely, Interpolation Hidden Markov Model has the highest accuracy (93.01% for the Mini corpus data set and 93.00% for the Brown corpus data set) since interpolating between higher order and lower order probabilities work very well for our Part of Speech Tagger.

Since the performances of our Part of Speech Tagger are similar for Mini corpus data set and Brown corpus data set, we infer that the number of tags does not have detrimental effect on the accuracy as long as we have sufficient data. For the Mini corpus data set, most of the classification mistakes are made on the NOUN tag. Our Part of Speech Tagger erroneously classified NOUN as VERB or classified NOUN as OTHER. Conversely, for the Brown corpus data set, our Part of Speech Tagger has difficulty distinguishing NN vs NP vs JJ and VB vs VBN vs VBD.

In the future, one may try using the Expectation Maximization Algorithm to calculate the optimal weights for the interpolation between higher order and lower order probabilities. One may also try to use the Expectation Maximization Algorithm to evaluate the optimal discounting values for the probabilities. Or one may even try to use a different probability smoothing scheme altogether. Finally, one may try to extend the Hidden Markov Model even further by looking at the previous three tags. Nevertheless, we are not hopeful for the last approach since our Extended Hidden Markov Model performed equally or slightly worse than our Interpolation Hidden Markov Model.

Bibliography

Brants, T. (2000). A Statistical Part-of-Speech Tagger. *Sixth Applied Natural Language Processing Conference.*

Jurafsky, D., & Martin, J. H. (2008). *Speech and Language Processing.* Prentice Hall.

Weischedel, R., Schwartz, M., & Ramshaw, R. (1993). Coping with Ambiguity and Unknown Words through Probabilistic Models. *Computational Linguistics* .