Course: Natural Language Processing

Assignment 2: Advanced Text Processing

Oreen Yousuf

November 29, 2020

1 POS-Tagging

The best version of my tagger was "t3-tagger", which had an Accuracy rate of 93.09 percent. The following are errors made by the tagger:

Sentence 1: "From the AP comes this story: [...]" - "AP" was mistagged as NOUN instead of PROPN. This can be chalked up to anything having the possibility of being a proper noun. This case can be considered as genuinely ambiguous. This instance would not be considered as mistagged by the gold standard ewt-dev-wt.txt file.

Sentence 2: "Bush nominated Jennifer M. Anderson for a 15-year term as associate judge of the [...]" - " - " was 'mistagged' as SYM instead of PUNCT. I do think that the PUNCT tagging is incorrect in this case. The hyphen (-) used here is just a connector for the full phrase "15-year". This mistagging in the gold standard of the ewt-dev-wt.txt file could've been avoided if the phrase "15-year" was tokenized as a single token rather than broken up into three individual tokens.

Sentece 3: "Bush nominated Jennifer M. Anderson for a 15-year term as associate judge of the [...]" - "associate" was mistagged as NOUN instead of ADJ. This instance is another case of ambiguity, as "associate" has three parts-of-speech; verb, noun, and adjective. This instance was utilizing the adjective form of associate, to mean "connected with" and is often seen as a way to rank something.

Sentence 4: "Bush also nominated A. Noel Anketel Kramer for a 15-year term as associate judge of the District of Columbia Court of Appeals, replacing John [...]". - "Appeals" was mistagged as NOUN instead of PROPN. The word "appeal" can be used as a noun or verb. Here the tagger believes it's being used as a noun meaning "a sincere or earnest request or plea", however it's used as an official proper noun for a specific judiciary body.

Sentence 5: "The sheikh in wheel-chair has been attacked with a F-16-launched bomb." - "F", in "F-16", was 'mistagged' as a PROPN instead of NOUN. However, just as in Sentence 2, I believe the tag of PROPN from the t3-tagger is in fact the correct tag, as the token in question is that of a specific item/noun. There isn't much ambiguity to discuss for this example.

Sentence 6: "He could be killed years ago and the israelians have all the reasons [...]". - "israelians" was mistagged as NOUN instead of PROPN. The incorrect tagging of this as a noun could be attributed to the fact that "israelians" is not capitalized in the text, as it should be (I believe). Furthermore, because it's referring to a specific group/nationality it should qualify as a proper noun.

Sentence 7: "...is the spiritual leader of Hamas, but they didn't." - "did" was mistagged as AUX instead of VERB. This one can be ambiguous for it's linguistic nature. For it to be an auxilary verb, a verb must follow "did" or "didn't" in it's base form. And because a base-form verb (and nothing, really) follows it, it's acting as a verb and not an auxiliary verb in this instance.

My native languages are Harari and English, and there isn't a tagset for Harari as it's a very small language. But the biggest, immediately related language of Amharic has one that we can use to make a relative comparison. Below are some of the tags in the tagset, with a couple in more detailed description:

Basic Class	Definition of the Tag	Code of the Tag
Noun	Any noun including verbal noun with a proclitic preposition and an enclitic conjunction	NPC
Pronoun	Pronoun attached with conjunction	PRONC
Verb	Any Verb including relative verbs and auxiliaries attached with preposition	VP
Adjective	Adjective attached with preposition	ADJP

Tokenizing highly optimizes NLP models in the end. This is achieved by parts-of-speech being allocated to all elements and tags being utilized for punctuation. This "simple" implementation can lead to punctuation, symbols, special characters, etc. being peeled or not peeled off of words. Denoting end-of-sentence punctuation characters (such as exclamation marks, question marks and periods) from word-internal punctuation (like e.g., i.e., Ph.D, and etc.) is greatly valued in part-of-speech tagging. Next, because tokenization is deployed to partition out a text into words and symbols, it becomes easier later on in the process to fundamentally understand background information in the original text in addition to neatly breaking up somewhat debated on aspects such as contractions, like ' n't ', and ' 's ', from their root/stem words. From this somewhat structured attempt at creating concrete tokens, you are more easily able to explicate raw text and data with the compiled list of each and every token that has been tagged.

Shortly, as we intake a word sequence we can then deduce the corresponding hidden tags from said word sequence. This enables us to obtain the key signal sequence and forecast tags, which is a bedrock of utilizing Hidden Markov Models (HMM) for Part of Speech (POS) tagging. HMM models are likelihood sequence models that pair off a list of observed incidents with specific tag sequences. As we learned in the Hidden Markov Models lab, We can use the phrase "emitted signals" to correlate to the observation events and we can use the phrase "hidden state of the model" for all possible word sequences for said emitted signals. When we use Hidden Markov Models for Part of Speech tagging, these previously described emitted signals will be the sequence of words and Part of Speech tags will be the hidden state. Once the probabilistic calculations of all possible word tagged sequences are done, we can then choose the likely one and start to find the transition probability (the probability of state i transferring to state j) and emission probability (the probabilities of observations o being generated from state q).

2 Lemmatization

In Lab 7, you have tuned a lemmatizer. Based on the best version of your lemmatizer, you should do a manual analysis of remaining errors. Describe at least 5 error types and discuss how they could be tackled in a more sophisticated lemmatizer.

The following are error analyses from my lemmatizer python file that reached 96 percent:

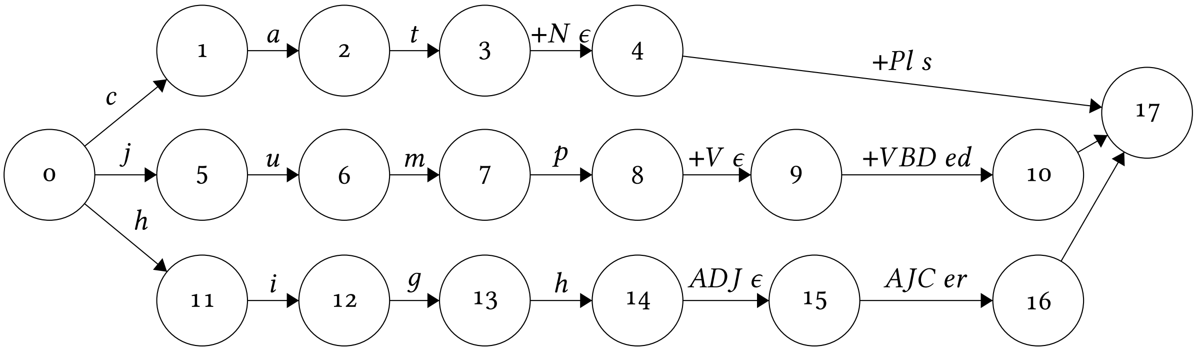
The ADJ lemma of "other" should correctly be "other" but it was seen as "oth". We can rectify this with an if-statement subcondition as ' if word=="other": return "other" '.

The VERB lemma of "went" should correctly be "go", but it was seen as "went". We can recify this with an if-statement subcondition for this special case as ’ if word=="went": return "go" ’.

The VERB lemma of "made" should correctly be "make", but it was seen as "made". We can rectify this with an if-statement subcondition, also for this special case" as ’ if word=="made": return "make" ’.

The VERB lemma of "r", broken off from the word "your", should be correctly seen as "be", but was seen as "r". When seeing this I thought again of a subcondition to fix this with an if-statement returning "be", but in my code I already accounted for "yours" or "your" to return "you".

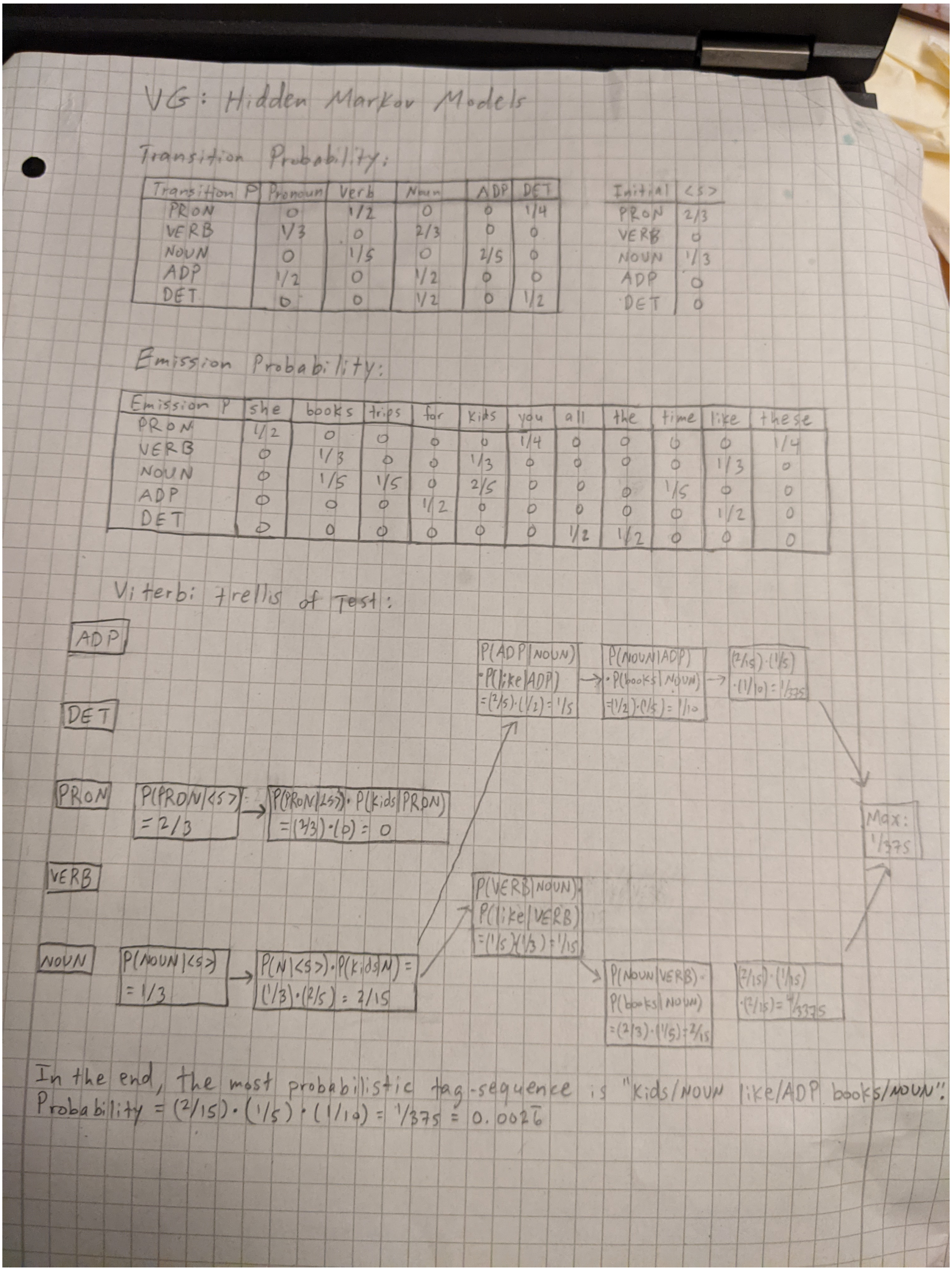
The VERB lemma "submitted" should correctly be "submit". This was a common occurrence with verb inflected in the past form. We can rectify this by seeing if it ends with 2 repeated consonants after removing the "ed", and if so we can remove an additional consonant.



The inputs of the finite state transmitter (FST) are cats/NOUN, jumped/VERB, higher/ADJ and the outputs are cat +Pl -s, jump +VBD -ed, and high AJC -er.

In Semitic languages like Harari, Amharic or Arabic, every verb can have 2-3 dozen distinct inflectional forms that also have many special cases. Additionally, because of the numerous plural forms, to be able to parse them all we’d have to save all the morphological variants for each verb. There can still be overlooked data within the corpus even if we create an incredibly large training corpus for morphologically rich languages such as these because it is not feasible to account for all forms of words.

3 VG: Hidden Markov Models



VG: Hidden Markov Models

Transition Probability:

Transition P	Pronoun	Verb	Noun	ADP	DET
PRON	0	1/2	0	0	1/4
VERB	1/3	0	2/3	0	0
NOUN	0	1/5	0	2/5	0
ADP	1/2	0	1/2	0	0
DET	0	0	1/2	0	1/2

Initial	<s>
PRON	2/3
VERB	0
NOUN	1/3
ADP	0
DET	0

Emission Probability:

Emission P	she	books	trips	for	kids	you	all	the	time	like	these
PRON	1/2	0	0	0	0	1/4	0	0	0	0	1/4
VERB	0	1/3	0	0	1/3	0	0	0	0	1/3	0
NOUN	0	1/5	1/5	0	2/5	0	0	0	1/5	0	0
ADP	0	0	0	1/2	0	0	0	0	0	1/2	0
DET	0	0	0	0	0	0	1/2	1/2	0	0	0

Viterbi trellis of Test:

ADP
DET
PRON
VERB
NOUN
P(PRON|<s>) = 2/3
P(PRON|<s>)·P(kids|PRON) = (2/3)·(0) = 0
P(NOUN|<s>) = 1/3
P(N|<s>)·P(kids|N) = (1/3)·(2/5) = 2/15
P(ADP|NOUN) ·P(like|ADP) = (2/5)·(1/2) = 1/5
P(NOUN|ADP) ·P(books|NOUN) = (1/2)·(1/5) = 1/10
(2/15)·(1/5) ·(1/10) = 1/375
P(VERB|NOUN)· P(like|VERB) = (1/5)·(1/3) = 1/15
P(NOUN|VERB)· P(books|NOUN) = (2/3)·(1/5) = 2/15
(2/15)·(1/15) ·(2/15) = 4/3375
Max: 1/375

In the end, the most probabilistic tag-sequence is "kids/NOUN like/ADP books/NOUN".
Probability = (2/15)·(1/5)·(1/10) = 1/375 = 0.002$\overline{6}$