CMPUT 497: Assignment 3

Shouyang Zhou
University of Alberta
Edmonton, Alberta, Canada
shouyang@ualberta.ca

1 Assignment Part One

Implementation
My assignment implements the models and components required as a collection of scripts which can be separated into three classes.

The scripts “test_file_adjuster.py” and “train_file_adjuster.py” preprocess the cleaned data files from a tabular format into underscore suffixed line-per-sentence text. These scripts take a file path as a command line argument to print out the adjusted text.

The scripts prefixed with “nltk” denote scripts that train a model using the first command line argument and tag a text preprocessed by the test adjuster script using the second command line argument. There are two variants, one that does not handle unknown tokenization and one that does. They are labelled as original and replacement. Under replacement the training text is processed such that words with word counts below a threshold (generally 1) are substituted with an unknown token. When tagging a sentence, words not seen in the training set (or below the threshold) are replaced with the unknown token for tagging.

Lastly, the script “tabluate_results.py”, takes a “truth file”, a tagged file from a model, and a training file to generate a .tsv file of results. These were used to construct confusion matrices via Excel pivot tables.

The Stanford POS tagger was trained and tagged using command line arguments and associated. props files. The input files were the files generated by the two adjuster scripts. The parameters used were based on the “English-left3words-distsim.tagger” as supplied by the Stanford site. The authors advocate for applications to use this model, as such I found it an intuitive basis for this assignment. The Stanford models do not use the replacement scheme as opposed the brill and HMM taggers. The Stanford tagger appears to have its own mechanism for handling unknowns via its rare word parameter.

The Brill taggers uses the rule-template set “fntbl37”, investigating the differing rulesets, others have found this one to be better performing. The brill tagger uses a back-off n-gram model (3,2,1, “NN”) for its initialization.

The two HMMs only differ in the training data, one uses the replacement method, one does not.

Results
For the following analysis, please refer to the data analysis Excel file suffixed “Part_1-2”. For brevity, I will report on the general trends between the test evaluations given the 24 cases recorded. Also, the .tsvs for Stanford taggers have been adjusted for alignment slightly in the Excel file. Some tokens are further processed by the Stanford tagger leading to misalignments, for example the token “Vol.” is interpreted as “Vol .” be the Stanford tagger. The Excel document contains the confusion matrices used to infer these results.

By accuracy within domain, the Stanford tagger performs significantly better than that of the Brill or HMM replacement taggers. The Stanford taggers were able to achieve accuracies of ~95% in contrast to accuracies of ~80%-85% of Brill and HMM taggers. Brill taggers do not significantly benefit from replacement. HMM models differ significantly given replacement, models without replacement have accuracies of below 30%.

Accuracy across domain uniformly drops in all models. Stanford models suffer the least, incurring a drop of ~2%-3%. Brill taggers incur ~5% drop in accuracy (to ~80%) consistently either with or without replacement. HMM models have consistently differing performances given their training dataset across replacement and no-replacement models. The accuracy of the first domain's model is consistent, between domains accuracy varies about ~1%. The accuracy of the second model varies more drastically, without replacement, accuracy drops from ~35% to 18%, and with replacement from ~87% to ~77%.

For brevity, I contrast the brill no-replacement baseline only. Between the baseline, the trigram back off tagger versus the brill tagger, accuracy only differs by a ~1%. The lack of differences may be due to the strength of the baseline tagger over rules. If n-gram models constitute a collection of rules to follow then perhaps the candidate choices under this back off model is already quite constrained.

2 Assignment Part Two

My error analysis is focused on the Pivot tables within each tab of the Excel file. The correct tagging is labelled column-wise, and the estimated tagging is row-wise. This tabulation allows us to analyze false positives row-wise and false negatives column-wise.

Stanford Tagger
One notable source of error within the Stanford tagger comes from its preprocessing. Rounded brackets from text are replaced by the tagger with a token and generally labeled as nouns. Errors are muted but concentrated on mis-tagging between nouns-verbs-adjectives. For example, "American" and "Understanding" are sometimes nouns and sometimes adjectives.

When going across domains, mis-tagging within classes become more common indicating that the overall tagging is still sound. For example, the Domain 2 tagger on Domain 2 makes a significant of misclassifications within verbs (48 -VBD / 317 - VBN).

When encountering out of vocabulary words, the accuracy rate is still quite high (above 80% in all cases). Out of vocabulary incurs a similar error profile as going across domains, intuitively these two are quite related.

A small amount of errors follows other errors, also errors in the middle of sentences do not appear to effect the tagging after them. The errors appear to be independent of each other, and/or that the Stanford tagger appears to be quite robust.

The Stanford tagger appears to balance precision and accuracy well. There is no strong imbalance of false positives and false negatives.

Brill Tagger
Brill taggers without replacement incur some of the same intra-class impreciseness (e.g. NN vs NNP) and ambiguity between noun-verb-adjectives. In the Brill tagger without replacement, there are two systemic errors. A preference for nouns and misclassification within verbs and nouns. The preference for nouns can be intuitively explained by the default tagger. In fact all words not encountered are tagged as nouns.

The misclassification is quite interesting in its patterning. Across all four tests, errors within verb classes form an "X" shape on the confusion matrix / pivot plot. This suggests there is consistent rule that is making these verb-subtype to another verb-subtype misclassifications. Examining the errors of VBN tagged as VBD in "brilO1_D2", suggests these may be due to trade offs when the model is learning its rules. Looking at this set of misclassifications, it is always due to words ending in "ed" or "t". The word "had" was encountered 101 times in the training set and all repeated tokens show consistent tagging. Comparing within and out of domain little to moderate influence on this pattern. One item to note is that these misclassifications are all on items that were seen in the training set.

When encountering out of vocabulary words, the brill taggers default to the default tagger hence all are assigned the noun tag "NN".

The Brill taggers using replacement exhibit the same general trends, however errors are more diffused within misclassifications within a class. For example, comparing between the same training and test sets of the two methods, the "X" shape within verb misclassification now forms a box around all verbs. Examining the out of vocabulary words, the tagging is more diverse and modesty more accurate.

Errors within the Brill tagger appear to incur some confounding. In contrast to the Stanford tagger, there are strings of two to four errors in a row or sequences where a single correct term separates four to six errors.

Brill taggers and their baseline have some issues with recall given that some misclassifications are due to systemic and default tagging.

HMM Tagger

Hidden Markov models without replacement structurally assigns most tags to “JJ” or “NNP” depending on the model. Intuitively, the assignment appears to be the case for zero probability assignments. Two trends are that all non-encountered words are assigned to this “JJ” or “NNP” tag and that sequences of errors often take the form of long “JJ” or “NNP” tagged sentences. Errors outside assignment to this are sparse. There is some similarity to the Brill tagger in the inter-class misidentification of verbs and nouns. Nearly all errors confound, there is a strong locality of tags matching correctly. Furthermore, the sentence must terminate before any correct tags that are not the zero-probability tag are encountered.

Hidden Markov models with replacement generates similar confusion matrices as the Brill taggers with replacement. Errors are more diffuse into noun-verb-adjective pairings and within verb subtypes just as with Brill taggers with replacement. Assignments of out of vocabulary words are much more diverse. Excluding the term for used as the zero-probability class, a modest amount of the assignments is correct with preference for larger classes (e.g. Nouns). The replacement process added some robustness to the model. In contrast to without replacement, the model can error on OOV words and continue and OOV words only infrequently generate error cascades.

3 Assignment Part Three

For the following analysis, please refer to the data analysis Excel file suffixed “Part_3”. For brevity, I will report on the general trends between the test evaluations recorded. I’ve only trained the models using replacement as shown before, they either improve or do not harm the accuracy of the models.

I encountered an issue with importing the data for the Stanford models into Excel. The csv file generated by my script is correct, it has the right number of items as per the tokens tagged by the Stanford tagger. However, something related to either the encoding or the character set breaks Excel’s ability to import the entire file into a spreadsheet. As such I’ve performed my analysis of the Stanford tagger on a limited range of cells I was able to copy and paste over {rows: 1-3800, 5000-end}.

Overall the same general trends appear when testing upon the ELL file as with out of domain files. The Stanford tagger achieves the highest overall accuracy (90% Domain, 95% ELL) with the baseline, Brill, and HMM taggers about the same (80% Domain, 90% ELL). One notable feature of the Stanford tagger is its high accuracy in OOV words (75%) compared to that of the other taggers (35%).

The general pattern of errors are akin to those between domain 1 and 2. There appears to be a moderate increase in the diversity and number of errors as indicated by more populated cells in the pivot tables.

I suspect errors of out of domain models to be largely due to the relative amount of OOV words and tokens. Comparing the OOV tokens of differing models, an ELL trained model has an OOV count of ~300 whereas a domain model has more than ~1000. Some errors such as misspellings and odd punctuation move tokens into the unknown token category where the accuracy rate is substantially lower. Some observable instances are “!!”, “…”, “t”, “sented”, “technologys”. Comparing other word counts, in general, of the ~9000 tokens used models that trained on the ELL training dataset had ~1000 additional tokens where they were seen more than five times in the training text. Words seen more than four times start to achieve a very high accuracy rate (90%).

The hidden Markov models exhibit error cascades on sentence fragments with odd grammar. For example, “There, it is for journalists to stop …” or “… you purse nearly empty,(when you don’t…”.

In the second example, the transition between the comma and round bracket is likely a zero-probability transition.

4 Conclusion

Overall the Stanford POS tagger proves to be an effective means of achieving high accuracy even when facing unknowns. The Brill and HMM taggers, when pre-processing for unknown tokens, can be a relatively simple but effective means of tagging. If the user doesn't require a high fidelity tag set, its likely that any of the three would satisfy using a reduced tag set. Nevertheless, all models show significant benefits for training on domain specific texts.