APPROACH: MLClassification

0. Introduction

The problem consists of the classification of questions according with a coarse and a fine-grained questions' taxonomy. Training and development sets are given, a test set will be released for further evaluation of the model.

1. Description of the Model

Firstly, the **dataset** is imported and suitably handled.
Then, the **preprocessing** takes place. For the coarse taxonomy, the text is examined per question - with *nltk*'s *pos_tag* and *ne_chunks* - and the chunks of words that correspond to one of the following labels - *person, gpe, gsp, organization, location, facility* - are replaced by their label, numbers of any kind are also replaced by *num*, this is done with the intent of facilitating the coarse label attribution as it does not require to differentiate the questions any further. For the fine taxonomy, this replacing does not happen and the words are simply stored, also per question. Afterwards, word-tokenization proceeds, as well as removal of a customized set of stop-words, created cautiously, observing the data set, according to the most common unnecessary words for the classification at hand. There is also punctuation removal, with a special case for the hyphened words. Stemming is done with *nltk*'s *PorterStemmer*, followed by lemmatization with *nltk*'s *WordNetLemmatizer*, as was found that these, in this order, performed better.
After, the **text is converted to numbers**. The training and testing sets, for coarse and fine taxonomies, are converted to feature vectors with *sklearn*'s *TfidfVectorizer,* as it outperformed *sklearn*'s *CountVectorizer.* It is set to support unigrams and bigrams, as this proved to be the combination with the best results.
Lastly, **training the classifier and making predictions**. The classifier chosen was *sklearn*'s *SVC*, with *linear kernel* and the C set to 1 (default). *Sklearn*'s *OneVsRestClassifier* which separates out all labels and trains multiple *SVM* objects (one for each label) on the given data was added to the model, which improved the results as SVM is innately designed for binary classification. *Sklearn*'s *BaggingClassifier* was also tested with *SVC* but impaired the results, so was removed. Other parameters for the SVC classifier were tried but underperformed those chosen. Other classifiers, namely *MultinomialNB*, *LogisticRegression, RandomForestClassifier*, *DecisionTreeClassifier* were also experimented with, for both coarse and fine taxonomies, but *SVC* showed the best results for both, although it is much slower than the others listed.
Regarding **evaluation metrics**, as has been mentioned throughout the description of the model, many different options were considered along the way. In order to compare the various models, mainly it was considered the accuracy resulting from evaluating them in the development set, but in some cases they were also examined using cross-validation and in others, recurring to *sklearn*'s metric functions.

2. **Accuracy** (resulting from evaluating the model in the development set)

Average - 81.80862250262882 % (Coarse - 87.1713985278654 % Fine - 76.44584647739222 %)
Comparing with the other classifiers tested: *MultinomialNB*: C-78,86%,F-41,43%, *LogisticRegression:* C-81,18%,F-60,46%, *RandomForestClassifier*: C-74,66%,F-51,21%, *DecisionTreeClassifier:* C-71,92%,F-51,84%

3. Short error analysis

Error Percentages in dev set: Average - 18.19%
Coarse: 12.83 | DESC - 20.69% ENTY- 48.49% HUM - 16.26% LOC -13.42% ABBR - 11.11% NUM - 9.7%
Fine: 23.55% | (top 5) code - 100.0% volsize - 50.0% veh - 66.66% gr - 44.83% def - 44.63%
Some classes appear very seldomly in the data. Also, there is a huge class imbalance in the given data set, which impacts the results heavily. Better preprocessing, more focused in the specifics of each class would probably also reduce the error.

4. Bibliography

Speech and Language Processing, Daniel Jurafsky & James H. Martin 2009
https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623
https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/
https://queirozf.com/entries/choosing-c-hyperparameter-for-svm-classifiers-examples-with-scikit-learn
https://scikit-learn.org/stable/modules/cross_validation.html