Report

qa-pipeline-with-learning-YikaiFeng created by GitHub Classroom

Overview

In this assignment, I implemented a QA pipeline with the extension of the tf-idf Featurizer, SVM classifier and MLP classifier. The data set used to train on is quasar-s_train which consists of 37,000 cloze-style questions. Since there is a memory limitation on executing with the whole dataset, for simplicity, I used the first 6000 samples for the training process. After completing training on different featurizers and classifiers, six models were applied to the development dataset, quasar-s_dev, which consists of 3139 questions. Then, the evaluator compared the prediction outputs by the models to the ground truth label and computed the accuracy, precision, recall and F-measure.

Performance Comparison

Features	Classifier	Accuracy	Precision	Recall	F-measure
Count based	MNB	0.062758840395	0.0429797493249	0.062758840395	0.0301701298451
Count based	SVM	0.0745460337687	**0.0685281155679**	0.0745460337687	0.0628445053033
Count based	MLP	0.0302644154189	0.00259956980684	0.0302644154189	0.00335852741917
Tf-idf based	MNB	0.0477859190825	0.00596826147975	0.0477859190825	0.00932774947469
Tf-idf based	SVM	**0.0949346925773**	0.0676321735188	**0.0949346925773**	**0.0640987975182**
Tf-idf based	MLP	0.0296272698312	0.000877775117648	0.0296272698312	0.00170503471182

Analysis

The table above showed several comparisons among two Featurizers and three Classifiers using metrics of Accuracy (a), Precision (p), Recall (r) and F-measure (f).

Count based vs Tf-idf based

Different featurizers performed on datasets have a different influence on a, p, r and f. However the selection of featurizers did not seem to be the main aspect to affect the overall performance. From the table, Count based featurizer outperformed Tf-idf featurizer under MNB and MLP classifier, while in SVM, Count based performed worse than Tf-idf featurizer. Therefore, I assumed that either different classifier has a different preference on the featurizer or the selection of classifiers played a more important role in this QA task than featurizers.

MNB vs SVM vs MLP

The performance of different classifiers is quite varied than each other on the a, p, r and f metrics. Support Vector Machine (SVM) achieved the first place for the average performance under two featurizers and is relatively better using Tf-idf based featurizer. Multinomial Naive

Bayes (MNB) ranked second, while Multi Layer Perceptron performed worst in this QA task. There was an obvious gap between the performance of the three classifiers, therefore I assumed that the selection of classifier is crucial for this specific task. Since SVM worked best here with a linear kernel, so the dataset is likely to be linearly seperable so that linear classifier will have better performance.

Error Analysis

I conducted an error analysis on two extreme cases (all false and all true) for the six models. Following are some example questions and answers where the two cases happened.

```
Question 1 is predicted false for all the six models.
Q: homestead -- laravel homestead is an official pre-packaged vagrant box that provides you a wonderful development-environment without requiring
 you to install @placeholder hhvm a web server and any other server software on your local machine.homestead runs on any windows mac or linux sys
tem and includes the nginx web server php-5 .6 mysql postgres redis memcached and all of the other goodies you need to develop amazing laravel ap
plications .
A: php

Question 2 is predicted false for all the six models.
Q: rinari -- rinari is an emacs minor mode that is aimed towards making emacs into a top-notch @placeholder and rails development-environment .
A: ruby

Question 5 is predicted false for all the six models.
Q: libreplan -- libreplan is a collaborative tool to plan monitor and control projects and has a rich web @placeholder which provides a desktop a
like user-experience .
A: interface

Question 6 is predicted false for all the six models.
Q: sharpziplib -- ziplib sharpziplib is a zip gzip tar and bzip2 library written entirely in c for the @placeholder platform .
A: .net

Question 7 is predicted false for all the six models.
Q: postmark -- postmark helps deliver and track @placeholder emails for web-applications .
A: transactional

Question 8 is predicted false for all the six models.
Q: oracle-manageddataaccess -- the oracle.manageddataaccess is a data-access namespace in @placeholder which is used in contrast with oracle.data
access .
A: odp.net

Question 9 is predicted false for all the six models.
Q: urllib -- python-module providing a high-level @placeholder for fetching data across the world wide web .
A: interface

Question 1662 is predicted true for all the six models.
Q: bluebutton -- blue button is the symbol for a patient @placeholder access to their own data .
A: s

Question 2117 is predicted true for all the six models.
Q: parallelism-amdahl -- amdahl s law also known as amdahl @placeholder argument is used to find the maximum expected improvement to an overall s
ystem when only part of the system is improved .
A: s

==============================
In total
All false: 2600
All true: 2
```

Comparing the prediction results output by the six models, I observed that the six models all got wrong on 2600 questions (defined as tough questions), about 82.83% of the development dataset; and they all got right answers for 2 questions (defined as easy questions), about 0.06% of the development dataset. This turned out that the models did not have good performance on the task. The reasons might be that the data for training is not enough (due to memory limitation, only 6000 out of 37000 training samples are selected), the models are not complicated enough to solve the task (parameter tuning and other model selections can be done to improve the performance).