Task 3: Dish Recognition

Jonathan Chang (jc26)

Overview

The goal of this task is to mine the data set to discover the common/popular dishes of a particular cuisine. Typically when you go to try a new cuisine, you don’t know beforehand the types of dishes that are available for that cuisine. For this task, we would like to identify the dishes that are available for a cuisine by building a dish recognizer.

Task 3.1: Manual Tagging

In this task, I chose to focus on the Indian cuisine because it had the fewest number of candidate dish names, and I am not a huge fan of manual effort 😀. Per the instructions, I did a quick pass through the ~150 dish names, and created a column that contained my manual label which I considered "truth". I generally would label any phrase that was not actually a dish name as 0, and even if the phrase could be classified as "Indian," I still labeled them as 0 because as the overview states, we are particularly focused on identifying the dishes that are available for a particular cuisine. However, if a dish could be considered Indian cuisine at all (e.g. coconut chicken or tomato soup), I would assign a 1.

Per the suggestions, I removed all the false positives that I identified (30), and I corrected the labels of all the false negatives (5).

The result of this task was a file called ***Indian_updated.label*** which I placed in the /data directory for SegPhrase section

Task 3.2 Mining Additional Names

Discussion:

I chose to use SegPhrase as opposed to ToPMine or word2vec primarily because it is recommended by the instructions as state-of-the-art. I also liked the idea of my manual effort from Task 3.1 being put to good use. As mentioned in the instructions, "SegPhrase has a classifier to assign a quality score to each phrase candidate based on their statistical features. The classification procedure will be enhanced by phrasal segmentation results. These two parts could mutually enhance each other."

As I understand it, ToPMine works well when we are trying to perform phrase mining without training data, however sometimes a small set of training data may enhance the quality of our phrase mining, and we should be able to improve our results by utilizing SegPhrase with our labeled phrases in conjunction with a constructed Knowledgebase.

SegPhrase is designed for phrase mining with tiny training datasets such as this problem, and it is able to use labels (such as our human annotated labels or from a knowledgebase) to indicate whether a phrase is high quality or not. It then performs classification via the random forest algorithm to construct models that distinguish between quality phrases and poor ones. It also uses phrasal segmentation to determine which phrases are more appropriate and partitions a sequence of words by maximizing likelihood. Overall, SegPhrase is fits the bill for pattern mining for this task because 1) we have a small training set 2) we need some way to perform feature extraction and distinguish how informative or quality a phrase is, and SegPhrase can utilize IDF and mutual information in order to mine relevant dishes and filter out irrelevant dishes.

Setup:

To set up SegPhrase on my Mac, I first cloned the repo from: https://github.com/shangjingbo1226/SegPhrase. I also needed to do some additional work to be able to build SegPhrase with the provided Makefile which included, per a helpful Github Issues Post referenced below:

1. Installing Xcode on my Mac (https://stackoverflow.com/questions/19580758/gcc-fatal-error-stdio-h-no-such-file-or-directory)
2. using brew to install gcc49
3. Changing the g++ variable in my Makefile to reference my gcc path (export CXX = /usr/local/Cellar/gcc@4.9/4.9.4_1/bin/g++-4.9)
4. created a conda environment for python 2.7 and installed scikt-learn and nltk (which i need to activate when running the shell scripts)

Next, I used Wikipedia to generate a Knowledgebase of common Indian dishes https://en.wikipedia.org/wiki/List_of_Indian_dishes#Unsorted. I added 250 Indian dish names I found from that Wikipedia article and appended it to the bottom of /data/EN/wiki_quality.txt (from AutoPhrase repo) to create /data/wiki_labels_quality_append_IndianDish.txt.

I then proceeded to modify the provided train_toy.sh script, by changing some of the following parameters. For RAW_TEXT, I used the Indian.txt from Task 2 where we have all the Yelp reviews related to Indian cuisine. For KNOWLEDGE_BASE, I used the knowledgebase created from the Wikipedia article as described above.

```
RAW_TEXT='data/Indian.txt'
AUTO_LABEL=0
WORDNET_NOUN=0
DATA_LABEL='data/Indian_updated2.label'
KNOWLEDGE_BASE='data/wiki_labels_quality_append_IndianDish.txt'
KNOWLEDGE_BASE_LARGE='data/wiki_labels_all.txt'
```

Then I could just run `./train_IndianDish.sh`

Take 1

First, I tried using the following data labels as described above. We see that although SegPhrase does a pretty good job with output phrases, it still has quite a few false positives and struggles with some of the ambiguous categories where the phrase is related to Indian culture, but not specifically cuisine.

Take 1	
fried_rice	0.998857168
tandoori_chicken	0.99792519
rice_pudding	0.997713903
indian_cuisine	0.997379549
goat_curry	0.997351527
basmati_rice	0.997261925
hot_sauce	0.99715519
rogan_josh	0.996804812
gulab_jamun	0.996804812
flat_bread	0.996391304
lamb_vindaloo	0.996236284
the_naan_was	0.995780933
ice_cream	0.995741925
chicken_tikka	0.995741925
bhindi_masala	0.995527236
tikka_masala	0.995261925
south_india	0.995075258
mother_india	0.994941925
tomato_soup	0.994107588
garlic_naan	0.993636662
chicken_tikka_masala	0.991797101
chick_peas	0.99135218
coconut_chicken	0.990149853
veggie_korma	0.98840519
curry_house	0.98840519
south_indian	0.986716284
masala_dosai	0.985939327
lunch_buffet	0.985713903
date_night	0.985276062
curry_houses	0.985276062
mango_chutney	0.97840519
india_oven	0.975728039

Take 2	
rice_pudding	0.999156153
chicken_tikka_masala	0.999156153
butter_chicken	0.999156153
indian_cuisine	0.997890711
tandoori_chicken	0.99674297
palak_paneer	0.996644708
basmati_rice	0.995076304
tikka_masala	0.993038088
chicken_tikka	0.993038088
fried_rice	0.989795259
flat_bread	0.986854083
the_naan_was	0.984043204
gulab_jamun	0.982393454
ice_cream	0.979565057
the_indian_sampler	0.979370539
coconut_chicken	0.97706684
hot_sauce	0.97684241
south_india	0.976455586
mother_india	0.976359002
garlic_naan	0.973038088
rogan_josh	0.969384489
tomato_soup	0.967324741
lamb_vindaloo	0.966972855
goat_curry	0.966817936
spice_level	0.962506452
chick_peas	0.96123202
lunch_buffet	0.95382734
paneer_tikka_masala	0.939596421
india_oven	0.895114952
eggplant_dish	0.895114952
chana_masala	0.894532005
weekend_buffet	0.894458144

mantra_masala	0.967741925
saag_paneer	0.967261925
shrimp_vindaloo	0.965939327
channa_masala	0.965276062
the_indian_sampler	0.963941898
yogurt_sauce	0.95840519
star_review	0.95840519
brown_rice	0.95840519
eggplant_dish	0.956391304
chicken_wings	0.956299927
tandoori_times	0.956129549
customer_service	0.956129549
white_meat	0.955939327
chicken_tikki_masala	0.954611021
masala_dosa	0.947831527
mutter_paneer	0.947741925
mattar_paneer	0.947741925
coconut_chutney	0.947741925

12 false positives

chilli_chicken	0.894458144
small_portions	0.876854083
lunch_hour	0.868971602
bhindi_masala	0.866854083
menu_items	0.850478774
saag_paneer	0.845609219
mango_ice_cream	0.84359657
south_indian_dishes	0.843173786
naan_bread	0.838366902
buffet_style	0.832993652
india_masala	0.829482784
medium_spice	0.825114952
chicken_tikki_masala	0.824031637
chicken_tika_masala	0.817067351
chicken_pakora	0.816362192
lentil_soup	0.809795259
masala_dosa	0.782792875
spice_levels	0.781108777

13 false positives

Take 2

I wanted to see if I could improve my results so I created a new file called: Indian_updated2.label. Here I decided to make two changes:

- I corrected the false positive dish names instead of removing them (I noticed that the false positives made their way back in)
- I added some new positive labels based on a Google Search of "indian dishes." (I took the top 25)
    1. biryani
    2. chicken tikka masala
    3. samosa
    4. butter chicken
    5. tandoori chicken
    6. panipuri
    7. naan
    8. tulab jamun
    9. chapati
    10. dosa
    11. laddu
    12. momo
    13. rasgulla
    14. papadum
    15. chana massala
    16. korma
    17. palak paneer
    18. khichdi
    19. dkhola
    20. pakora
    21. idli
    22. appam
    23. pav bhaji
    24. raita
    25. kheer

I changed **DATA_LABEL='data/Indian_updated2.label'** and ran ./train_IndianDish.sh again. Now, we take a look at the top 50, and there is no noticeable improvement, although we do mine a few other dishes that may not have been discovered in our first run. I suppose that I could continue to iterate and improve my results by manually correcting the output results for the false positives which is what makes SegPhrase awesome because it can be an iterative process with a feedback loop for improvement.

References

- Jialu Liu*, Jingbo Shang*, Chi Wang, Xiang Ren and Jiawei Han, "**Mining Quality Phrases from Massive Text Corpora**", Proc. of 2015 ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'15), Melbourne, Australia, May 2015. (* equally contributed, slides)
- SegPhrase Setup: https://github.com/shangjingbo1226/SegPhrase/issues/4