Cuisine Classification by Ingredient Embeddings

Chenxin Xiong (168449)
CS 390DD text mining report
KAUST

October 15, 2019

1 Introduction

As is known to all, food is essential for human's life, and there are lots of different categories of cuisines around the world, like Chinese, Korean and Arabic. A specific cuisine is characterized by many sets of special ingredients which are firmly dependent on the cultures and geographic regions [5]. Without considering the cooking techniques and instructions, how to tell or classify accurately which cuisine one set of ingredients belongs to by machine based on some machine learning techniques has been drawn some interests and attentions since several years ago, such as automatic recipe cuisine classification by ingredients using Support Vector Machine (SVM) [4], cuisine prediction based on ingredients using Tree Boosting Algorithms (TBAs) [2].
However, the methods used to do cuisine classification before usually are classical and traditional machine learning techniques, like decision tree, random forest, multi-nomial logistic regression and K-means clustering, and the ingredients are encoded into binary features by one-hot format or represented by a learnable or defined feature matrix.
These classical methods might have very good performance, but one of the obvious drawbacks is that we cannot analyse the underlying connections between two ingredients according to their corresponding feature representations. For example, the feature vector of ingredient "avocado" is expected to be very close to that of ingredient "guacamole", in other words, in feature space, the "distance" between two similar ingredients should be small.
In order to get better understanding and know some interesting relations between different ingredients, the idea of Ingredient embeddings was come up with under the inspiration of word embeddings that are produced by Word2vec [3], which was proposed in 2013 and has a large impact due to many apparent advantages in comparison to earlier algorithms.
Basically, there are two main Word2vec architecture that are used to produce a distributed representation of words, one is Continous-Bags-of-Words (CBOW) and the other is Continous-Skip-ngrams (Skip-gram).
Thus, Ingredient embeddings can be obtained using either CBOW or Skip-gram model. In this project, Skip-gram model is selected. After get the Ingredient embeddings matrix, fully connected layer and convolutional neural network (CNN) are used to do the following cuisine classification task respectively.

2 Dataset

The dataset used in this project was provided by Yummly on Kaggle in 2015 when a competition named *what's cooking* launched. In the dataset, the recipe id, the type of cuisine, and the list of ingredients of each recipe (of variable length) are included. The data is stored in JSON format and the total number of training samples is 39774. Before process the data, some knowledge about the data should be explored first. Some training samples as Figure 1 shows.

	cuisine	id	ingredients
0	greek	10259	[romaine lettuce, black olives, grape tomatoes...
1	southern_us	25693	[plain flour, ground pepper, salt, tomatoes, g...
2	filipino	20130	[eggs, pepper, salt, mayonaise, cooking oil, g...
3	indian	22213	[water, vegetable oil, wheat, salt]
4	indian	13162	[black pepper, shallots, cornflour, cayenne pe...
5	jamaican	6602	[plain flour, sugar, butter, eggs, fresh ginge...

Figure 1: Training data examples

Then, visualize the counts of every unique cuisine. The top three cuisines which have the most number of training samples are Italian, Mexican and Southern US.

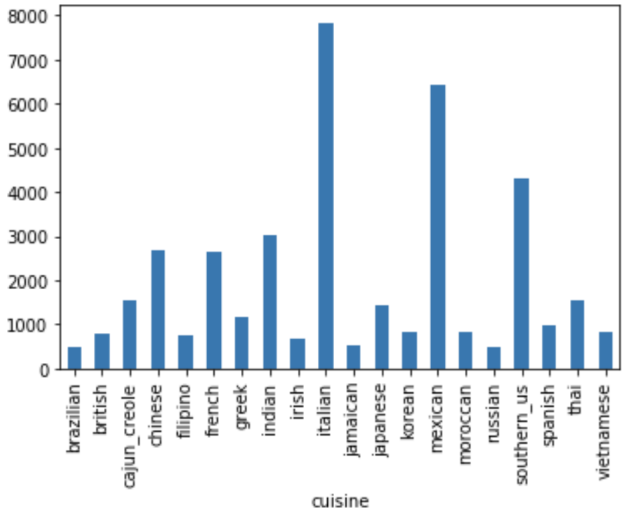

Figure 2: Twenty various cuisines

Next, count the number of ingredients each training sample contain. From Figure 4, which can be observed is that the number of ingredients of most training examples concentrates on the range of [3, 20].

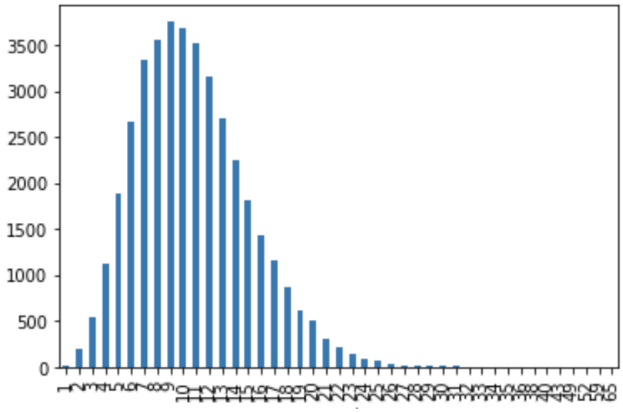

Figure 3: The number of ingredients counts

3 Experiments

The experiments part consists of two stages. The first stage is getting the ingredient embeddings based on Skip-gram model and the second stage is do cuisine classification using the pretrained embeddings from the first stage.

3.1 Skip-gram model

Skip-gram model is used to get the ingredient embeddings. The values determined for each hyperparameter of the model are shown in the table below (many values have been tried and the values displayed in the table make the model have the best performance so far).

embedding dimension	100
batch size	32
window size	5
min count	5
iterations	10
learning rate	0.01

Where $mincount = 5$ means if the occurrence times of one ingredient is less than 5, then this ingredient will be directly discarded from the initial ingredients table.
With the concern of the situation, that the most frequent words usually provide less information value, which commonly happens in Word2vec when the corpora is huge.

```
[('salt', 18049),
 ('onions', 7972),
 ('olive oil', 7972),
 ('water', 7457),
 ('garlic', 7380),
 ('sugar', 6434),
 ('garlic cloves', 6237),
 ('butter', 4848),
 ('ground black pepper', 4785),
 ('all-purpose flour', 4632)]
```

Figure 4: Top ten most frequent ingredients in the dataset

A simple but useful approach called subsampling is able to counter the imbalance between rare and frequent ingredients [3].
For each ingredient in the training set (exclude the ingredients which already are discarded due to too less occurrence) is kept with probability computed by the formula:

$$p(I_i) = \sqrt{\frac{t}{f(I_i)}} + \frac{t}{f(I_i)} \tag{1}$$

Where $f(I_i)$ represents the frequency of ingredient I_i and t as a threshold is determined by value 10^{-4} in my project's case.

3.2 Cuisine Classification

As mentioned before, two different architectures are used to do cuisine classification task, FC and CNN. Before train the models, I remove all the samples in the training set whose number of ingredients is out of the range of [3, 20] for doing less padding work and saving the space and then randomly divide the samples into training set and validation set with the ratio of 4 to 1.

	number	ratio
discarded samples	1219	
training samples	30844	80%
validation samples	7711	20%

Besides, the size of the ingredient embedding matrix obtained from Skip-gram model is 3257×100.

3.2.1 Fully connected model

After use Skip-gram model to get the ingredient embeddings, the most simple and straightforward thought is to use a fully connected layer to do cuisine classification.
The design of FC model is quite simple, one embedding layer and one fully connected layer which is of size 20 with softmax active function as the output, as Figure 5 shows.

```
CuisineModel(
  (embedding): EmbeddingBag(3258, 100, mode=mean)
  (fc): Linear(in_features=100, out_features=20, bias=True)
  (soft_max): Softmax(dim=1)
)
```

Figure 5: Fully connected model

To be more specific, the input of the FC model is a set of indexes where each index indicates the row number of each ingredient's corresponding embedding in the embedding matrix. Although the number of ingredients of each training sample is varied, padding the input is not necessary cause each training sample will be represented by a 100-dimensional vector after the embedding layer. What does the embedding layer do actually is to average the embeddings of the ingredients.

3.2.2 CNN model

Based on the demonstration that CNN is able to be applied to do text classification and have pretty well performance [1], CNN is selected to be compared with the FC model in my project. The architecture of CNN model is shown by Figure 6.

```
CuisineModelCNN(
  (embedding): Embedding(3258, 100)
  (conv1): Conv2d(1, 8, kernel_size=(3, 100), stride=(1, 1))
  (maxpool1d): MaxPool1d(kernel_size=18, stride=18, padding=0, dilation=1, ceil_mode=False)
  (fc1): Linear(in_features=16, out_features=20, bias=True)
  (soft_max): Softmax(dim=1)
)
```

Figure 6: Fully connected model

One of the differences between these models is that the input of CNN model needs padding whereas FC model doesn't need. Because the maximum number of ingredients is 20, the padding operation can be realized by adding enough number of padding index whose value is equal to the max index plus one to make the length of input equal to 20.

4 Results and Analysis

Using PCA to reduce the dimension of ingredient embeddings and visualize 100 number of ingredients, as Figure 8 shows. What can be observed is that the distances among some similar ingredients are small, like lime juice and lemon, shrimp and fish.

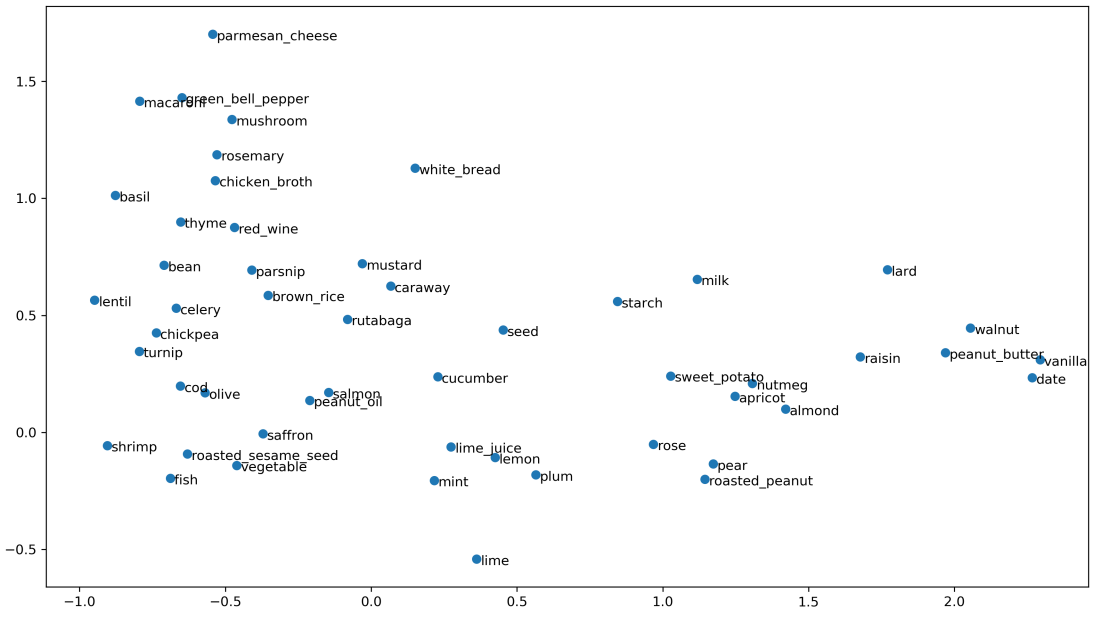

Figure 7: Partial ingredients visualization

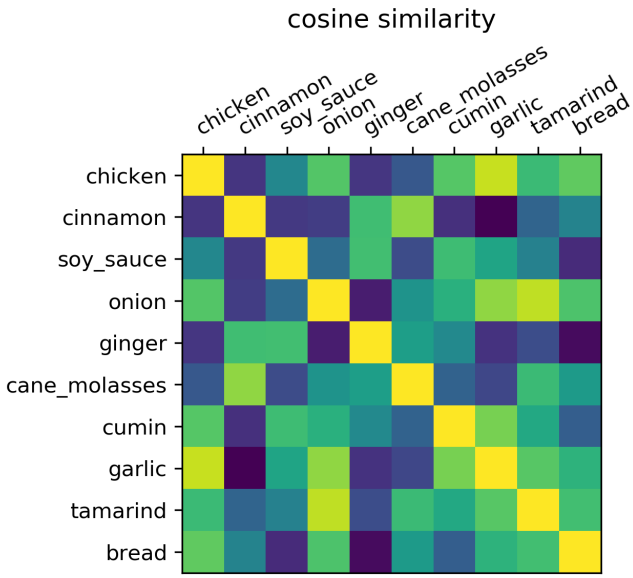

Figure 8: Partial cosine similarity matrix

The embedding of each recipe can be obtained by average the embeddings of ingredients and then is able to be visualized as well, however just some embeddings of recipes from one specific cuisine are quite close.

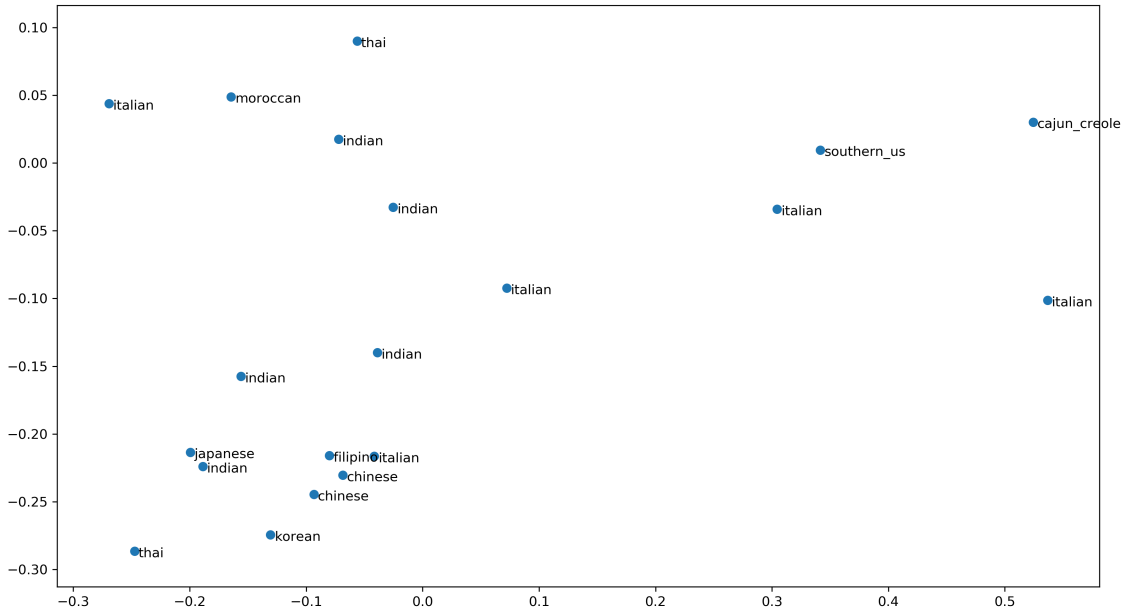

Figure 9: Partial recipes visualization

About the classification results, the validation accuracy of FC model and CNN model are 23% and 53.1% respectively. The performance of CNN model is better than FC model, although both of them perform very bad in comparison to other classic algorithms.

References

[1] Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[2] R. Kumar, M. A. Kumar, and K. Soman. Cuisine prediction based on ingredients using tree boosting algorithms. *Indian J. Sci. Technol*, 9:45, 2016.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[4] H. Su, T.-W. Lin, C.-T. Li, M.-K. Shan, and J. Chang. Automatic recipe cuisine classification by ingredients. In *Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: adjunct publication*, pages 565–570. ACM, 2014.

[5] Wikipedia contributors. Cuisine — Wikipedia, the free encyclopedia, 2019. [Online; accessed 14-October-2019].