Imperial College London

HyperFoods: Machine intelligent mapping of cancer-beating molecules in foods

Luís Artur Domingues Rita

Thesis to obtain the Master of Research Degree in

Biomedical Research

Supervisors: Prof. Kirill Veselkov

Prof. Michael Bronstein

Examination Committee

Chairperson:

Supervisor:

Members of the Committee:

March 2020

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of Imperial College London.

Luís Rita

Preface

The work presented in this thesis was performed at the Imperial College London during the period _____, under the supervision of Prof. Kirill Veselkov and Michael Bronstein.

Abstract

HyperFoods | Ingredient/Recipe Networks | Data Visualization | Web Application

Acknowledgments

Table of Contents

List of Figures

Não foi encontrada nenhuma entrada do índice de ilustrações.

List of Tables

Não foi encontrada nenhuma entrada do índice de ilustrações.

List of Algorithms

Não foi encontrada nenhuma entrada do índice de ilustrações.

List of Acronyms

PCA	Principal Component Analysis
T-SNE	T-Distributed Stochastic Neighbour Embedding
JSON	JavaScript Object Notation
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
IoU	Intersection over Union

1. Introduction

There are many factors known that influences individual's health. Physical exercise, sleeping, nutrition, heredity, pollution, among other external factors. Being nutrition one of the easiest and biggest modifiable factors in our lives, it is not surprising that small changes can actually bring meaningful outcomes.

Having our diets strong cultural ties, it is possible to identify around the world a big number of cuisines. The most common ingredients in each one is closely related to characteristics of the region, such as the climate. This plays a big influence in the availability of each of the components present in the local recipes.

Some molecules are known to have a positive effect in health, namely, in fighting cancer. Being able to identify which ingredients contain the higher concentrations, may help us treating and preventing a broad range of diseases. Further on, being able to include these ingredients in tasty and affordable meals, it can promote a shift on habits in the population. In a world where fast food consumption is rising, it is clear that additionally to the two previous points, speed of preparation is also an important issue.

With increasingly more data being made available online, whether from research studies or simple web applications, it is an opportunity to analyse it and create new recipe recommendation systems that not only take into account factors like anticancer properties, but also flavour and nutritional content. This would empower the user to take better decisions when preparing his next meal.

1.1. Objectives

The aim of the project was to develop the algorithms and retrieve the ingredients, quantities and cooking processes from the Recipe1M+ dataset of recipes. Query API from FoodData Central to extract caloric information for each ingredient/recipe. To add flavour molecule information to each ingredient/recipe using FlavorDB. Label each recipe from Recipe1M+ dataset to a cuisine after training an SVM model in a dataset where this information was known. Calculate the number of anticancer molecules present in each recipe and present an ordered list including the ones with the higher value. Cluster ingredients and recipes in terms of their similarity, considering how often 2 ingredients appear together in the dataset.

Finally, to build a web application that retrieves complete recipes from images, suggests new healthier ones based on the previous and that recommends new ingredients in substitution to the originals.

1.2. Thesis Outline

In the Background section, it will be discussed the fundamental concepts used along the project. Methodology will detail and justify alongside, the different approaches that were considered to obtain the results present in the respective section – Results. Finally, in the Conclusion, the aim of the thesis is recalled, it is discussed whether its goals were achieved, main difficulties and suggestions for future work are provided.

2. Background

This section will start by a thorough analysis on the biological aspects important for the understanding of the project. Later on, all the modelling techniques and mathematical models that were used are carefully explained. Finally, all the techniques and underlying tools are presented.

2.1. Cancer

The XXI century disease. Belonging to a broader spectrum called tumours, cancers are a subtype where uncontrolled cell division occurs and has potential to spread to different tissues. In opposition, benign tumours are confined to a certain organ. Existing a close correlation between ageing and the loss of function of some regulatory pathways, as lifespan is increasing, the incidence of the disease is following the same trend.

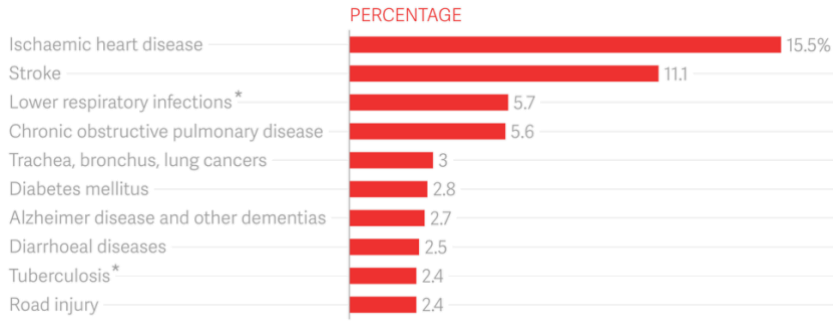

Figure 1 Cancer is the fifth main cause of death worldwide. Data collected by WHO between 2000-2015 from a population of over 90 000 people, in WHO member states. Adapted from [1].

2.2. Dimensionality Reduction

As part of the efforts to visualize high dimensional data, Principal Component Analyses (PCA) and T-Distributed Stochastic Neighbour Embedding (T-SNE) methods are commonly used. They both aim to find the components that contain the higher variability of the data. Although some of information will inevitably be present, they are very useful when trying to visualize and analyse high-dimensional vectorized representations of it.

Although PCA and T-SNE techniques will be detailed below, the first method aims to separate points as far as possible, while the second works by grouping points as close as possible.

PCA

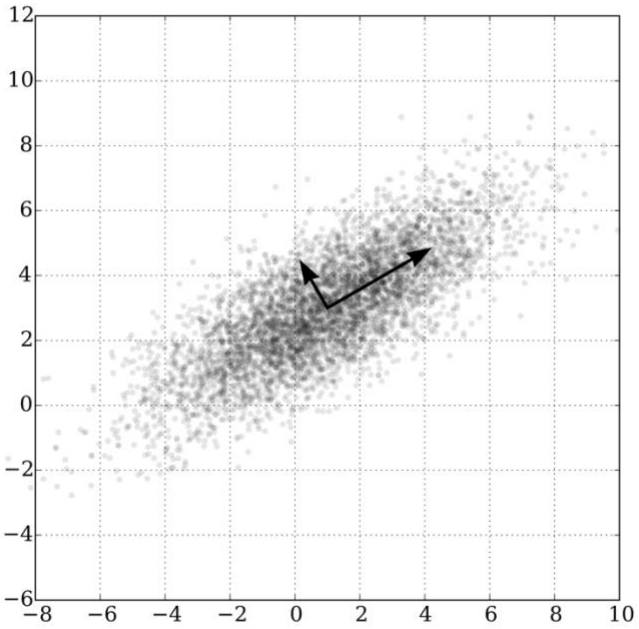

Figure 2 - Application of PCA [2].

T-SNE

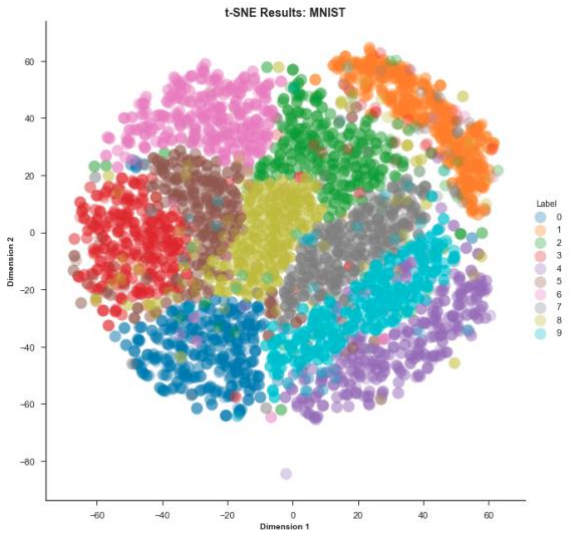

Figure 3 – Application of T-SNE [3].

2.3. Community Finding

Unsupervised learning algorithms are a type of approach that allows us to define clusters in a given network, based on the similarity of the nodes.

2.3.1. Louvain

$$M = \frac{1}{2m}\sum_{i,j}(A_{ij} - p_{ij})\delta(c_i, c_j) = \frac{1}{2m}\sum_{i,j}(A_{ij} - \frac{k_i k_j}{2m})\delta(c_i, c_j) \quad (1)$$

Modularity Optimization

$$\Delta M = \left[\frac{\Sigma_{in} + 2k_{i,in}}{2m} - \left(\frac{\Sigma_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right] \quad (2)$$

$$\Delta M = \frac{k_{i,in}}{m} - \frac{2\sum_{tot} k_i}{(2m)^2} \Leftrightarrow \Delta M m = k_{i,in} - \frac{\sum_{tot} k_i}{2m} \quad (3)$$

Community Aggregation

2.3.2. Infomap

$$L = \log_2 N \quad (4)$$

$$H(X) = -\sum_{1}^{n} p_i log\ (p_i) \quad (5)$$

$$L(M) = qH(Q) + \sum_{m=1}^{n_m} p_{\odot}^{m} H(P_m) \quad (6)$$

$$H(Q) = -\sum_{i=1}^{m} \frac{q_i}{\sum_{j=1}^{m} q_j} log\left(\frac{q_i}{\sum_{j=1}^{m} q_j}\right) \quad (7)$$

$$H(P_m) = -\frac{q_i}{q_i + \sum_{\beta \in i}^{m} p_\beta} \sum_{i=1}^{m} log\left(\frac{q_i}{q_i + \sum_{\beta \in i}^{m} p_\beta}\right) - \sum_{\alpha \in i} \frac{p_\alpha}{q_i + \sum_{\beta \in i}^{m} p_\beta} log\left(\frac{p_\alpha}{q_i + \sum_{\beta \in i}^{m} p_\beta}\right) \quad (8)$$

$$L(M) = \left(\sum_{m \in M} q_m\right) log\left(\sum_{m \in M} q_m\right) - 2 \sum_{m \in M} q_m \, log(q_m) - \sum_{\alpha \in V} p_\alpha \, log(p_\alpha) + \sum_{m \in M} (q_m + \sum_{\alpha \in m} p_\alpha) \, log\left(q_m + \sum_{\alpha \in m} p_\alpha\right) \quad (9)$$

2.3.3. Density-Based Spatial Clustering of Applications with Noise

It was Initially proposed by Martin Ester, Hans-Peter Kriegel, Jörg Sander and Xiaowei Xu, in 1996. One of its main features is the ability to handle with noisy data points, being able to identify them as outliers.

2.3.4. Mean Shift

This algorithm was originally presented by Fukunaga and Hostetler in 1975.

Given a set of discrete data, this algorithm locates the maxima of a given density function.

2.4. Natural Language Processing

Although online datasets and APIs contain structured information that can be easily retrieved, most online sources do not have such an organized structure. This means, algorithms that are able not only to extract data, but also to get its context are needed.

Word Embedding

Developed by a team of researchers led by Tomas Mikolov at Google, Word2Vec is a specific category of models that produces word embeddings. They take as input a corpus of text and produces a vector space, where each word is mapped into a vector. One of the features of the goals of this approach is that words that appear more often in similar contexts will be mapped into vectors which Euclidean distance is shorter.

2.5. Inverse Cooking Facebook's Algorithm

Efforts to retrieve recipes from images of foods previously prepared led to the development of the Inverse Cooking Algorithm. It not only retrieves the ingredients predicted to be present, but also their quantities, a full set of instructions for preparation and it suggests a title for the recipe.

2.6. HyperFoods

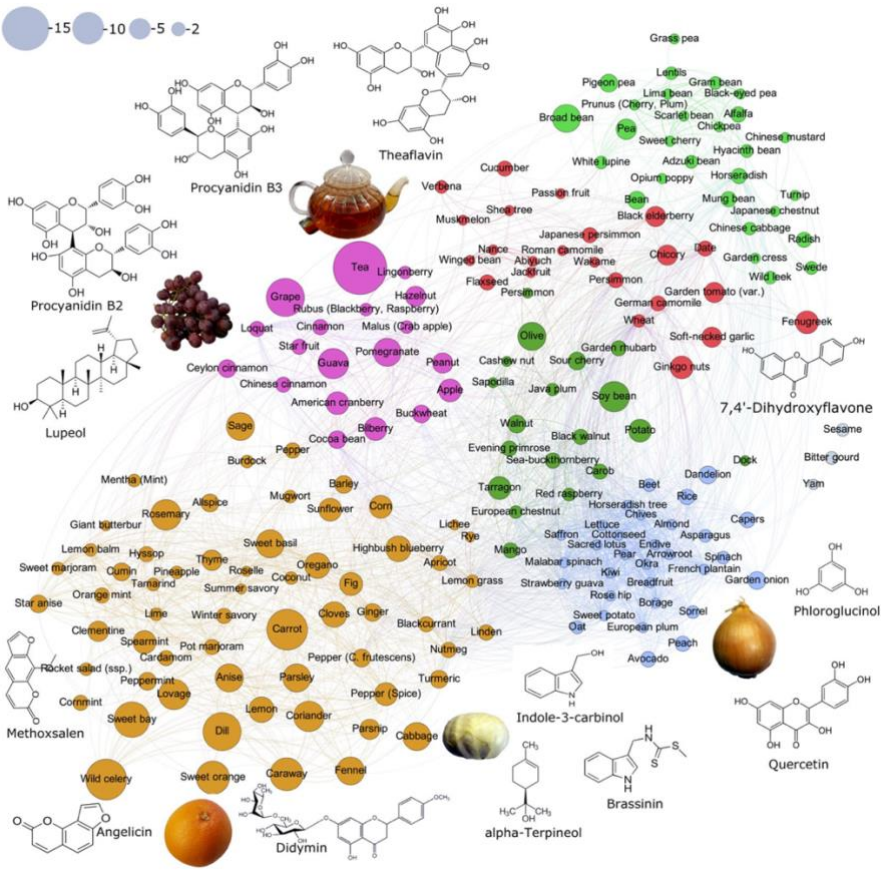

Figure 4 - The size of each node represents the number of anticancer molecules present in the given ingredient. The width of each link depicts the number of anticancer molecules that are shared by two ingredients [5].

2.7. Flavor

(FlavorDB)

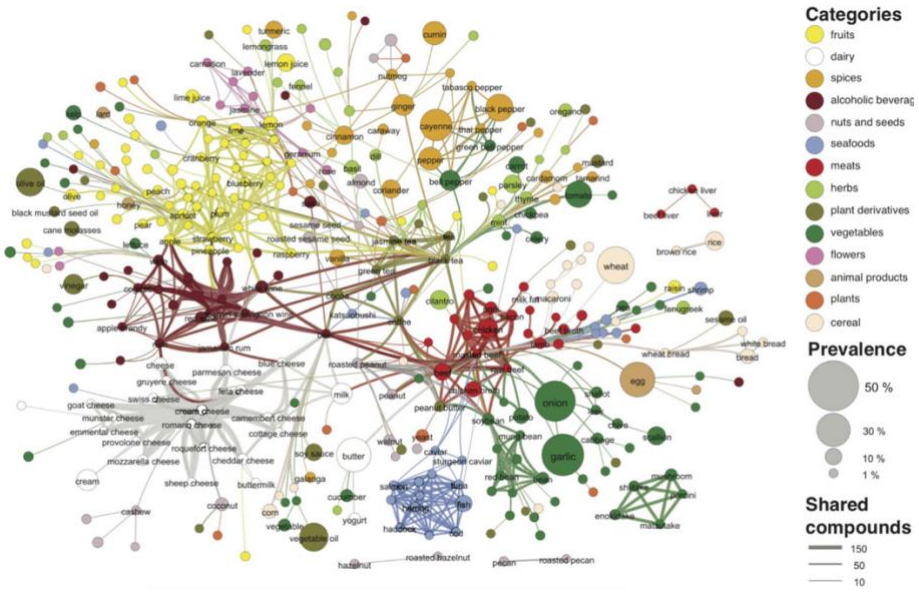

Figure 5 – The size of each node represents the prevalence of each ingredient in the dataset. The width of each link depicts the number of favour molecules that are shared by two ingredients [4].

2.8. Nutritional Content

(FoodData Central)

3. Methodology

Introduction covered all theoretical concepts important for this section.

3.1. Recipe1M+ Dataset

This dataset contains over one million recipes. It is defined in two JSON files:

layer1.json

The first organizes attributes to each recipe one ID and, then, defines the ingredients present, instructions, title, URL and attributes one last value so that each recipe belongs to the train, validate or test set. These three distinctive groups were used for training, validating and testing the *Inverse Cooking* algorithm developed by the Facebook Research Team.

layer2.json

This second file includes each recipe IDs present in *layer1.json* and a set of URLs pointing towards images present in the websites the recipes were scrapped from.

ingr_vocab.pkl & *instr_vocab.pkl*

Attached to the same dataset are two pickle files containing a vocabulary file to the ingredients (*ingr_vocab.pkl*) and instructions (*instr_vocab.pkl*) of all recipes. To facilitate posterior analysis, they were unpickled to a paragraph separated text file.

Once Recipe1M+ dataset was scrapped from 22 (??) websites open to any user, it is expected the presence of misinformation, typos, non-Latin characters, among others. Although Facebook Research team did a significant work in harmonizing it, the presence of incoherent information was still a hurdle to retrieve the amount of each ingredient present, the units it was expressed, and the cooking processes associated to each recipe.

First, for each recipe, each ingredient and instruction field were iterated and the ones only containing numbers or punctuation were removed. Recipes with no ingredients and instructions were removed. It was later identified that all recipes retrieved from *food.com* did not have their fractions represented in the dataset. Both numerators and denominators were joint as a single number.

3.2. Kaggle and Nature Dataset

kaggle_and_nature.csv

Dataset containing over 100 000 recipes. Comma-separated file represents a different recipe in each line. The first value of each line is the cuisine the recipe belongs. The remaining are all the ingredients represented in a tokenized way.

Kaggle & Nature dataset contains a vocabulary list named in a different way than in Recipe1M+. A synonymous file was created to match the difference between both.

3.3. Ingredient/Recipe Embedding

In order to be able to compare how often two ingredients appear together in a recipe, as well as how

3.3.1. Principal Component Analysis

3.3.2. T-Distributed Stochastic Neighbour Embedding

3.4. Clustering Recipes/Ingredients

In order to infer groups of ingredients that most often are used in a similar setting, as well as to

3.4.1. Louvain Algorithm

3.4.2. Infomap Algorithm

3.4.3. Density-Based Spatial Clustering of Applications with Noise

3.4.4. Mean Shift

3.5. Inverse Cooking Facebook's Algorithm

3.5.1. Benchmark

In order to test the effectiveness of the algorithm, a benchmark test was performed. Accuracy was calculated using the following metrics: Intersection over Union (IoU) and F1 score.

	IoU	F1
Human	21.36	35.20

Retrieved	18.03	30.55
Inverse Cooking	32.52	49.08

3.6. Visualization Tools

3.6.1. Matplotlib

The most fundamental data visualization library from Python.

3.6.2. Plotly

It distinguishes from the previous by allowing the dynamic representation of datapoints along with the respective labels. By uploading our models to Plotly Chart Studio, one can embed them in an html webpage.

3.6.3. Seaborn

Seaborn is a Python library built on the top of Matplotlib. It is powerful as the latter in representing a high number of datapoints, plus it allows the user to explore new visualization options in a easier way.

4. Results

In this section, the web application available online is first described. Then, the results from benchmarking the Facebook algorithm is presented. After, three Python visualization tools are discussed in terms of their pros and cons. Finally, all community finding algorithms (with varying input) are run in the recipe dataset, to assess which one performs the best in detecting similar ingredients or recipes.

4.1. Web Application

In order to facilitate the analysis and visualization of the results after the execution of the algorithms, a web application was implemented.

Its backend was conceived using Node.js and it runs in a Heroku server which is linked to a GitHub repository (github.com/warcraft12321/HyperFoods) containing all the implementations and documents related to the thesis. This repository was also integrated in Zenodo (DOI:). An image of the app is available at Docker Hub (). The following actions are strictly executed in the cloud:

The frontend is both static and dynamic. The static counterpart uses HTML, CSS and JavaScript. The dynamic execution is managed by Node.js, which runs in the server-side. Several sections were individualized in the interface:

4.2. Community Finding Algorithms

4.3. Benchmark Facebook Algorithm

4.4. Visualization Frameworks

Matplotlib, seaborn, plotly

5. Conclusion

References

Não existem fontes no documento atual.