Recipe Suggestion Tool

CS298 Project Presentation

By

Priya Gangaraju

Agenda

- **Motivation and Project Description**
- Clustering and Implementation
- Integration with Yioop!
- Tests and Results
- Demo
- Conclusion

Motivation

- Current search engines like Google, Yahoo and Bing
  - do not provide recipes search based on ingredient.
  - Result set is huge with irrelevant data.
- Recipe specific search engines like AllRecipes.com, recipepuppy.com –
  - Not generic search engines.
  - Do not show the other ingredients used in the recipes.

Project Description

- Dedicated recipe search based on ingredient within Yioop! Search Engine.
- Recipes specific results showing ingredients needed.
- Used clustering to return relevant results.
- User can search for recipes by typing “ingredient: query term”.
- Tested on 1000 crawled recipes using Yioop!.

Agenda

- Motivation and Project Description
- **Clustering and Implementation**
- Integration with Yioop!
- Tests and Results
- Demo
- Conclusion

Clustering

- Clustering involves grouping of data into clusters such that similar data belong to the same cluster.
- Clustering applications in information retrieval -
  - Search Result Clustering – Clustering search results into categories. Example: Yippy.com
  - Cluster based retrieval – retrieving clusters of data.

Clustering cont'd.

- An important step in clustering is the selection of the distance measure which calculates the similarity between two elements.
- Different distance measures include Euclidean distance, Manhattan distance and Hamming distance.
- Euclidean distance is used for this project.

Clustering using Minimum Spanning Tree

- Data is represented in a minimum spanning tree where edge weight is the distance between two points.
- Clusters are formed by removing the edge with maximum weight.
- Commonly used algorithms for constructing minimum spanning tree –
  - Kruskal's algorithm.
  - Prim's algorithm.

Pseudo Implementation

- Recipes graph is constructed with recipes as vertices.
- Kruskal's algorithm is implemented which involves:
  - List the edges in the increasing order of the weights.
  - Choose those edges with smallest weights such that adding an edge doesn't form a cycle.
- Clustering is done by removing the most weighted edges from the minimum spanning tree.

Implementation of clustering

- For each two recipes unique ingredients in both the recipes are used to construct a vector.
- 0-1 vector is constructed for each recipe – if the ingredient in the vector is used in the recipe, it is marked 1, otherwise it is marked 0.
- Euclidean distance is calculated between vectors constructed.

Implementation cont'd.

- A graph with all the recipes as vertices and the Euclidean distances as the edge weights is constructed.
- Each recipe in the graph will have an edge with every other recipe.
- Minimum spanning tree (MST) using Kruskal's algorithm is constructed from the graph.

Implementation cont'd.

- Recipes are clustered by removing the most expensive edges from the graph.
- Breadth-first search is implemented to traverse the recipes in each clusters.
- Common ingredient in the recipes of a cluster represent the cluster.

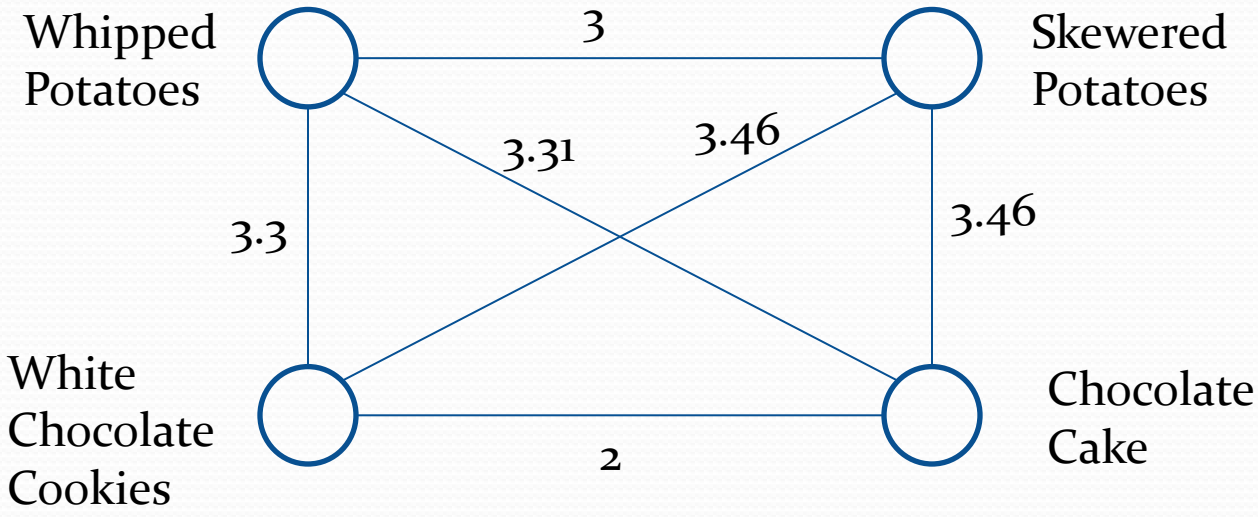
Example

- Consider the following four recipes:
  1. Whipped Potatoes - potatoes, cheese, cream, butter, garlic, pepper, paprika
  2. Skewered Potatoes – potatoes, water, mayonnaise, chicken, rosemary, garlic
  3. White Chocolate Cookies – butter, sugar, egg, cocoa powder, flour, chocolate
  4. Chocolate Cake - chocolate, butter, salt, flour, sugar, vanilla extract

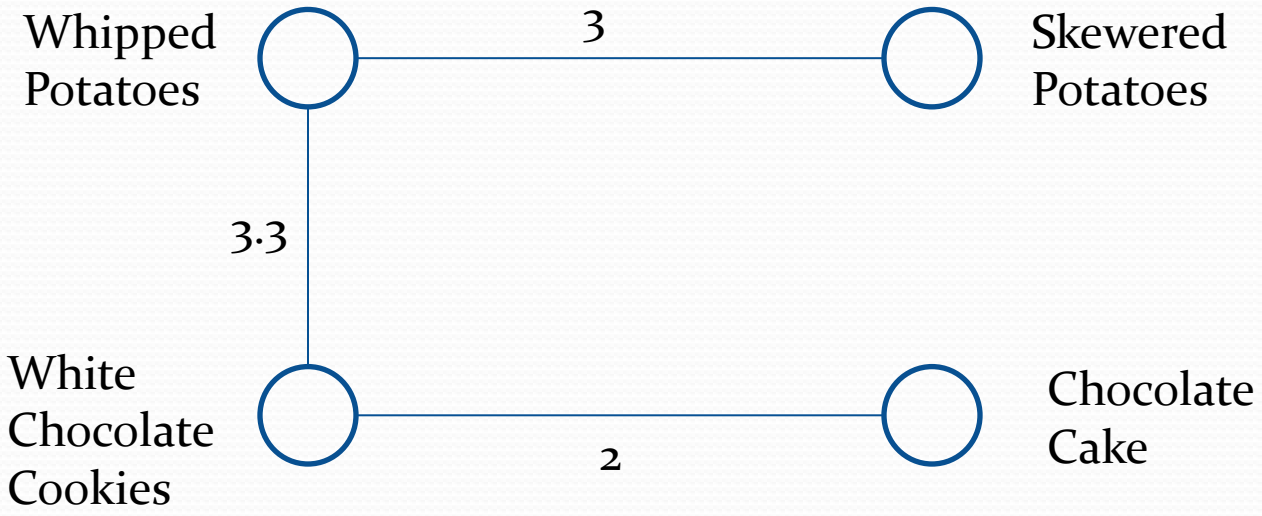
Vector representation

- Ingredients vector for recipes, Whipped Potatoes and Skewered Potatoes will be - [butter, cheese, chicken, cream, garlic, mayonnaise, paprika, pepper, potatoes, rosemary, water].
- The 0-1 ingredient vector for each recipe will be -
  - Whipped Potatoes - 1 1 0 1 1 0 1 1 1 0 0
  - Skewered Potatoes – 0 0 1 0 1 1 0 0 1 1 1
- Euclidean distance will be 3.

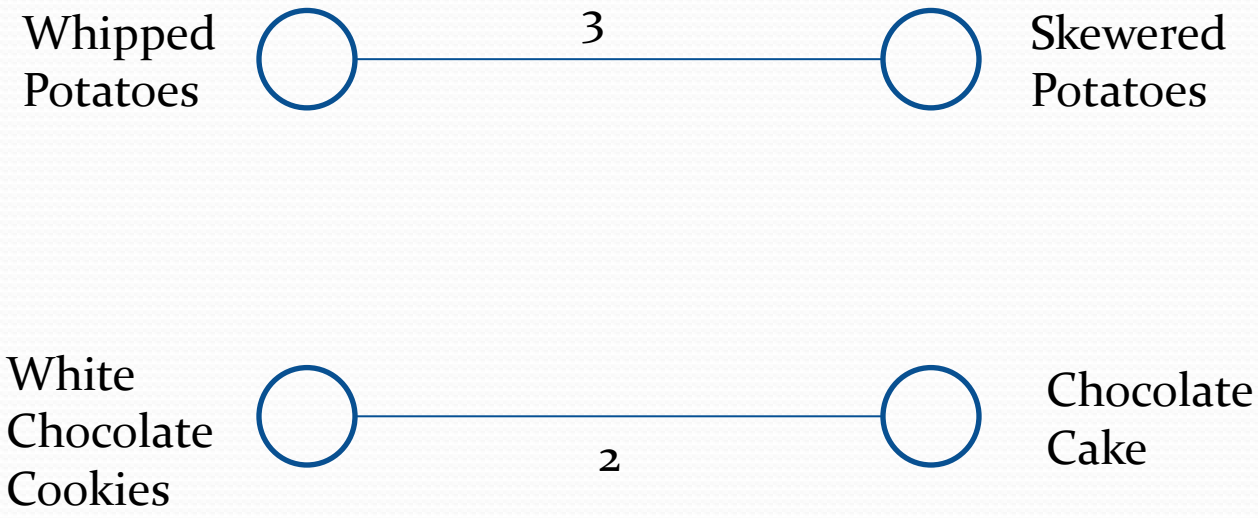
Recipe Graph



Minimum spanning tree



Recipe Clusters



Resultant clusters

- Resultant clusters will be:
  - Cluster 1 = {Whipped Potatoes, Skewered Potatoes}
  - Cluster 2 = {White Chocolate Cookies, Chocolate Cake}
- Ingredient present in most of the recipes in a cluster is considered as the common ingredient.
- Common ingredient represents the cluster. Here, potatoes represents Cluster 1 and Chocolate represents Cluster 2.

Agenda

- Motivation and Project Description
- Clustering and Implementation
- **Integration with Yioop!**
- Tests and Results
- Demo
- Conclusion

Yioop!

- Yioop! is a GPLv3, open source, PHP search engine developed by Dr. Chris Pollett.
- Version used for this project is version 0.66.
- The queue-server is the coordinator for the crawls. It sends URLs to the fetcher.
- The fetcher crawls and downloads the summaries of pages. It also creates a partial index.

Yioop! Features

- The summaries are sent back to the queue-server which merges into the index.
- Text searches can be done as soon as the crawl is stopped.
- Yioop! allows users to add meta words to the documents.
- Meta word is a word which was not in the downloaded document but which is added to the index as if it has been in the document.

Yioop! Features cont'd.

- Meta words added to the documents can be used as keywords to search for specific documents.
- For example, filetype : pdf returns documents found with the extension pdf.
- Meta words 'recipe' and 'ingredient' are added to identify recipes and clusters of recipes for this project.
- Meta word 'ingredient' is used while searching for recipes.

Modifications to Yioop!

- New folder ‘components’ was added to Yioop! which includes implementation of the clustering algorithm.
- Option to select the Recipe Processor was added to the Yioop! crawls option interface.
- Clustering was performed after the crawling is stopped.
- The queue-server was modified to call post processing once the crawling is stopped.

Modifications to Yioop! cont'd.

- Html processor was modified to detect the recipe pages. Detection is done using XPath.
- If a recipe page is detected, only ingredients are extracted from the page.
- The extracted document is marked as 'recipe'.
- The fetcher adds meta word 'recipe:all' to the documents while building the partial index.

Modifications to Yioop! cont'd.

- The meta word is used to extract the recipe pages for clustering in the post processing.
- Ingredients of the recipes are scrubbed to extract the main ingredient.
- Recipe vectors are constructed and the Euclidean distances are calculated.
- Clustering is performed and the common ingredient for the cluster is determined.

Modifications to Yioop! cont'd.

- Meta word 'ingredient:<common ingredient>' is added to each recipe page of each cluster by the fetcher.
- The recipe pages are added back to the index by the queue-server.
- Search can be performed by querying 'ingredient: query term'.
- Recipes with the ingredient typed are displayed along with the other ingredients needed.

Agenda

- Motivation and Project Description
- Clustering and Implementation
- Integration with Yioop!
- **Tests and Results**
- Demo
- Conclusion

Tests

- Important step is clustering is the selection of distance measure.
- Dot product of vectors, Manhattan distance and Euclidean distance were chosen for Testing.
- Testing is done on a sample of 100 recipes.
- Error rate is calculated as –

Error rate = (False Positives/total number of recipes)

Comparison of distance measures

Distance Measures	Error rates
Dot product	0.1
Manhattan distance	0.07
Euclidean distance	0.07

- Manhattan distance and Euclidean distance have less error rate than Dot product.

Performance of distance measures

Distance Measures	Time taken (in sec for 400 recipes)
Dot product	11
Manhattan distance	13
Euclidean distance	13

- Euclidean distance was selected as it has lower error rate than dot product.

Scalability Test

Recipe Size	Number of edges	Edges = n(n-1)/2
10	45	(10*9)/2
30	435	(30*29)/2
50	1225	(50*49)/2
100	4950	(100*99)/2

- The number of edges generated are n(n-1)/2 where n is the number of recipes.

Scalability Test cont'd.

- As the number of recipes increases, the number of edges calculated also increases quadratically.
- The clustering algorithm can be modified to implement clustering incrementally.
- Clustering can be implemented for every n recipes.
- Similar recipes will have the same common ingredient.

Agenda

- Motivation and Project Description
- Clustering and Implementation
- Integration with Yioop!
- Tests and Results
- **Demo**
- Conclusion

Agenda

- Motivation and Project Description
- Clustering and Implementation
- Integration with Yioop!
- Tests and Results
- Demo
- **Conclusion**

Conclusion

- This project provides a dedicated recipe-based search feature within Yioop! Search Engine.
- Clustering algorithm implemented in this project can be applied in other domains.
- This project can be extended to provide ingredient-based search for Yioop! on mobiles.

Thank You