EXPLOITING DOMAIN KNOWLEDGE IN PREDICTIVE LEARNING FROM FOOD AND NUTRITION DATA

Gordana Ispirova

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Prof. Dr. Barbara Koroušić Seljak, Computer Systems Department, Jožef Stefan Institute, Ljubljana, Slovenia
Co-Supervisor: Assist. Prof. Dr. Tome Eftimov, Computer Systems Department, Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:
Prof. Dr. Sašo Džeroski, Chair, Computer Systems Department, Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Zoran Bosnić, Member, Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
Assoc. Prof. Dr. Riste Stojanov, Member, Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University, Skopje, North Macedonia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Gordana Ispirova

EXPLOITING DOMAIN KNOWLEDGE IN PREDICTIVE LEARNING FROM FOOD AND NUTRITION DATA

Doctoral Dissertation

IZKORIŠČANJE DOMENSKEGA ZNANJA PRI NAPOVEDNEM UČENJU IZ PODATKOV O ŽIVILIH IN PREHRANI

Doktorska disertacija

Supervisor: Prof. Dr. Barbara Koroušić Seljak

Co-Supervisor: Assist. Prof. Dr. Tome Eftimov

Ljubljana, Slovenia, November 2022

To my family.

Acknowledgments

Getting a PhD was never a chapter in the life that I imagined I would have, but, as it turns out, you cannot skip chapters, that is not how life works. You have to read every line. Meet every character. You will not enjoy all of it. Some chapters will make you cry. Some will make you smile. Some will bring you to the verge of giving up. Some will show you endless opportunities. In some chapters, you will read things you do not want to read. In some you will not want the pages to end. My PhD was all of these chapters for me, it was a chapter that could not be skipped.

If there is one person who showed me that this is possible, it is my supervisor, Prof. Dr. Barbara Koroušić Seljak. I would like to express my heartfelt gratitude and appreciation to her for never failing me, for allowing this to be my own work, for helping me in every step of the way, for showing me how to be patient and forgiving and truly a good person, as I could not have had a better example of it than her.

I would also like to express my sincere gratitude to the members of the evaluation board: Prof. Dr. Sašo Džeroski, Prof. Dr. Zoran Bosnić, and Assoc. Prof. Dr. Riste Stojanov for their constructive feedback on this work.

Achieving a goal requires people who bring passion and positivism into your life, people who understand what you are going through and share your burdens, people who give you courage and laughter, and people who give you unconditional love. I am fortunate to have them all. First, a thank you from the bottom of my heart goes to my co-supervisor and friend – Assist. Prof. Dr. Tome Eftimov for being the person who can change my mood immediately with his positivism and passion for science, for making all of this happen, for believing in me when I did not, for showing me how to be humble and reach for the stars at the same time, and for teaching me one big lesson: "If you were not ready, you would not have the opportunity, and if you were not capable, you would not have the desire."

Tough times require a reminder that we are not alone. I had incredible girls to share my struggles with and find ways to cope with the stressful times of a PhD journey. This thank you goes to Ana Kostovska, Nataša Malinova and Milka Ljoncheva. Tough times also require encouragement and a reminder to laugh. This thank you goes to my best friends Georgi Kostov and Gjorgi Peev – thanks for always pushing me towards better things.

We are nothing without love and family. Thanks to my four grandparents, my four aunts, and my four uncles for always believing in me and making me feel loved and appreciated all the time. Thanks to my cousins, all nine of them, for their true love and trust, their contagious smiles, and their never-failing ability to cheer me up. My very profound gratitude goes also to my parents, Betka and Panče, and my brother, Igor, for their unconditional love and support in everything I do. I love you infinitely!

Lastly, thanks to my love, Martin, for finding me when I was lost, for enduring with me in the last year of my PhD, for allowing me to see myself with different eyes, and showing me that the best way to predict the future is to create it.

At the end, a reminder that persistence makes all the difference, so the last thank you goes to ME — past ME, present ME and future ME, none of whom none the wiser.

Abstract

Human knowledge about food and nutrition has evolved drastically with time. With food and nutrition-related data being mass produced and easily accessible, the next step is to use Artificial Intelligence (AI) to translate data into knowledge. The majority of AI research is model-driven, and classical Machine Learning (ML) pipelines concentrate on the model-centric approach, prioritizing training the best model for a specific task, with the main focus on improving model parameters, overlooking the importance of data.
We propose a novel ML pipeline that fused data and domain-driven knowledge for a predictive task from the Food and Nutrition domain – fast prediction of nutrient values from unstructured recipe text. Our proposed pipeline consists of three parts: representation learning (RL), unsupervised ML, and supervised ML. In the RL part, word and paragraph embeddings are learned for text short descriptions of foods (recipe titles), in the unsupervised ML part the recipes are separated in clusters based on a domain-specific coding (FoodEx2 classification) from external domain resource, and in the supervised ML part, the two parts are combined – separate predictive models are trained for each cluster for separate nutrients using the learned embeddings as input features. The pipeline is evaluated with a criteria defined using domain knowledge (nutrient tolerance levels) and compared to baselines also calculated using the same criteria.
As the evaluation results showed that including the domain knowledge in the unsupervised ML part improved the results compared to the baseline, we propose an alteration of the ML pipeline. We include two different external sources of domain knowledge for clustering in the unsupervised ML part, to explore the domain bias for the same prediction task.
To further improve the ML pipeline, we include domain knowledge in the RL part of the pipeline. Instead of obtaining recipe title embeddings, we introduce a domain heuristic for merging embeddings of the ingredients of the recipe. This proved to be a successful way to train excellent performing predictive models for predicting nutrient values, as the accuracies obtained were significantly higher than the baseline.
As the domain-specific embeddings showed to be high performant, through the process of data normalization using dictionary and rule-based Named Entity Recognition and data mapping to a Food Composition Database from six heterogeneous multilingual recipe datasets, we composed two predefined corpora of embeddings – ingredient and recipe embeddings. Training embeddings tailored for a specific task is a very time-consuming process, therefore these corpora of predefined embeddings can be used for research purposes as well as transferred to other tasks for application purposes.
To explore the major impact data has on model-performance, we focused on generalization of predictive models, by defining a generalizability index that indicates the trust of transferring a predictive model learned on one dataset to another. Going a step further to show the importance of data in predictive modeling, we show different ways of selecting a representative training dataset, and the results show how different selections of the training dataset produce different outcomes. The training data should be representative of the data expected in deployment, covering all variations that deployment data will present.

Povzetek

Ekspertno znanje o živilih in prehrani se je v zadnjem času drastično povečalo. Umetna inteligenca (UI) omogoča dodatno nadgrajevanje tega znanja s (pol)avtomatsko izluščenim znanjem iz množično zbranih podatkov o živilih in prehrani, ki so relativno enostavno dostopni. Vendar pa raziskave na področju UI pogosto zanemarjajo pomen podatkov in so bolj usmerjene v samo modeliranje. Klasični razvoj cevovodov strojnega učenja (SU) je osredotočen na učenje najboljših možnih modelov za izbrane naloge in optimizacijo parametrov izbranih modelov. V doktorski nalogi predstavljamo novo metodologijo za razvoj cevovoda SU, ki z zlivanjem ekspertnega znanja z znanjem, pridobljenim iz podatkov, omogoča hitro napovedovanje hranilnih vrednosti iz receptov, zapisanih v nestrukturirani tekstovni obliki, kar je zahtevna naloga s področja živilstva in prehrane. Predlagani cevovod temelji na predstavitvenem učenju (PU), nenadzorovanem SU in nadzorovanem SU. PU omogoča začetno učenje vdelav (angl. embeddings) besed in odstavkov, ki opisujejo živila oziroma naslove receptov. Nenadzorovano SU porazdeli vdelave obravnavanih receptov v gruče (angl. clusters) upoštevajoč domensko znanje o klasifikaciji živil in jedi po standardiziranem sistemu FoodEx2. Nadzorovano SU pa je namenjeno učenju modelov napovedovanja za vsako gručo posebej upoštevajoč posamična hranila. Cevovod smo ovrednotili s primerjavo kriterija (t-j. stopnje tolerance hranil), ki temelji na domenskem znanju, z izhodiščnim. Ker se je izkazalo, da upoštevanje domenskega znanja v nenadzorovanem SU izboljša rezultate napovedovanja, smo predlagali nadgradnjo cevovoda SU. Z namenom preučitve morebitne domenske pristranskosti – smo v nenadzorovano SU vključili dva različna zunanja vira domenskega znanja za porazdeljevanje v gruče. Prav tako smo vključili domensko znanje v PU in s tem dodatno povečali učinkovitost cevovoda SU. Tako namesto vdelav naslovov receptov uvajamo domensko hevristiko za združevanje vdelav posameznih sestavin recepta. Izkazalo se je, da je to uspešen način učinkovitega učenja modelov napovedovanja hranilnih vrednosti, saj je bila natančnost značilno višja od izhodiščnih vrednosti. Ker so se v procesu normalizacije in preslikave podatkov domensko pogojene vdelave izkazale kot visoko zmogljive, smo izdelali dva ločena korpusa vnaprej določenih vdelav, enega z vdelavami sestavin in drugega z vdelavami receptov. Normalizacija podatkov je temeljila na prepoznavanju imenskih entitet z uporabo slovarja in pravil, medtem ko so se podatki iz šestih baz z mednarodnimi recepti preslikali na podatke o sestavi živil. Učenje vdelav, prilagojenih za določeno nalogo, je časovno zahteven proces, zato lahko izdelana korpusa z vnaprej določenimi vdelavami uporabimo v raziskovalne namene, možna pa je tudi njihova prevedba na druge aplikativne naloge. Da bi preučili glavni vpliv podatkov na zmogljivosti modela, smo se osredotočili na problem posploševanja modelov napovedovanja. Uvedli smo indeks posplošljivosti, ki ocenjuje stopnjo zaupanja v prenos modela napovedovanja, pridobljenega na eni, za uporabo na drugi podatkovni množici. V naslednjem koraku smo raziskali, kakšen pomen imajo podatki pri modeliranju napovedovanja. Izkazalo se je, da ima izbor množice podatkov za učenje pomemben vpliv na končni rezultat. Predvsem je pomembno, da so podatki, namenjeni učenju, dovolj reprezentativni in zajemajo dovolj variabilnosti, kot je pričakovano v končni uporabi.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

AI	... Artificial Intelligence
ANNs	... Artificial Neural Networks
CBOW	... Continuous Bag-of-Words Model
CS	... Computer Science
DM	... Data Mining
DS	... Data Science
EFSA	... European Food Safety Authority
EHRs	... Electronic Health Records
FCDB	... Food Composition Database
FDA	... Food and Drug Administration
FSA	... Food Standards Agency
IBS	... Irritable Bowel Syndrome
ICT	... Information and Communication Technologies
IE	... Information Extraction
LASSO	... Least Absolute Selection Shrinkage Operator
ML	... Machine Learning
NER	... Named Entity Recognition
NLP	... Natural Language Processing
NNs	... Neural Networks
P-NUT	... Predicting Nutrient content
PAM	... Partition Around Medoids algorithm
PCA	... Principal Component Analysis
PV-DBOW	... Distributed Bag of Words version of Paragraph Vector
PV-DM	... Distributed Memory version of Paragraph Vector
RL	... Representation Learning
SG	... Skip-gram Model
SI	... The International System of Units
USDA	... United States Department of Agriculture
WHO	... World Health Organization

Glossary

Artificial intelligence (AI) – is the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings.
BERT – a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection.
Clustering – is a ML technique, which groups the unlabeled dataset.
Cross validation – is a procedure that combines (averages) measures of fitness in prediction to derive a more accurate estimate of model prediction performance.
Data mining – is the process of extracting and discovering patterns in large data sets involving methods at the intersection of machine learning, statistics, and database systems.
Decision tree regression – observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output.
Doc2Vec – is an NLP tool for representing documents as a vector and is a generalization of the Word2Vec method.
Elastic Net regression – is an extension of linear regression that adds regularization penalties to the loss function during training.
Electronic health record (EHR) – is the systematized collection of patient and population electronically stored health information in a digital format.
EuroFIR AISBL – is a non-profit international association, which supports use of existing food composition data and future resources through cooperation and harmonization of data quality, functionality and global standards.
European Food Safety Authority (EFSA) – is the agency of the European Union that provides independent scientific advice and communicates on existing and emerging risks associated with the food chain.
Food and Drug Administration (FDA) – is a federal agency of the Department of Health and Human Services, responsible for protecting the public health by ensuring the safety, efficacy, and security of human and veterinary drugs, biological products, and medical devices; and by ensuring the safety of our nation's food supply, cosmetics, and products that emit radiation.
Food composition database (FCDB) – are detailed sets of information on the nutritionally important components of foods and provide values for energy and nutrients including protein, carbohydrates, fat, vitamins and minerals and for other important food components such as fibre.
Food Safety Agency traffic light system – is a system for indicating the status of a variable using the red, amber or green of traffic lights.
FoodEx2 classification system – is the measure of a well-balanced ratio of the essential nutrients carbohydrates, fat, protein, minerals, and vitamins in items of food or diet concerning the nutrient requirements of their consumer.
GloVe – is an unsupervised learning algorithm developed by Stanford for generating word embeddings by aggregating global word-word co-occurrence matrix from a corpus.

Hyperparameter tuning – is the problem of choosing a set of optimal hyperparameters for a learning algorithm.
Hyperparameter – is a parameter whose value is used to control the learning process of a ML algorithm.
Information extraction (IE) – is the process of extracting information from unstructured textual sources to enable finding entities as well as classifying and storing them in a database.
International System of Units (SI) – a system of physical units (SI units) based on the metre, kilogram, second, ampere, kelvin, candela, and mole, together with a set of prefixes to indicate multiplication or division by a power of ten.
K-means clustering – is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters.
LASSO regression – is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model.
Linear regression – is a linear approach for modelling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables).
Machine learning (ML) – is defined as a discipline of AI that provides machines the ability to automatically learn from data and past experiences to identify patterns and make predictions with minimal human intervention.
Named entity recognition (NER) – is a sub-task of information extraction (IE) that seeks out and categorizes specified entities in a body or bodies of texts.
Natural language processing (NLP) – is a sub-field of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data.
Neural network regression – predicts an output variable as a function of the inputs of a neural network. The input features (independent variables) are numeric types.
Neural network – are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.
Nutrient values – is the measure of a well-balanced ratio of the essential nutrients carbohydrates, fat, protein, minerals, and vitamins in items of food or diet concerning the nutrient requirements of their consumer.
Nutrition label – is a label required on most packaged food in many countries, showing what nutrients and other ingredients (to limit and get enough of) are in the food. Labels are usually based on official nutritional rating systems.
Partition around medoids (PAM) algorithm – is and algorithm intended to find a sequence of objects called medoids that are centrally located in clusters.
Principal Component Analysis (PCA) – is an unsupervised, non-parametric statistical technique primarily used for dimensionality reduction in machine learning.
Random forest regression – is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging.
Regression – is a statistical technique that relates a dependent variable to one or more independent (explanatory) variables.
Regular expression (regex or regexp) – A regular expression (regex or regexp) is a string of characters that indicates a search pattern in text.
Representation learning (RL) – is a set of techniques that allows a system to automat-

ically discover the representations needed for feature detection or classification from raw data.
Ridge regression – is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated.
Supervised machine learning – is a subcategory of ML that learns from labeled training data to help you predict outcomes for unforeseen data.
United States Department of Agriculture (USDA) – is the federal executive department in the USA responsible for developing and executing federal laws related to farming, forestry, rural economic development, and food.
Unsupervised machine learning – is a subcategory of ML that is used to identify patterns in data sets containing data points that are neither classified nor labeled.
Word2Vec – is a two-layer neural network that processes text by "vectorizing" words.

Chapter 1

Introduction

We live in a time of a global epidemic of obesity and diabetes, as well as, inactivity, all connected to bad dietary habits. Many other non-communicable chronic diseases such as high blood pressure, cardiovascular disease, some cancers [1], bone-health diseases etc. are linked to, again – poor dietary habits [2]. Dietary assessment is essential for patients suffering from many diseases (especially diet- and nutrition-related ones), it is also very much needed for professional athletes, and because of the accessibility of meal tracking mobile applications, it is becoming part of everyday habits of a vast majority of individuals, for health, fitness, or weight loss/gain. Obesity is spiking each day in developed western countries and this contributes to raised public health concern about some subcategories of nutrients, specifically about saturated fats, and added or free sugar. Nutritional epidemiologists are also raising concern about micronutrients like – sodium, whose intake should be monitored for individuals suffering from specific diseases like osteoporosis, stomach cancer, kidney disease, kidney; and dietary fiber, whose intake is critical for patients suffering from irritable bowel syndrome (IBS). This being said, there is no denying that nutrition has become a core factor to today's society, and one of the clear solutions to the global health-crisis [3]–[6]. The path towards making the average human diet healthier and environmentally sustainable is a fundamental part of the solution for numerous challenges from the ecological, environmental, societal and economic perspective, and the awareness for this has just started to grow and be fully appreciated. Human understanding and knowledge about food and nutrition is constantly evolving, and has significantly improved recently, one of the main contributors to this is data. However, the possibilities of gaining knowledge from food and nutrition-related data are yet to be explored.

Predictive modeling of issues connected to human health has received a lot of attention in recent decades, and a substantial amount of research has been published in this direction. The availability of numerous available biomedical vocabularies and standards, which play a critical role in comprehending health information, as well as a huge volume of health data, makes it possible to resolve difficulties linked to healthcare. However, concentrating primarily on healthcare data restricts the potential advantages that artificial intelligence (AI) and its vast capabilities might bring to our lives. As a result, the Lancet Planetary Health journal in 2019 stated that exploring the linkages between food systems, human health, and the environment will be the focus of future advances in our welfare and society. Although – there is extensive data available, the disconnect between collecting data and putting the information contained in it in use is a valid barrier to be overcome, but it is not insurmountable. In domains, such as the biomedical, which is connected to the healthcare industry, vigorous steps have been taken in the direction of putting the data generated and available into use. In the wake of a situation like a global pandemic, there is a spike

in the interest in predictive analysis for improving healthcare. With a strong integration of biomedical and healthcare data, including electronic health records, modern healthcare organizations can possibly revolutionize the medical therapies and personalized medicine. Different biomedical annotated corpora have been produced in recent years [7]–[11], where the main aim is to challenge and encourage research teams on Natural Language Processing (as the data mainly comes in text format) and data analyzing problems.

Despite the huge number of accessible resources and work done in applying AI in the domains of health [12], the resourcefulness and extensive research done in this area for biomedical tasks, the situation in the Food and Nutrition domain is seemingly different – there is a scarcity of resources. Therefore, there are only a few predictive studies that use food and nutrition data for predicting some targets [13], [14]. This is especially vital during the current world situation with COVID-19, when food security, as well as healthy diet and the environment, are critical for speedy recovery and long-term sustainability of our society [15]. However, to address these challenges developing data-driven AI methods is extremely required. All of this is in line with the focus of the European Commission project calls under Horizon Europe funding framework, where different funding programmes highlight the importance of involving the food and nutrition data to be utilized with AI methods, to estimate the world's big challenges in health and make the world a better place to live.

The backbone of Data Science and Data Mining (DM) per se – are machine learning algorithms. Machine Learning is one part from AI, where algorithms have the benefit of being able to model data in a non-linear and non-parametric way. Machine Learning (ML) algorithms have the benefit of being able to model data in a non-linear and non-parametric way. This can provide us with new understandings, insights and open new possibilities in many domains. A classical supervised learning prediction task involves several steps: i) pre-processing data, ii) feature extraction, iii) training a predictive model, and iv) evaluating the performance of the model. When dealing with textual data (NLP task) the second part is learning representations with representation learning algorithms for obtaining text embedding vectors (presented in Figure 1.1). If the data are textual, then the second part is representation learning algorithms for obtaining text embedding vectors. The last goal of having such a model is to be further used to predict the outcomes when new data, which has not been involved in the training process, becomes available. Using ML algorithms to solve a specific application in some domain is a task of Data Mining (DM).

Despite the significant effort put into developing ML/DM pipelines for biomedical studies, only a small percentage of them are put into practice [16], and one of the most difficult challenges is generalizing knowledge learned from one domain dataset – to a new dataset that is described with the same characteristics of the data instances. This means that the ML pipelines that are developed are biased to the quality of the data used for learning the models and applying them to new data requires fine-tuning (i.e., making the models adaptive). In line with this, the Food and Drug Administration (FDA) recently proposed an adaptive AI systems framework [17], where they emphasized that doing ML/DM pipelines in healthcare predictive modeling requires updating the models over time and personalizing them to specific health applications.

Modeling a domain-specific application is a task that requires a deep understanding of the problem in hand, domain expertise, and domain knowledge (represented with semantic resources, and taxonomies created by the domain experts). In this thesis, our goal is to explore how the synergism of domain and data-driven knowledge for food and nu-

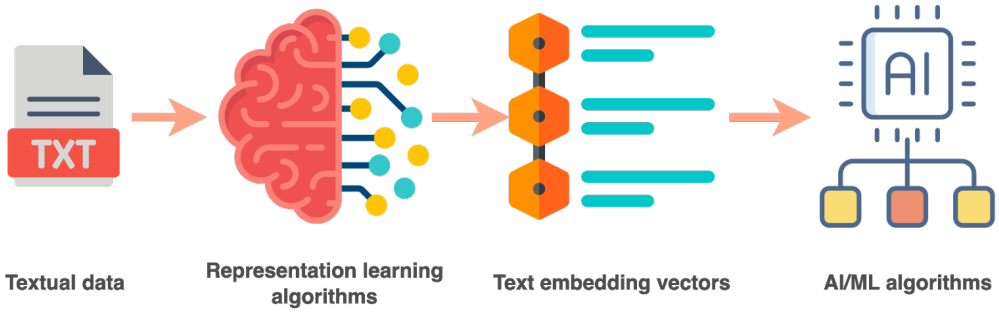

Figure 1.1: Flowchart of a classical ML pipeline.

trition prediction tasks can improve their effectiveness. Moreover, we have investigated if incorporating domain knowledge in different steps (i.e., more specifically through feature extraction, training the model, and model evaluation) of developing an ML/DM pipeline improves the performance of the model. The thesis focuses on exploring this synergism using textual data related to food and nutrition.

1.1 State-of-the-art and Thesis Ambitions

In this section, we provide state-of-the-art in different steps of developing ML/DM pipelines in healthcare predictive modeling.

1.1.1 State-of-the-art in Textual Representations

Nowadays, representation of textual data is a task of representation learning [18]. In general, the idea behind representation learning is to learn a vector of continuous numbers (i.e., embeddings) which represents a text instance. It has been shown that such representations improve state-of-the-art in natural language processing (NLP) comparing them with the classical sparse text representations used in the 90s. The textual embeddings can be trained on different levels (word – e.g., Word2Vec [19], [20], GloVe [21], and sentence/paragraph – Doc2Vec [22], XLNet [23], BERT [24], etc.). In most cases, co-occurrence methods and deep neural architectures are used to calculate them. All these methodologies are further applied on different corpora related to a specific domain of biomedicine from which a vector representation (i.e., an embedding) for each concept is learned and further the embeddings are published to be reused for future predictive modeling studies. For example, by using different biomedical text corpora, biomedical concept representations have been learned using Word2Vec methodology [25]. Further, it has been shown that learning text embeddings on a specific biomedical dataset cannot transfer the knowledge to other predictive studies performed on new data from other domains [26]. For this purpose, cui2vec [27] has been proposed in the biomedical domain which applies GloVe methodology using multi-modal health textual data (i.e., including electronic health records (EHRs), health insurance data, full text biomedical journals, etc.). However, in 2019, one study [28] points out that learning biomedical concepts using graph-based embeddings of a biomedical semantic resource can improve the predictive performance compared to using cui2vec embeddings. The reason reported is that learning representation from the semantic resource is actually learning from domain knowledge, since all concepts and relations in the semantic resource are determined by domain experts [28]. Even more, it has been shown that BERT, which is state of the art in representation learning in NLP, trained on biomedical data leads to BioBERT

and improves the performance in predictive healthcare. In the Food and Nutrition domain there are few smaller predefined corpora of embeddings, in [29], [30], the embeddings are learned from the instructions of the recipes (contextualized embeddings). Other work in this direction is gained as a byproduct of recipe-image retrieval task [31]–[34].

1.1.2 Thesis ambition in textual representations

The thesis will go beyond state-of-the-art in biomedical representations by learning representation for food concepts, with which it will populate a missing part of resources required in predictive healthcare. For this purpose, we are going to explore text representation learning methods to learn food and nutrition concepts embeddings. Even more, since a food concept can be complex (e.g., a recipe consists of several foods, therefore will be composed of several food concepts), its representation will be defined by fusing the separate food concept embeddings with a heuristic that is defined using the domain knowledge. We could not explore graph-based representation since there is a lack of semantic resources in the food domain, and those available are under development [35], [36]. Having such domain representations can further help the predictive studies that involve them.

1.1.3 State-of-the-art in learning predictive models

Learning predictive models involves training a ML algorithm which links an input vector of a data instance characteristics to an end target. More healthcare predictive studies are performed by training only one general model that can be used to predict the outcome for all data instances. The challenge that appears here is that the learned knowledge cannot be transferred on new data, since a lot of bias exists starting from the quality of the data through the modeling decisions that are taken [37]. One way to go beyond such learning is to apply predictive clustering, which differs from a classical clustering approach, in a way that it finds clusters of instances that are similar based on the descriptive and target variables, and further learns a separate predictive model for each cluster [38].

1.1.4 Thesis ambition in learning predictive models

The thesis goes beyond state-of-the-art by developing ML/DM pipelines that can be used for predicting outcomes (e.g. nutrient value prediction) from food and nutrition data. For this purpose, instead of performing predictive clustering, the pipeline first, involves the domain knowledge encoded in available external semantic resources, to find similar clusters of data instances (using the food domain knowledge), and further uses the clusters to learn predictive models using the descriptive features (i.e., the learned representation) and the target variables. In addition, since the domain semantic resource can be developed for different applications (i.e., involving different bias), we test the sensitivity of such modeling using different resources in this step, including data-driven and deterministic rules. Data-driven clustering utilizes a semantic resource with graph-based embeddings, while the deterministic rules are all possible combinations that can appear in the resource. Furthermore, we explore if the same text representation methodology provides the best results for each cluster, or if some representations are more specific to some of the clusters. All in all, the development of the ML/DM pipelines is led by combining the domain knowledge and the data-driven modeling approach across its different steps. The ML/DM pipelines are tested using recipe description data, from which we predict nutrient values. The learning predictive task is a supervised regression.

1.2 Purpose of the Thesis

The purpose of the thesis is to:

P1. Explore the synergism of domain and data-driven knowledge for prediction tasks in the food and nutrition domain.

P2. Explore the domain bias impact in different stages of predictive food data modeling.

P3. Shorten a critical gap in Data Science applications for the Food and Nutrition domain, by estimating the trust of the developed ML/DM pipeline when applied to new data.

1.3 Goals of the Thesis

The goals of this thesis are:

G1. Develop an ML/DM pipeline for learning predictive models that incorporates the domain knowledge encoded in external semantic resources.

G2. Introduce a domain knowledge-based heuristic for fusing multi-word representations for learning food concept representation that improves the prediction results.

G3. Sensitivity analysis of incorporating the domain knowledge bias in different stages of the food predictive modeling.

G4. Develop quantitative indicators to evaluate the effectiveness of the developed ML/DM pipeline on new, unseen data, as well as estimate the trust of applying it.

1.4 Hypothesis

The thesis' hypotheses are related to methods for resolving the problem in hand and exploring the chosen domain. The first hypothesis is related to including domain knowledge when constructing a ML/DM pipeline, the second one is related to the effect of the domain bias over the predictive task and it is split in two parts, depending on the type of domain knowledge incorporated in the DM/ML pipeline – the first one is related to semantic domain resources, and the second one to domain heuristic for merging representation vectors. At the end, the third hypothesis is related to generalizing predictive models:

H1. Including the domain knowledge into different stages of data-driven modeling improves the performance in a prediction task.

H2. Handling the domain knowledge bias can improve the results of a predictive task.

- **H2.a.** Incorporating different semantic information about the domain knowledge in the modeling process has a significant impact on the prediction task and has the potential to enhance prediction outcomes.
- **H2.b.** Fusion of multi-word representations with a domain knowledge-based heuristic provides representations (i.e., embeddings) that improve the results from a prediction task.

H3. Integrating domain knowledge into data-driven modeling generalizes the predictive models and allows generalization of the knowledge over food and nutrition data that come from different sources.

1.5 Scientific Contributions

The research presented in this thesis results in several scientific contributions, the Bibliography section contains an exhaustive list of all publications that are connected to this thesis. Each of the hypotheses presented is related to one/or more scientific contributions (SC) that are relevant in the scientific community.

SC1. A novel ML/DM pipeline for learning predictive models that incorporates the domain knowledge encoded in external semantic resources.
This work has been presented in a peer-reviewed journal article [39].

> Assessing nutritional content is very relevant for patients suffering from various diseases, professional athletes, and for other health reasons it is becoming part of everyday life for many. However, it is a very challenging task as it requires complete and reliable sources. We introduce a machine learning pipeline for predicting nutrient values of foods using learned vector representations from short text descriptions of food products. On a dataset used from health specialists, containing short descriptions of foods and nutrient values: we generate paragraph embeddings, introduce clustering in food groups, using graph-based vector representations – that include food domain knowledge information, and train regression models for each cluster. The predictions are for four nutrients: carbohydrates, fat, protein and water. The results from this study imply that inferring domain knowledge before the predictive modeling in a ML pipeline for the task of predicting nutrient values improves the results compared to the baseline (without the inclusion of domain knowledge).

SC2. Exploring domain knowledge bias in the ML/DM modeling by including domain semantic resources that represent different information about the domain.
This work has been presented in a conference paper [40].

> We explore the effect of domain bias in a predictive study in the food and nutrition domain. Having a ML pipeline for predicting nutrient values with learned vector representations from short text description of recipes, we introduce domain knowledge before the prediction algorithms are applied. On a large corpus of recipe data containing short description and nutrient values (both nutrients and others) we introduce word and paragraph embeddings, learn concept representations for the textual descriptions, introduce domain knowledge for clustering the data, and apply machine learning algorithms for predicting the nutrient content of the recipes. We explore the impact of the domain knowledge by introducing two different criteria of clustering the dataset – using graph embedding of the FoodEx2 codes, and using the traffic light labelling system from the FSA; at the end we compare the two different criteria. Evaluating the ML pipeline with the incorporation of both types of domain knowledge showed higher prediction accuracies compared to the baseline, with a slight favor of the version clustering the recipes with the FSA traffic light system, which is expected since it is based on the nutrient values.

SC3. A representation learning pipeline for learning food concept embeddings by introducing a domain knowledge-based heuristic for fusing multi-word representations for improving the prediction results.
This work has been presented in a peer-reviewed journal article [41].

Using the concept of our proposed ML/DM pipeline we constructed a representation learning pipeline in order to explore how the prediction results change when, instead of using the vector representations of the recipe description, we use the embeddings of the list of ingredients. The nutrient content of one food depends on its ingredients; therefore, the text of the ingredients contains more relevant information. We define a domain-specific heuristic for merging the embeddings of the ingredients, which combines the quantities of each ingredient in order to use them as features in machine learning models for nutrient prediction. The results from the experiments indicate that the prediction results improve when using the domain-specific heuristic. The prediction models for protein prediction were highly effective, with accuracies up to 97.98%. Implementing a domain-specific heuristic for combining multi-word embeddings yields better results than using conventional merging heuristics, with up to 60% more accuracy in some cases.

SC4. Evaluation of the proposed ML/DM pipeline through benchmarking it against models obtained without the incorporation of the domain knowledge in a predictive task.
This work has been presented in a peer-reviewed journal article [39], and conference paper [40].

When evaluating the three aforementioned pipelines, we benchmarked the results obtained from the proposed methodology with and without the incorporation of domain knowledge. When evaluating the ML/DM pipeline we compared the results obtained from the proposed ML/DM pipeline against results obtained using conventional heuristics. In the process of evaluation of the extended ML/DM pipeline with the two different clustering approaches, we compared the results obtained from the proposed extension of the ML/DM pipeline with results obtained without incorporating the clustering according to the external semantic resources.

SC5. Evaluation of the newly proposed representation learning pipeline that incorporates the domain knowledge against conventional textual representations in a predictive task.
This work has been presented in a peer-reviewed journal article [41].

In the evaluation process of the newly proposed representation learning pipeline for comparison purposes, we repeated the same steps with the same experimental setup but using conventional embedding merging heuristics – sum and average, which we consider as baselines. The results show that using the domain heuristic for merging the embeddings yields better results than the baselines.

SC6. Creating a predefined domain-specific embeddings of food concepts and recipes and testing the ML/DM pipeline on heterogeneous recipe datasets.
This work has been presented on two conferences [42]–[44].

Although recipe data are very easy to come by nowadays, it is really hard to find a complete recipe dataset – with a list of ingredients, nutrient values per ingredients, and per recipe, allergens, etc. Recipe datasets are usually collected from social media websites where users post and publish recipes. Usually written with little to no structure, using both standardized and non-standardized units of measurement. We collect six different recipe datasets, all publicly available, all in different formats and some including data in different languages. Bringing all of these datasets to the needed format for applying the ML/DM pipeline and

the RL pipeline – includes data normalization using dictionary-based named entity recognition, rule-based named entity recognition, as well as conversions using external domain-specific resources. After the normalization, the domain-specific embeddings are created using the same embedding space for all recipes – one ingredient dataset is generated. The result from this normalization process are two corpora – one with predefined ingredient embeddings and one with predefined recipe embeddings. The Ml/DM pipeline is then evaluated on all recipe datasets. The results from this use case also confirm that the embeddings merged using the domain heuristic yield better results than the baselines.

SC7. Estimating the generalization of the performance of the ML/DM pipeline across different datasets. In addition, determining quantitative indicators of what will be the trust of using the developed pipelines for new data that will become available in the future. By this, we will bring closer the scientific relevance of the proposed thesis to the industry. This work is presented in a journal article [45].

On our predefined corpus of recipe embeddings from the normalized six recipe datasets we apply a pipeline for testing the generalization of predictive models. We train predictive models on one of the six recipe datasets and test the models on the rest of the datasets. After the predictive modeling, we define and calculate generalizability indexes which form a generalizability matrix. These numbers indicate the trust with which predictive models trained on one dataset can be transferred to the other. The evaluation results prove the validity of these indexes – their correlation with the accuracy of the predictions. We also define a generalized training dataset – consisting of sampling from all six recipe datasets; train predictive models with this generalized dataset, and test them on instances from the six recipe datasets that are not selected and included in the generalized dataset. The results from the evaluation of these predictive models show improvement compared to the results from the predictive models trained on one recipe dataset and tested on the others separately.

1.6 Methodology

To achieve the worked-out goals and test the hypotheses, first we start with developing one general machine learning pipeline for predicting nutrient content from short text descriptions, consisting of three parts:

i. Representation learning – learning concept representations of short text descriptions of recipes.

ii. Unsupervised machine learning – including external domain semantic resource for clustering the instances, independently from the RL process.

iii. Supervised machine learning part – training separate predictive models – single-target regressions (one per each cluster) for the nutrients that we want to predict.

When evaluating regression algorithms, the most common baseline is using mean or median (central tendency measures) of the train part of the dataset for all the predictions. We compare these baseline measures with an evaluation metric defined using domain knowledge – as the main goal is obtaining nutrient values which are expressed in grams, and by international legalizations and regulations they can have defined tolerances, we use the tolerance levels stated in [46], [47] and incorporate them in a new defined accuracy measure

for determination on how accurate the predicted nutrient values are. By developing the ML/DM pipeline we test **H1.** Further, we make an extension of the pipeline in order to address **H2**. The extension will include the following:

1. To test **H2.a.** we incorporate two different types of semantic information about the domain knowledge in the unsupervised ML part of the pipeline.

2. To test **H2.b.** (investigating the domain knowledge impact in the representation learning stage on the predictive performance) we construct a RL pipeline that includes fusion of multi-word representations with a domain knowledge-based heuristic.

The third hypothesis **H3.** is tested using six recipe datasets from different sources, available online. To provide explanations in generalizing the knowledge learned by the model, we determine quantity indicators that provide us the level of trust of applying the learned model to a new dataset that was not used for training.

1.7 Structure of the Thesis

The thesis consists of eights Chapters:

1. In Chapter 1 – Introduction, we provide an overview of all the things that are going to follow. We defined the problem in hand, the domain that is going to be explored in this thesis, and the limitations of the approaches that exist that can be used to resolve the problem, then we defined the research hypotheses to overcome these limitations and the scientific contributions that are an outcome of this work.

2. In Chapter 2, we provide the background and related work needed for understanding the methods that are presented in the thesis. We start with explaining the basic concepts of representation learning, what are word, sentence/paragraph and graph embeddings, the methods used to obtain them, and current state-of-the-art. Then we continue with a section about ML – everything connected to ML that is used in the methods described in this thesis. Next, we present what has been done in the direction we are going in generally – across all domains, and specifically in the Food and Nutrition domain. We also give an overview of existing predefined corpora of embeddings for Food and Nutrition-related data, as well as the importance of generalization of predictive models.

3. In Chapter 3, we present our suggested method for proving hypothesis **H1.** – a ML/DM pipeline consisted of three parts that incorporate domain knowledge.

4. In Chapter 4, we present our approach to proving hypothesis **H2.a.** – how does incorporating domain knowledge from different external semantic resources affect the prediction results.

5. In Chapter 5, we present a methodology for proving hypothesis **H2.b.** – introducing a domain heuristic for merging word/sentence embeddings, and exploring how does that affect the prediction results.

6. In Chapter 6, we present a case study of using the presented ML/DM pipeline on heterogeneous datasets. As an indirect output of this study we present corpora of predefined domain-specific embedding of food concepts and recipes.

7. In Chapter 7, we present a use case of a methodology for proving hypothesis **H3.** – we apply the pipeline previously proposed in [48] on the corpus from Chapter 6. The pipeline was first suggested for landscape analysis of ML datasets, which enables us to choose a wide portfolio of benchmark datasets and eliminate performance assessment bias using a bootstrapping evaluation.

8. In Chapter 8, we conclude the thesis with the a discussion of the importance of the presented methodologies and the results obtained, and the scientific significance of each of the methodologies and the thesis as a whole, as well as future work that can be done in this direction.

Chapter 2

Background

In this Chapter we present the background and consider the results from previous research, necessary for comprehending, following and using the methods presented in this thesis. The chapter starts with a description of Food- and Nutrition – related data, and concepts from the Food and Nutrition domain related to this thesis – the FoodEx2 classification system, and the StandFood method. The chapter continues with the basis of Representation Learning (RL) (Section 2.2 – origin, basic definitions, different methods for generating word and sentence/paragraph and graph-based vector representations. After this section, we follow with a section dedicated to Machine Learning (ML) (Section 2.3) – explaining ML approaches used in the methodologies presented in this thesis. Additionally, in each following chapter after this one, we go into more detail regarding the related work connected to the method in question.

2.1 Food- and Nutrition – related data

Food- and nutrition-related data comes in many different forms and formats. Examples include data about the nutritional composition of food (food composition data), data about actual and recommended food consumption on local, national and even international level (food consumption data), recipe data, and many other kinds of data. Different food description and classification systems exist that provide structure for food- and nutrition-related data. When we talk about recipe data, we usually mean text descriptions of composite foods, which are readily available on user-based recipe websites. Recipe data can contain: recipe description – which is usually just a short textual description of the recipe, recipe instructions – which is a longer text with a description of the cooking method and preparation instructions, and ingredient data – the recipe ingredients with quantities. This data are unstructured textual data that can have several variations for conveying the same information, due to the differences in the way of human text expression. There are recipe databases, which are an essential information resource for providing meal planning and shopping assistance. Often the platforms allow to adjust the amount of ingredients needed to the intended number of portions to be cooked. This type of data are important in the means of predicting macro- and micronutrient information on recipe level, i.e. obtaining important nutritional values from the textual recipe data.

2.1.1 FoodEx2 classification system

FoodEx2 [49] is a thorough method for classifying and describing foods which aims to address the requirement for food descriptions in data collections across several food safety domains. To conduct an informed risk assessment and risk management, it is necessary

to collect precise, comparable data. To the extent practicable, EFSA has implemented a number of technical processes and systems to guarantee that the data it receives is uniform. The EFSA's data providers employ the Standard Sample Description (SSD2) [50] data model to describe food and feed samples and analytical results. For chemical contaminants and residues, microbiological contaminants, zoonotic agents, and information on antimicrobial resistance in food, feed, animals, environmental samples, and materials in contact with food, it specifies the data elements and data structure of samples.

FoodEx2, a standardized method for categorizing and characterizing foods, is a supplement to SSD2, and it is made up of descriptions of numerous different food items that have been combined into food groups and larger food categories in a parent-child hierarchy. A core list of food items or generic food descriptions that represent the bare minimum of detail required for intake or exposure evaluations is at the heart of the system. After its first release in 2011, the system underwent extensive testing in a variety of real-world scenarios, allowing for its appraisal and the identification of potential improvement areas. Following this testing phase, FoodEx2 underwent a review and revision process to better meet the needs of the various users. The terminology was greatly increased, new hierarchies were added, and the relationship between the terms and the most crucial aspects was streamlined, especially in the sections on raw materials and natural resources. A food item on the core list has a parent-child relationship with the food items on the extended list that are related to it. Depending on the requirements of the various food safety domains, the terms from the core and expanded list may be aggregated in various ways. The current version comprises seven hierarchies: a service hierarchy for terminology management, five domain-specific hierarchies, and one general purpose structure accessible to users. Facets are groups of supplementary words that describe characteristics and features of dishes from multiple angles. They are utilized to further elaborate on the information the food list term provides.

2.1.2 StandFood

StandFood [51] is a method that standardizes foods according to FoodEx2 classification system, and it consists of three parts. The first part uses a ML approach to classify foods into four FoodEx2 categories: raw (r) and derivatives (d) for single foods, and simple (s) and aggregated (a) for composite foods (c). To describe foods, the second uses a natural language processing (NLP) approach combined with probability theory. In order to improve the classification result, the third component of the StandFood method integrates the results from the first and second parts by creating post-processing criteria.
The StandFood evaluation results demonstrate that the system produces promising results and may be used to classify and describe food items in accordance with FoodEx2. FoodEx2 codes can be found missing in food composition databases and food consumption data using StandFood, allowing users to compare and combine them.

2.2 Representation Learning

Representation learning is method for learning representations (vectors) of input data by transforming it or extracting features from it, which then makes it easier to perform a task like classification or prediction [18]. There are two different categories of vector representations: *non-distributed* or sparse, which are much older and *distributed* or dense, which have been in use for the past few years. Our focus is on distributed vector representations, more specifically we will focus on three methods of generating word embeddings – Word2Vec,

GloVe, BERT and Doc2Vec embedding algorithms.

2.2.1 Word embeddings

Word representations were first introduced as an idea in 1986 [52]. Since then, word representations have changed language modelling [53]. Following up is work that includes applications to automatic speech recognition and machine translation [54], [55], and a wide range of Natural Language Processing (NLP) tasks [56]–[62]. Word embeddings have been used in combination with ML, improving results from biomedical named entity recognition [63], capturing word analogies [64], extracting latent knowledge from scientific literature and going towards a generalized approach to the process of mining scientific literature [65], etc. Word embeddings are vector space models (VSM) that in a low-dimensional semantic space (much smaller than the vocabulary size) represent words in a form of real-valued vectors. Having distributed representations of words in vector space helps improve the performance of learning algorithms for various NLP tasks.

- Word2Vec was introduced as a word embedding method by Mikolov et al. in 2013 at Google [19], and it is a neural network-based word embedding method. There are two different Word2Vec approaches [20]:
  - Continuous Bag-of-Words Model (CBOW) – This architecture consists of a single hidden layer and an output layer. The algorithm tries to predict the center word based on the surrounding words – which are considered as the context of this word. The inputs of this model are the one-hot encoded context word vectors.
  - Skip-gram Model (SG) – In the SG architecture we have the center word and the algorithm tries to predict the words before and after it, which make up the context of the word. The output from the SG model are C number of V dimensional vectors, where C is the number of context words which we want the model to return and V is the vocabulary size. The SG model is trained to minimize the summed prediction error and gives better vectors with increments of C [19], [20].

  If compared, CBOW is a lot simpler and faster to train but SG performs better with rare words.

- GloVe [21] is another method for generating word embeddings. It is a global log-bilinear regression model for unsupervised learning of word representations that has been shown to outperform other models on word analogy, word similarity, and named entity recognition tasks. It is based on co-occurrence statistics from a given corpus.

- BERT [24] or Bidirectional Encoder Representations was made available in late 2018, and it is a transformer-based ML approach for pre-training natural language processing (NLP) created by Google [66]. Models were created using the BERT method of pretraining language representations, which NLP practitioners can then download and use for free. Using your own data, you may either fine-tune these models on a particular job (classification, entity recognition, question answering, etc.) to provide cutting-edge predictions or use them to extract high quality language features from your text data.
  According to a 2020 literature review, which included over 150 research publications assessing and enhancing the model, BERT has become a ubiquitous baseline in NLP studies in just over a year [67]. There are two models of the original English-language BERT: 12 encoders with 12 bidirectional self-attention heads make up the

BERTBASE, and 24 encoders with 16 bidirectional self-attention heads make up the BERTLARGE. Both models were pre-trained using unlabeled data that was collected from the 800 million word BooksCorpus [68] and the 2,500 million word English Wikipedia.
At its core, BERT is a transformer language model with self-attention heads and a variable number of encoder layers. The architecture resembles the transformer presented by Vaswani and colleagues in 2017 in [69].
BERT gains knowledge of word contextual embeddings as a result of training. BERT can be fine-tuned with fewer resources on smaller datasets after pretraining, which requires expensive computational resources, to maximize its performance on certain tasks [24], [70].

2.2.2 Paragraph embeddings

In 2014 [22], an unsupervised paragraph embedding method, called Doc2Vec, was proposed. Doc2Vec in contrast to Word2Vec generated vector representations of whole documents, regardless of their length. The paragraph vector and word vectors are concatenated in a sliding window and the next word is predicted; the training is done with a gradient decent algorithm. The Doc2Vec algorithm also takes into account the word order and context. The inspiration, of course, comes from the Word2Vec algorithm: the first part, called Distributed Memory version of Paragraph Vector (PV-DM), is an extension of the CBOW model with an additional, unique to the document, feature vector (Paragraph ID) added. The word vectors represent the concept of a word, while the document vector represents the concept of a document. The second algorithm, called Distributed Bag of Words version of Paragraph Vector (PV-DBOW), is similar to the Word2Vec SG model. In PV-DM, the algorithm considers the concatenation of the paragraph vector with the word vectors for the prediction of the next word, whereas in the PV-DBOW, the algorithm ignores the context words in the input, and the words are predicted by random sampling from the paragraph in the output. The authors recommend using a combination of the two models, even though the PV-DM model performs better and usually will achieve state of the art results by itself.

2.2.3 Graph-based representation learning

Besides word and paragraph embedding, there are methods that are used for embedding data represented as graphs, consequently named graph embedding. Usually, embedding methods learn vector embeddings represented in the Euclidean vector space, but as graphs are hierarchical structures, in 2017, the authors in [71] introduced an approach for embedding hierarchical structures into hyperbolic space – Poincaré ball. Poincaré embeddings are vector representations of symbolic data, the semantic similarity between two concepts is the distance between them in the vector space, and their hierarchy is waved by the magnitudes of the vectors. Graph embeddings have improved performance over many of the existing models on tasks such as text classification, distantly supervised entity extraction, and entity classification [72], they also have been used for unsupervised feature extraction from sequences of words [73]. In [74], the authors generate graph embeddings (Poincaré) for the FoodEx2 hierarchy [49]. The only graph embeddings used in this thesis are the Poincaré embeddings of the FoodEx2 hierarchy, and these are used for clustering instances. As we are working with structured and unstructured textual recipe data, we focused on the use of word and paragraph embeddings.

2.3 Machine Learning

Machine learning (ML) is a branch of computer science (CS) and artificial intelligence (AI) which focuses on the usage of data and algorithms for imitating the way that humans learn, gradually improving its accuracy [75]. ML is a crucial component of the rapidly growing discipline of data science. Algorithms are trained to generate classifications or predictions using statistical approaches, revealing crucial insights in data mining tasks. Following that, these insights drive decision-making within applications and enterprises, with the goal of influencing important growth key performance indicators. As big data expands and grows, the demand for data scientists will rise, necessitating their assistance in identifying the most relevant business questions and, as a result, the data needed to answer them. The data are presented as a training set – a group of instances described by the same set of characteristics. And then we have three different types of ML learning depending on the output we are after:

1. Supervised ML – if the training set comprises output labels (classes) that are the expected algorithmic output and are provided by an expert from the data domain;

2. Unsupervised ML – when the required output labels are not included in the training set, and its objective is to discover any hidden patterns in the data;

3. Semi-supervised ML – which combines both supervised and unsupervised ML, involves a small number of labeled examples and a large number of unlabeled examples from which a model must learn and make predictions on new examples.

2.3.1 Supervised machine learning

In its simplest form, supervised learning refers to the process of teaching or training a computer system how to utilize labeled data, which indicates that the right answer has already been assigned to certain data (train data) [76]. The next step, is to provide the machine with an unseen set of data without labels (test data), so that the supervised ML algorithm analyses the training data and produces labels for the test data. Two groups of algorithms are used in supervised learning:

- Classification – when the output variable is a category.

- Regression – when the output variable has a real value.

In this thesis we are going to deal with data that are represented not as categories, but as real values – nutrient values (which are Positive Rational Numbers: $n \in \mathbf{Q}+$). Therefore, we are using regression algorithms.

1. Linear regression – is a ML approach used for supervised learning. Based on the provided independent variable, linear regression accomplishes the task of predicting a dependent variable/s (target/s) [77]–[79]. As a result, this regression approach determines if a dependent variable and the other independent variables are linearly related.

    - Advantages:
        - The implementation is straightforward.
        - Lower level of complexity than alternative methods.
        - May lead to over-fitting, but it can be prevented by utilizing dimensionality reduction techniques, regularization techniques, and cross-validation.

- Limitations:
  - Outliers affect the algorithm a lot.
  - It over-simplifies real-world problems.

2. Least Absolute Selection Shrinkage Operator (LASSO) Regression – is a regularization technique [80], [81]. Lasso regression is a type of linear regression that uses shrinkage, which happens when data values shrink toward the mean. Simple, sparse models are encouraged by the lasso approach (i.e. models with fewer parameters). The approach works by identifying and imposing a restriction on the model characteristics that causes some variables' regression coefficients to decrease until they are equal to zero. The model does not include variables having a regression coefficient of zero. Therefore, lasso regression analysis is essentially a shrinkage and variable selection approach, and it aids in identifying the most crucial predictors.

   - Advantages:
     - Overfitting is avoided.
   - Limitations:
     - Features that are chosen might be quite biased.
     - For $n << p$ (n-number of data points, p-number of features), LASSO selects at most n features. LASSO will select only one feature from a group of correlated features, the selection is arbitrary in nature.

3. Ridge Regression – is a linear regression extension that, during training, adds regularization penalty to the loss function [82]. It is used to eliminate multicollinearity [83] in data models. Hoerl and Kennard proposed the idea for the first time in 1970 [82], [84], [85]. This was the end product of ten years of a ridge analysis study [86]. This method was developed as a potential remedy to the imprecision of least square estimators [87] when linear regression models contain certain multicollinear (highly correlated) independent variables – by creating ridge regression estimator. This gives a more accurate estimate of the ridge parameters since its variance and mean square estimator are frequently lower than the previously computed least square estimators [88], [89].

   - Advantages:
     - Trades variance for bias.
     - Prevents overfitting.
   - Limitations:
     - Increases bias.
     - Model iterpretability is low.
     - Need to select perfect alpha (hyperparameter).

4. ElasticNet Regression – This regression algorithm approach combines regularization with variable selection [90] and regularizes regression models by using the penalties from the lasso and ridge procedures. In order to enhance the regularization of statistical models, the strategy combines the lasso and ridge regression approaches. Where the dimensional data exceeds the number of samples utilized, the elastic net approach is best suited. The main functions of the elastic net technology are groupings and variable selection.

- Advantages:
  - Does not have the problem of selecting more than n predictors when $n << p$, whereas LASSO saturates when $n << p$.
- Limitations:
  - Computationally more expensive than LASSO or Ridge.

5. Decision Tree Regression – is a tool for non-linear regression in ML [91]. The decision tree regression algorithm's primary task is to divide the dataset into more manageable portions – it divides the data set into decision and leaf nodes, creating a decision tree. This model is favorable amongst ML specialists given their intelligibility and simplicity [92], [93], especially when there is not enough change in the dataset. One thing to keep in mind when using decision tree is that a small modification in the data might have a significant impact on the structure of the tree.

   - Advantages:
     - Demand less work for preparing the data for pre-processing.
     - Data normalization is not necessary.
     - The construction of a decision tree is not significantly impacted by missing data values.
     - Not significantly impacted by outliers.
     - New features can be easily added.
     - Can handle both categorical and numerical data.
     - As a non-parametric technique it makes no assumptions about space distribution or the structure of a classifier.
     - It is one of the quickest ways to determine correlations between variables and identify the most important ones.
     - Very easy to explain as it is highly intuitive.
   - Limitations:
     - One of the practical issues with decision tree models is overfitting – it occurs when the learning algorithm keeps creating hypotheses that decrease the error in the training set but increase the error in the test set. However, this problem may be fixed by trimming and imposing restrictions on the model's parameters.
     - Instability – a slight change in the data can result in a big change in the decision tree's structure.
     - Compared to other algorithms, the calculations can occasionally become significantly more complicated.
     - Continuous numerical variables do not lend themselves well to the usage of decision trees.
     - The model training process for decision trees is often very long.
     - Due to its intricacy and lengthier training period, decision tree training is highly costly.

6. Random Forest Regression – is another popular approach for non-linear regression in ML. A random forest employs many decision trees, to predict the outcome as opposed to decision tree regression (single tree) [94], [95]. With the help of this approach, a decision tree is constructed using k randomly chosen data points from the provided

dataset [96]. The value of every new data point is then predicted using a number of decision trees. Because a random forest employs many decision trees it will predict many output values, and the final result for a new data point is calculated from the average of all the predicted values. Of course, there are drawbacks of this algorithm, as it maps a large number of decision trees, it requires more computational power, as well as more input in terms of training.

- Advantages:
  - Since there is a part of the dataset that is not used for training, it can be used for testing.
  - Highly performant and accurate.
  - Feature importance estimate is provided.
  - Automatically handles missing values.
  - Does not require feature scaling.
- Limitations:
  - Black box approach, less interpretability.
  - Overfitting can occur.
  - Greater computing resources are required.
  - Prediction time is high.

7. Neural Network Regression – We are all aware of how effective neural networks are when it comes to predictions. Artificial neural networks (ANNs), or neural networks (NNs) or even just "neural nets", are a group of interconnected artificial neurons that serves as a loose representation of the neurons found in a human brain [97], [98]. In a neural network, each node has a unique activation function that determines the node's output depending on a set of inputs. The final activation function can be altered in order to convert the neural network into a regression model. "Keras" [99] is a popular and ideal Python package for creating neural networks for ML purposes. Neural network regression ensures non-linearity by mapping a neuron's output to a range of values. Using neural network regression, you may select a single parameter or a set of parameters to predict the outcome. The neurons (a neural network's outputs) are closely coupled to one another, and each neuron has a weight. The ones that are well-connected aid in mapping the relationships between dependent and independent factors as well as helping in the prediction of future values.

   - Advantages:
     - Given that a neural network is a mathematical model with approximation functions, the model may be applied to any data that can be converted to a numeric format.
     - Neural networks are effective at modeling nonlinear data with many inputs.
     - It is trustworthy while handling tasks with a variety of characteristics.
     - It functions by breaking the categorization issue down into a tiered network of easier components.
     - Once trained, predictions are made rather quickly.
     - Any number of inputs and layer combinations can be used to train neural networks.
     - More data points are beneficial for neural networks.
   - Limitations:

- Since neural networks are "black boxes", we are unable to determine the exact degree to which each independent variable affects the dependent variables.
- Training on conventional CPUs is time-consuming and computationally costly.
- Training data are extremely important for neural networks, which causes the overfitting and generalization issues.

2.3.2 Unsupervised machine learning

Unsupervised learning is the process of teaching a computer to use unlabeled, unclassified data and enabling the algorithm to operate on the data without supervision [100]. Without any prior data training, the machine's objective in this case is to categorize unsorted data according to similarities, patterns, and differences. In contrast to supervised ML, the computer is limited in its ability to discover the hidden structure in unlabeled data on its own.

Unsupervised learning is divided into two groups:

- Clustering – Identifying the natural groups in the data, such as classifying clients based on their purchase patterns, is a clustering problem.
- Association rules – When you wish to find rules that broadly characterize your data, such as "those who purchase X also tend to buy Y".

In this thesis we are exploring the area of unsupervised ML in order to identify different groups of recipes, therefore we are interested in clustering algorithms. The general approach of a clustering algorithm is identifying similar groups of data in a dataset. There are different types of clustering algorithms:

1. k-means clustering – this approach aims to divide n observations into k clusters ($k << n$), where each observation belongs to the cluster that has the closest mean (also known as the cluster centroid or cluster centers), acting as a prototype for the cluster [101].

2. k-medians – is a variant of k-means clustering where the median is calculated rather than the mean for each cluster to find its centroid [102], [103].

3. k-medoids clustering – is also called Partitioning Around Medoids (PAM) clustering [104]–[106]. The PAM algorithm selects real data points as center i.e. medoids of the cluster, in contrast to the k-means and k-medoids algorithms, which do not need the cluster center to be one of the input data points, and whose goal is to reduce the distance between points classified as being in a cluster and a point designated as the center of that cluster. In this way the k-medoids algorithm allows for enhanced interpretability of the cluster centers.
   Moreover, whereas k-means often requires Euclidean distance for effective solutions, k-medoids may be utilized with any dissimilarity measure. k-medoids are more resistant to noise and outliers than k-means because they minimize a sum of pairwise dissimilarities rather than a total of squared Euclidean distances. The approach is again – dividing the dataset of n items into k clusters ($k << n$) using the traditional clustering partitioning approach known as k-medoids, where the number of clusters is supposed to be known in advance. This implies that the programmer must specify k before the execution of a k-medoids algorithm. Methods like the silhouette approach [107] can be used to judge the "goodness" of the supplied value of k.

ML algorithms cannot be discriminated against based on pros and cons. First, when selecting whether or not we should go with a supervised or an unsupervised approach we should consider whether or not we have labeled data or not, and what we want to accomplish with the modeling. Next, when selecting a kind of supervised or unsupervised approach we should consider the type of data we are working with. At the end when choosing a regression algorithm or a clustering algorithm, we should base the selection on the specific use case that we are trying to solve (No free lunch theorem [108], [109]).

Chapter 3

Machine-Learning Pipeline for Integrating Domain and Data Driven Knowledge

In this chapter we present, a ML/DM pipeline – called P-NUT (Predicting NUTrient content from short text descriptions), for nutrient value prediction from short text description including domain knowledge from external resource – the FoodEx2 classification system. We begin the chapter with the problem definition, the motivation for doing this study, then we present related work as an addition to what is mentioned in Chapter 2. Next, we describe the methodology, the three parts of the ML/DM pipeline, the data used in the experiments, the experimental setup and the results obtained from the experiments, as well as a discussion of the results.

3.1 Problem Definition

Data are a major factor in the ongoing advancement of human knowledge and understanding of food and nutrition, which has recently made tremendous strides. The possibilities of gaining knowledge from food and nutrition-related data are yet to be explored. One of the most important pieces of information about food is nutrient content, which is very relevant for patients suffering from various diseases, professional athletes, and slowly part of everyday life of many for health or fitness goals. Nutrient content from one food to another can vary a lot, even though they have roughly the same type of ingredients. This makes nutrient tracking and calculating very challenging, and predicting nutrient content very complicated.

In this study, we propose an approach, called P-NUT (Predicting NUTrient content from short text descriptions). P-NUT is a ML/DM pipeline for predicting nutrient values of a food item considering learned vector representations of text describing the food item. Food items are generally unbalanced in terms of nutrient content. When there is a broad variety of foods, they can go from one extreme to another for one nutrient content, for example the content of fat can go from "fat-free" foods to "fat-based" foods (ex. different kinds of nut butters), which can be a good base for grouping foods. Therefore, a general model for prediction will not be efficient in nutrient prediction. For this reason, we decided to apply unsupervised ML – clustering as a method to separate foods in order to obtain clusters (groups) of foods with similar characteristics. Subsequently, on these

separate clusters we predict the nutrients with applying supervised ML. This is a case of supervised ML, as we are dealing with labeled data. The instances – data points are the food items and the labels are the nutrients. The learned vector representations i.e. the embeddings of the food items are the inputs, and the individual nutrient values (for each nutrient) are the targets of this supervised ML task. Predicting nutrients is not a task that has been approached in such a manner before, usually nutrient content of food is calculated or estimated from measurements and exact ingredients [110]–[112]. These calculations are pretty demanding, the detailed procedure for calculation of the nutrient content of a multi-ingredient food has a few major steps: selection or development of an appropriate recipe, data collection for the nutrient content of the ingredients, correction of the ingredient nutrient levels for weight of edible portions, adjustment of the content of each ingredient for effects of preparation, summation of ingredient composition, final weight (or volume) adjustment, and determination of the yield and final volumes. This is when all the ingredients and measurements are available. When the data for the ingredients are not available, this procedure gets more complicated [110], [111].

With using just short text descriptions of the food products – either a simple food or complex recipe dish, the results from this study show that this way of combining RL with unsupervised and supervised ML provides results with accuracy as high as 80%, compared to the baseline (mean and median – calculated from the values of a certain nutrient of all the food items in a given cluster) in some cases there are differences in accuracies of up to 50%.

3.2 Related Work

ML and prediction modeling has been heavily applied in the health domain, its most prevalent use is for disease diagnosis [113], like cancer [114]–[120], neurodegenerative diseases [121], [122], and most recently COVID 19 [123]. To the best of our knowledge, predicting nutritional content of foods/recipes using only short text description has never been done before. There has been some work involving ML done in this direction, mainly involving image recognition: employing different deep learning models for accurate food identification and classification from food images [124], dietary assessment through food image analysis [125], and calculating calorie intake from food images [126]–[130]. More recently, tools in the form of mobile applications for food item recognition from image: FoodAI [131], Clarifai [132], and Lose It [133]; as well as for predicting calorie content of food images have emerged: Calorie Mama [134], FoodVisor [135], BiteSnap [136], LogMeal [137]. The apps for food recognition work with databases of larger sizes, while the ones for calorie prediction work with significantly smaller databases (the origin is not stated for neither) and give you multiple options to choose for the identified food alongside portion sizes. All this work in the direction of predicting total calories mainly with neural networks. There are numerous mobile and web applications for tracking nutrient intake [138], [139]. Systems like these are used for achieving dietary goals, allergy management or simply, maintaining a healthy balanced diet. The biggest downside is the fact that they require manual imputation of details about the meal/food.

In Chapter 2, we explained the concept behind RL and the RL methods that can be used for words and paragraphs/documents, as well as graphs. In [140], we explored the idea of applying text-based representation methods in the food domain for the task of finding similar recipes based on cosine similarity between embedding vectors, which was the starting point for the idea for this study. To develop this ML/DM pipeline we used

Word2vec [19], [20] and Glove [21], to generate word embeddings, Doc2Vec [22] to generate paragraph/phrase embeddings, and Poincaré [71] to generate graph embeddings.

As mentioned in Chapter 2, Subsection 2.1.1, FoodEx2 is a standardized system for food classification and description developed by the European Food Safety Authority (EFSA), it has domain knowledge embedded in it and it contains descriptions of a vast set of individual food items combined in food groups and more broad food categories in a hierarchy that exhibits a parent-child relationship. The domain knowledge contained in the FoodEx2 hierarchy in the ML/DM pipeline is transcended through graph embeddings, which are later used in order to group the food items from the FoodEx2 system in clusters. The clustering is done using the Partition Around Medoids algorithm (PAM) [106]. The PAM algorithm works on the following principle: it searches for k representative objects in a given dataset (called medoids) and then assigns each object to the closest medoid in order to create clusters. The aim of the algorithm is minimizing the sum of dissimilarities between the objects in a cluster and the center of the same cluster (medoid). The PAM algorithm is a more robust version of the k-means [101] as it is considered to be less sensitive to outliers [141]. Another method used and mentioned in this study is the silhouette method [107], which is a technique for determining the optimal number of clusters, as well as for interpreting and verifying consistency within data clusters.

3.3 Methodology

In Figure 3.1, a flowchart of the methodology is presented. The methodology consists of three separate parts: RL and unsupervised ML, conducted independently, and then combined in supervised ML. The idea is to:

1. Represent text descriptions in vector space using embedding methods, i.e. semantic embeddings at sentence/paragraph level of short food descriptions;

2. Cluster the foods based on their FoodEx2 codes [49] using graph embeddings [74];

3. Perform post-hoc cluster merging in order to obtain more evenly distributed clusters on a higher level of the FoodEx2 hierarchy;

4. Apply different single-target regression algorithms on each cluster having the embedding vectors as features for predicting separate nutrient values (carbohydrates, fat, protein and water);

5. Evaluate the methodology by comparing the predicted with the actual values of the nutrients.

3.3.1 Tolerance levels for nutrient values

The main goal is obtaining nutrient values which are expressed in grams, and by international legalizations and regulations can have defined tolerances. The European Commission Health and Consumers Directorate General in 2012 published [46], with the aim to provide advised recommendations for calculation of the acceptable differences between quantities of nutrients on the label declarations of food products and the ones established in Regulation EU 1169/2011 [46]. These tolerances for the food product labels are important as it is impossible for foods to contain the exact levels of nutrients that are presented on the labels, as a consequence of the natural variations of foods, as well as the variations occurring during production and the storage process. However, the nutrient content of foods

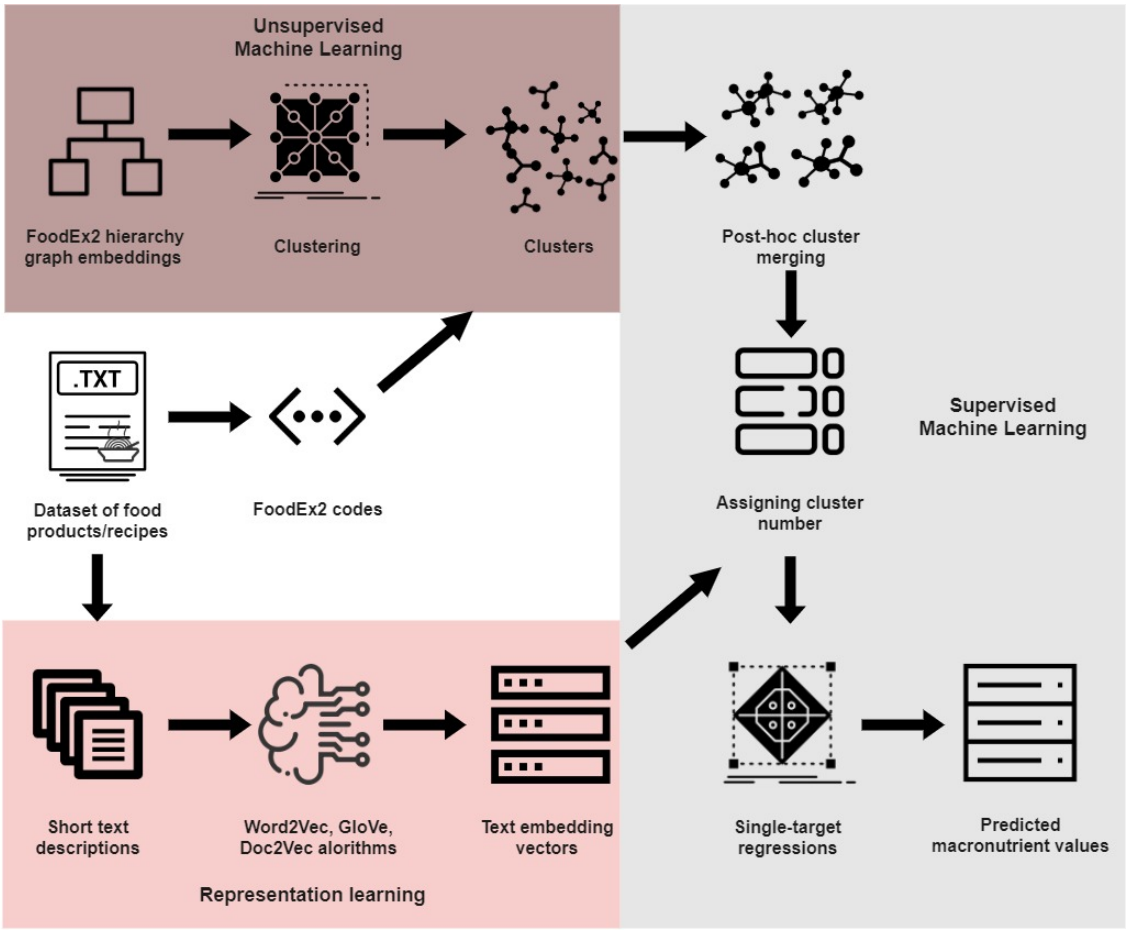

Figure 3.1: Flowchart of the P-NUT methodology.

should not deviate substantially from labelled values to the extent that such deviations could lead to consumers being misled. From the tolerance levels stated in [47], for our particular case we used the tolerance levels for the nutrition declaration of foods that do not include food supplements, out of which we used the needed information presented in Table 3.1 – where the allowed deviations are presented for each of the four nutrients, depending on their quantity in 100 grams of the food in question. These tolerance levels are included at the very final step in our methodology in the determination on how accurate the predicted nutrient values are. With this we are embedding domain knowledge in the evaluation process.

Table 3.1: Tolerated differences in nutrition content in foods besides food supplements.

Quantity of nutrient per 100 g of food	**Tolerances (allowed deviations in quantity)**			
	Carbohydrates	**Protein**	**Water**	**Fat**
$< 10g$ per $100g$	$\pm 2g$	$\pm 2g$	$\pm 2g$	$\pm 1,5g$
$10 - 40g$ per $100g$	$\pm 20\%$	$\pm 20\%$	$\pm 20\%$	$\pm 20\%$
$> 40g$ per $100g$	$\pm 8g$	$\pm 8g$	$\pm 8g$	$\pm 8g$

3.3.2 Representation learning

The starting point is the textual data, in our case the short text descriptions of the food products/recipes, alongside with their FoodEx2 codes and nutrient values. For representing the textual data as vectors, the embeddings are generated for the whole food product name/description, using two different approaches:

1. Learning word vector representations (word embeddings) with the Word2Vec and GloVe methods – The vector representations of the whole description are obtained with merging separate word embeddings generated for each separate word in the sentence (food product name/description). If D as a food product description consisted of n words:

$$D = word_1, word_2, ..., word_n \tag{3.1}$$

And $E[word]$ is the vector representation (embedding) of a separate word:

$$E[word_a] = [x_{a1}, x_{a2}, ..., x_{ad}] \tag{3.2}$$

where $a \in \{1, ..., n\}$, n being the number of words in the description, and d is the dimension of the word vectors, which is defined manually for both Word2Vec and GloVe. These vectors are representations of words, to obtain the vector representations for the food product description we apply two different heuristics for merging the separate word vectors. Our two heuristics of choice are:

- Average – The vector representation for the food product description is calculated as an average from the vectors of the words from which it consists of:

$$E_{average}[D] = [\frac{x_{11} + ... + xn1}{n}, \frac{x_{12} + ... + x_{n2}}{n}, ..., \frac{x_{ad} + ... + x_{nd}}{n}] \tag{3.3}$$

- Sum – The vector representation for each food product/recipe description is calculated by summing the vector representations of the words it consists of:

$$E_{sum}[D] = [x_{11} + ... + x_{n1}, x_{12} + ... + x_{n2}, ..., x_{ad} + ... + x_{nd}] \tag{3.4}$$

Where $E_{average}[D]$ and $E_{sum}[D]$ are the merged embeddings, i.e. embeddings for the whole description. When generating the Word2Vec and GloVe embeddings, we considered different values for the parameters for the embedding algorithms. Such parameters are: dimension size of the vectors, sliding window, architecture of the algorithm (for Word2Vec). The values for the dimension of the embedding vectors of choice are $[50, 100, 200]$, as they are among the frequently used dimensions for textual embeddings for these algorithms, and are efficient in means of execution time and computational power. For these dimensions we assign different values to the parameter called sliding window. This parameter indicates the distance within a sentence between the current word and the word being predicted, and the values of choice for this parameter are $[2, 3, 5, 10]$, because our food product descriptions are not very long – the average number of words in a food product description in the dataset is 11, while the maximum number of words is 30. For the Word2Vec embeddings we considered the two types of feature extraction available i.e. the two architectures of Word2Vec: CBOW and SG. By combining these values for all the parameters, 24 Word2Vec models were trained, plus considering the heuristics for combining – sum and average, a total of 48 models, while with GloVe, a total of 24 models were trained.

2. Learning paragraph vector representations with Doc2Vec algorithm – The Doc2Vec algorithm is used to generate vector representations for each description (sentence). If D is the description of the food product/description, then $E_{Doc2Vec}$ is the sentence vector representation generated with Doc2Vec as follows:

$$E_{Doc2Vec}[D] = [x_1, x_2, ..., x_d] \tag{3.5}$$

Where d is the predefined dimension of the vectors. Same as the two chosen word embedding methods, we considered different values for the parameters of the Doc2Vec algorithm: dimension of the embedding vectors, sliding window, architecture of the algorithm. Again, for the sliding window the values: $[2, 3, 5, 10]$ are chosen and for the dimension of the embedding vectors – $[50, 100, 200]$. As for the architecture both of the architectures in the Doc2Vec model are taken into account – PV-DM and PV-DBOW, and we used the non-concatenative mode (separate models for the sum option, and separate for the average option) because if we used the concatenation of context vectors rather than sum/average, the result would be a much-larger model. Taking into account all these parameters there are 48 Doc2Vec models trained in total.

The pseudo code for the algorithm of this part of the methodology is presented in Algorithm 3.1.

3.3.3 Unsupervised machine learning

Foods exhibit large variations in the nutrient content, therefore have a very unbalanced nutrient content. The dataset in our experiments includes a broad variety of foods, which implies that the content of a nutrient can go from one extreme to another. Therefore, it goes without saying that in order to have better predictions for the content of nutrients, food items should be grouped by some similarity. Here, the FoodEx2 codes that are available come into use, since they already contain domain knowledge, and based on them food items are grouped in food groups and broader food categories in the FoodEx2 hierarchy [49]. Independently of the RL process, we used the method presented in [74], where the FoodEx2 hierarchy is presented as Poincaré graph embeddings [71] and then the FoodEx2 codes based on these embeddings are clustered into 230 clusters. This clustering process is performed on the bottom end of the hierarchy, i.e. on the leaves of the graph. Given that our dataset is rather small compared to the total number of FoodEx2 codes in the hierarchy, and the fact that when assigned a cluster number, some of the clusters in our dataset will contain very few or no elements at all, we decided to do a post-hoc cluster merging. The post-hoc cluster merging is performed following a bottom-up approach, the clusters are merged based on their top-level parents, going level deeper until we have as evenly distributed clusters as possible. The pseudo code for the algorithm of this part of the methodology is presented in Algorithm 3.2.

3.3.4 Supervised machine learning

The last part of the methodology is the supervised ML part, which on input receives the outputs from the RL part and the unsupervised ML part. This part consists of applying single-target regression algorithms in order to predict the separate nutrient values. Single-target prediction models are trained for each nutrient, because from the conducted correlation test (Pearson's correlation coefficient) we concluded that there is no correlation between the target variables. In a real-time scenario, it is somewhat hard to select the right ML algorithm for the purpose. The overall most accepted approach is to select

Algorithm 3.1: RL part of the P-NUT methodology.

Data: Dataset of n recipes represented by a textual description – title
Result: 120 datasets of embedding vectors for the n recipe descriptions

for *each recipe description* r_i **do**
  Tokenize the text;
  Remove punctuation signs and numbers;
  Lemmatize the text;
  for *each token* $word_a$ *of* r_i **do**
    Generate $Word2vec$ embedding of $word_a$
     $E_{word2vec}[word_a] = [x_{a1}, x_{a2}, ..., x_{ad}]$;
    Generate $GloVe$ embedding of $word_a$ $E_{GloVe}[word_a] = [x_{a1}, x_{a2}, ..., x_{ad}]$;
  end
  for *each dimension of the embedding vectors* $d \in [50, 100, 200]$ **do**
    for *each value of the sliding window parameter* $s \in [2, 3, 5, 10]$ **do**
      for *each metric for merging the word embeddings* $m \in [sum, avg]$ **do**
        for *each Word2Vec architecture* $a \in [SG, CBOW]$ **do**
          Generate $Word2vec$ embedding of r_i:
          if $m == sum$ **then**
            $E_{Word2Vec_{a-m-s-d}}[r_i] = [\frac{x_{11}+...+xn1}{n}, ..., \frac{x_{ad}+...+x_{nd}}{n}]$;
          else
            $E_{Word2Vec_{a-m-s-d}}[r_i] = [x_{11} + ... + x_{n1}, ..., x_{ad} + ... + x_{nd}]$;
          end
        end
        Generate $GloVe$ embedding of r_i:
        if $m == sum$ **then**
          $E_{GloVe_{m-s-d}}[r_i] = [\frac{x_{11}+...+xn1}{n}, ..., \frac{x_{ad}+...+x_{nd}}{n}]$;
        else
          $E_{GloVe_{m-s-d}}[r_i] = [x_{11} + ... + x_{n1}, ..., x_{ad} + ... + x_{nd}]$;
        end
      end
      for *each Doc2Vec architecture*
       $a \in [PV - DM_{sum}, PV - DM_{avg}, PV - DBOW_{sum}, PV - DBOW_{avg}]$
       do
        $E_{Doc2Vec_{a-m-s-d}}[r_i] = [x_1, ..., x_d]$
      end
    end
  end
end

Algorithm 3.2: Unsupervised learning part of the P-NUT methodology.

Data: Dataset of n recipes represented with FoodEx2 codes and the full FoodEx2 hierarchy clustered into 230 clusters according to their Poincaré graph embeddings

Result: Dataset of n recipes grouped in k clusters

for *each cluster* k **do**
  Identify the number of recipes belonging to the cluster k by matching the FoodEx2 codes;
end
while *The number of instances in the cluster is imbalanced* **do**
  Merge k_i and k_j if they have the same top-level parent in the FoodEx2 hierarchy;
end

few algorithms, select ranges for the hyper-parameters for each algorithm, perform hyper-parameter tuning, and evaluate the estimators' performances with cross-validation by the same data in each iteration, benchmark the algorithms and select the best one(s). When working with regression algorithms, the most common baseline is using mean or median (central tendency measures) of the train part of the dataset for all the predictions. The pseudo code for the algorithm of this part of the methodology is presented in Algorithm 3.3.

Algorithm 3.3: Supervised learning part of the P-NUT methodology.

Data: 120 datasets of embedding vectors for the n recipe descriptions and dataset of n recipes grouped in 9 clusters

Result: Nutrient values for *carbohydrates*, fat, *protein*, and *water* for the n recipes

for *each cluster* k **do**
  for *each embedding dataset* E **do**
    for *each regression algorithm* $reg \in [Linear, Ridge, Lasso, ElasticNet]$ **do**
      Perform hyper-parameter tuning with GridSearchCV;
      Apply k-fold cross-validation;
      Obtain predicted nutrient values as outputs from the regressions;
      Apply tolerance levels and calculate accuracy;
    end
  end
end

3.4 Results and Discussion

3.4.1 Data

In our experiments we used a dataset that contains nutritional information about food items recently collected as food consumption data in Slovenia [142] with the collaboration of subject-matter experts for the aims of the EFSA EU Menu project [143] – designed for more accurate exposure assessments and to ultimately support risk managers in their decision-making on food safety. The ultimate goal being – enabling quick assessment of

Table 3.2: Subset from the dataset used in the experiments.

Food Name		FoodEx2 Code	Nutrient value in grams			
Slovenian	**English**		**Water**	**Fat**	**Carb**	**Protein**
Zelenjavna rižota s parboiled rižem, sezonsko zelenjavo in repičnim oljem	Vegetable risotto with parboiled rice, seasonal vegetables and rapeseed oil	A041G#F04. A036V	79.36	1.55	16.77	1.79
Medenjaki iz pirine in ržene moke ter hojevega medu	Gingerbread biscuit made of spelt and rye flour and honey	A00CT#F04. A004H$F04. A003J$F04. A033K	0.00	1.81	91.41	8.96
Čokoladna rezina Kit Kat	Candies, KIT KAT Wafer Bar	A009Z	1.63	25.99	64.59	6.51
Zeleni ledeni čaj z medom, arizona	Tea, ready-to-drink, green iced tea, Arizona	A03LD	93.00	0.00	6.80	0.01

exposure to chronic and acute substances possibly found in the food chain [49]. The dataset consists of 3,265 food items, some of which are simple food products and others are recipes with short descriptions. For each food item we have available:

- Food name in Slovene;
- Food name in English;
- FoodEx2 code assigned by a domain expert;
- Nutrient values for: carbohydrates, fat, protein and water.

In this dataset for each instance (food/recipe) besides the energy content the values for carbohydrates, protein, fat and water are provided. Although water content is usually not included in food labels it is considered as the fourth macronutrient, and it is included in this dataset, whereas sugar, sodium and fiber content are not included. It was decided to include water content, as it is an instant indicator of whether the food is dry or not (example: cranberries vs. dried cranberries). This dataset consists of proven nutrient values, as it was a part of a clinical trial. We repeated our experiments for both English and Slovene names of the food products and the recipes. In Table 3.2 a few instances from the datasets are presented in as an example.

3.4.2 Data pre-processing

The first step towards the evaluation is pre-processing of the data. Our dataset for evaluation is a subset from the original dataset, obtained by extracting the English food product descriptions, alongside the columns with the nutrient values (carbohydrates, fat, protein and water). The text descriptions are tokenized. The punctuation signs and numbers that represent quantities are removed, whereas the percentage values (of fat, of sugar, of cocoa, etc.) which contain valuable information concerning the nutrient content, and stop words which add meaning to the description, are kept. The next step is word lemmatization [144],

separate lemmatizers are used for the English names and the Slovene names. In Table 3.3, a few examples of the pre-processed data for the English names are presented.

Table 3.3: Examples of pre-processed English descriptions.

Original description	**Pre-processed description**
Potatoes, mashed, dehydrated, prepared from flakes without milk, whole milk and butter added	["potato", "mashed", "dehydrated", "prepared", "from", "flake", "without", "milk", "whole", "milk", "and", "butter", "added"]
Milk chocolate with 30% cocoa, Gorenjka (250g)	["milk", "chocolate", "with", "30", "cocoa", "gorenjka"]

3.4.3 Experimental setup

After obtaining the data in the desired format, the next step is to apply the algorithms for generating embeddings. For this purpose we used the Gensim [145] library in Python, and the corresponding packages for the Word2Vec and Doc2Vec algorithms. The embedding vectors represent our base for the next steps. Independently of this process, the data are clustered, i.e. the instances are divided in clusters based on their FoodEx2 codes. The clustering presented in [74] clusters the FoodEx2 hierarchy in 230 clusters with the PAM algorithm. To cluster our dataset, using the FoodEx2 codes of the instances, we check to see to which cluster the instance belongs and we assign a cluster number to each instance. Due to the fact that our dataset is substantially smaller, we can observe that some clusters from the clustering of the whole FoodEx2 hierarchy do not contain any instances from our dataset, and other contain very few. As a result, post-hoc cluster merging is carried out, in which the clusters are combined using a bottom-up method. We chose the parents at the third level of the FoodEx2 hierarchy for our dataset, meaning we merged the FoodEx2 codes going up until we reached the third level of the FoodEx2 hierarchy. The final result is 9 clusters. In Table 3.4, a few examples from each cluster are given (the English names are given for convenience purposes).

The next step in our methodology is the ML part – applying single-target regressions according to the following setup:

1. Select regression algorithms – Linear regression, Ridge regression, Lasso regression, and ElasticNet regression (using the Scikit-learn library in Python [146]).

2. Select parameter ranges for each algorithm and perform hyper-parameter tuning – Ranges and values are a priori given for all the parameters for all the regression

Table 3.4: Example instances from each cluster.

Cluster number	**Example food products**		
Cluster 1	Oil, industrial, mid-oleic, sunflower, principal uses frying and salad dressings	Homemade minced lard, Mesarija Kragelj	Margarine (with added vegetable sterols 0,75g/10g), line Becel pro-activ, Unilever
Cluster 2	Peanuts, all types, oil-roasted, with salt	Seeds, pumpkin and squash seed kernels, dried	Avocados, raw, California
Cluster 3	Cheese, processed, 60% fat in dry matter	Yogurt, fruit (peach, cereals), low fat 2.6% milkfat	Baby food, cottage cheese, creamed, fruit (strawberry, banana), FruchtZwerge, Danone
Cluster 4	Plums, canned, purple, light syrup pack, solids and liquids	Segedin cabbage with pork meat	Buckwheat porridge sauted with onion and garlic
Cluster 5	Fried chicken filet (canola oil, without breadcrumbs)	Trout with parsley and garlic sauce	Beef, rib, whole (ribs 6–12), separable lean and fat, trimmed to 1/8 of an inch of fat, all grades, cooked, roasted
Cluster 6	Fruit tea infusion, with sugar and lemon	Soup made of turnip cabbage, peas and tomato (olive oil, stock)	Chicken stew with seasonal vegetables, without roux
Cluster 7	Fish, salmon, pink, canned, without salt, solids with bone and liquid	Salty anchovies in vegetable oil	Tuna with beans, canned
Cluster 8	Ham, sliced, regular (approximately 11% fat)	Chicken hot dog, pan-fried	Turkey ham, sliced, extra lean, prepackaged or deli-sliced
Cluster 9	Egg, whole, cooked, scrambled	Fried egg (olive oil)	Egg spread

algorithms. From all the combinations the best parameters for the model training are then selected with GridSearchCV (using the Scikit-learn library in Python [146]). This is done for each cluster separately.

3. Apply k-fold cross-validation to estimate the prediction error – We train models for each cluster using each of the selected regression algorithms. The models are trained with the previously selected best parameters for each cluster and then evaluated with cross-validation. We chose the matched sample approach for comparison of the regressors, i.e. using the same data in each iteration.

4. Apply tolerance levels and calculate accuracy – The accuracy is calculated according to the tolerance levels in Table 3.1. If a_i is the actual value of the i^{th} instance from the test set on a certain iteration of the k-fold cross-validation, and p_i is the predicted value of the same, i^{th}, instance of the test set, then:

$$d_i = |a_i - p_i| \tag{3.6}$$

d_i is the absolute difference between the two set values. We define a binary variable that is assigned a positive value if the predicted value is in the tolerance level.

$$\begin{aligned} & & & allowed = 1 \;\; if: \\ a_i \leq 10 \wedge & \begin{cases} d_i \leq 2, & for\ protein\ and\ carbohydrate \\ d_i \leq 1.5, & for\ fat \end{cases} \\ & a_i > 10 \;\wedge\; a_i \leq 40 \;\wedge\; d_i \leq 0.2 \times a_i \\ & a_i > 40 \;\wedge\; d_i \leq 8 \end{aligned} \tag{3.7}$$

At the end we calculate the accuracy as the ratio of predicted values that were in the "allowed" range, i.e. tolerance level:

$$Accuracy = \frac{\sum_{i=1}^{n} allowed}{n} \tag{3.8}$$

Where n is the number of instances in the test set. The accuracy percentage is calculated for the baseline mean and baseline median as well – the percentage of baseline values (means and medians from each cluster) that falls in the tolerance level range, calculated according to Equations (3.6), (3.7), and (3.8), where a_i is the actual value of the i^{th} instance from the test set on a certain iteration of the k-fold cross-validation, and instead of p_i we have:

$$b = \begin{cases} \frac{\sum_{i=1}^{m} x_i}{m}, \textit{the baseline is the mean} \\ \mathbf{X}_{[\frac{(m+1}{2}]}, \textit{the baseline is the median and } m \textit{ is odd} \\ \frac{\mathbf{X}_{[\frac{m}{2}]} + \mathbf{X}_{[\frac{m}{2}+1]}}{2}, \textit{the baseline is the median and } m \textit{ is even} \end{cases} \tag{3.9}$$

Where m is the number of instances in the train set, and $\mathbf{X}$ is the train set sorted in ascending order.

The accuracy percentages are calculated for each fold in each cluster, and at the end for each cluster we calculate an average of the percentages from each fold.

3.4.4 Evaluation outcomes

In Table 3.5, the results obtained from the experiments with the embeddings generated from the English names are presented, and in Table 3.6, with the embeddings generated from the Slovene names. In these tables we give the accuracy percentages from the predictions for each target nutrient in each cluster. From these tables we can see that having the Word2Vec and Doc2Vec embeddings as features for the regressions yielded better results in more cases than having the GloVe embedding vectors as inputs to the regressions, but this difference is not big enough to say that these two embedding algorithms outperformed GloVe. In Figures 3.2, 3.3, 3.4 and 3.5, the results for each target nutrient are presented graphically.
In the figures, for each target nutrient, for each cluster, we give the best result obtained with the embedding vectors from the English and Slovene names and compare them with the baseline mean and median for the particular cluster. The embedding algorithm that yields the best results alongside with the parameters and heuristic in the figures is given as:

$$E\_h\_d\_w = \begin{cases} h \in [sum, average], \text{is the chosen heuristic} \\ d \in [50, 100, 200], \text{is the dimension} \\ w \in [2, 3, 5, 10], \text{is the sliding window} \end{cases} \tag{3.10}$$

Where E is the embedding algorithm (Word2Vec, GloVe or Doc2Vec). We can see that the embedding algorithm that yields the best results changes, but in all cases the embedding algorithm gives better results than the baseline methods. In Table 3.7, we present the embedding algorithms (with all the parameters used) that gave the best results for each target nutrient in each cluster, alongside with the regression algorithm used for making the predictions.

From the obtained results we can observe that the highest percentage of correctly predicted nutrient values is obtained in Cluster 9, for the prediction of carbohydrates: 86,36%, 81,82% and 72,73% for the English names and 86,36%, 72,73% and 86,36% for the Slovene names, and for the Word2Vec, GloVe and Doc2Vec algorithms appropriately, whereas the baseline (both mean and median) is more than half less. Following these results are the predictions for protein quantity in the same cluster, and then the predictions for protein and carbohydrates in Cluster 7. When inspecting these two clusters, we concluded that these were the only two clusters that were not merged with other ones, therefore, the FoodEx2 hierarchy is on a deeper level, and the foods inside these clusters are more similar to each other compared to food in other clusters. Cluster 9 consists of types of egg products, and simple egg dishes – each of these foods has almost identical nutrients because they only contain one ingredient – eggs. Cluster 7, on the other hand, contains fish products, either frozen or canned. If we do not consider the results from these two clusters, then the best results are obtained for protein predictions in Cluster 4 (70%-72%) and fat predictions (66%-68%), but compared to the baseline median of that cluster, they are not much better, but if we look at the results from the protein predictions in Cluster 8 (60%-67%), we can see that the obtained accuracies are much higher than the baseline mean and median for this cluster. Cluster 8 mainly contains types of processed meats, which can vary notably in fat content, but have similarities in the range of protein content. For comparison reasons, we also ran the single-target regressions without clustering the dataset. The results are presented in Figure 3.6. From this graphical representation we can conclude the same – the embedding algorithms give better results than the baseline mean and median (in this case of the whole dataset), for each target nutrient. The best results, again, are obtained for the prediction of protein content (62%-64%). In Table 3.8, we give the parameters for the embedding algorithms and the regressors with which the

Table 3.5: Accuracy percentages after k-fold cross validation on each cluster obtained with the embeddings for the **English names** of the food products.
Target: C – Carbohydrates, F – Fat, P – Protein, W – Water.
The numbers shown in bold in the table represent the overall best performance for each nutrient in the given cluster.

Cluster	Target	Accuracy				
		Word2Vec	**GloVe**	**Doc2Vec**	**Mean**	**Median**
1	C	**59.21**	47.84	50.11	1.00	17.47
	F	44.26	35.95	**49.32**	5.05	10.21
	P	56.37	**60.32**	56.95	13.16	14.26
	W	40.32	**52.32**	48.21	8.05	9.26
2	C	**34.84**	34.32	33.22	10.95	13.27
	F	**67.22**	64.69	64.69	7.93	60.55
	P	**63.87**	61.34	59.22	7.58	31.89
	W	50.44	**52.83**	52.41	17.89	19.73
3	C	46.51	46.18	**46.98**	11.13	15.74
	F	**67.42**	63.62	64.00	6.84	59.81
	P	69.64	65.47	**70.74**	8.75	58.55
	W	56.68	**60.85**	58.70	12.18	29.83
4	C	40.92	**43.32**	40.53	12.95	16.40
	F	**68.28**	66.40	66.67	4.79	62.43
	P	**72.50**	70.85	71.71	7.23	66.07
	W	59.09	**61.51**	60.99	11.24	33.86
5	C	**46.38**	37.65	46.07	9.58	15.80
	F	**66.12**	62.38	62.38	4.57	42.43
	P	66.12	63.63	**66.83**	8.73	52.38
	W	49.87	48.95	**53.98**	12.80	21.53
6	C	29.46	30.55	**33.30**	7.90	10.24
	F	41.66	41.08	**43.26**	6.68	29.76
	P	53.35	54.37	**55.81**	15.09	20.11
	W	38.01	39.69	**41.28**	11.03	15.45
7	C	**72.78**	**72.78**	**72.78**	11.11	41.11
	F	42.78	48.33	**53.33**	5.56	11.11
	P	**73.89**	**73.89**	**73.89**	31.67	15.00
	W	46.67	48.89	**57.22**	15.56	20.00
8	C	**58.31**	51.60	55.58	0.95	21.69
	F	48.27	39.74	**50.17**	6.58	15.15
	P	60.48	63.25	**67.06**	7.62	19.74
	W	41.60	48.27	**49.83**	11.34	11.26
9	C	**86.36**	81.82	72.73	27.27	36.36
	F	**50.00**	40.91	45.45	4.55	4.55
	P	**77.27**	72.73	63.64	36.36	31.82
	W	45.45	40.91	**50.00**	9.09	18.18

Table 3.6: Accuracy percentages after k – fold cross validation on each cluster obtained with the embeddings for the **Slovene names** of the food products.
Target: C – Carbohydrates, F – Fat, P – Protein, W – Water.
The numbers shown in bold in the table represent the overall best performance for each nutrient in the given cluster.

Cluster	**Target**	**Accuracy**				
		Word2Vec	**GloVe**	**Doc2Vec**	**Mean**	**Median**
1	C	**61.37**	54.11	52.00	1.00	17.47
	F	**44.26**	37.00	41.26	5.05	10.21
	P	**58.26**	50.00	53.26	13.16	14.32
	W	34.05	**37.05**	33.89	8.05	9.26
2	C	27.47	24.43	**32.15**	10.95	14.00
	F	**70.28**	67.04	64.69	7.93	60.55
	P	**63.72**	60.12	59.22	7.58	31.54
	W	**49.96**	43.28	48.38	17.89	19.55
3	C	**47.28**	41.51	45.00	11.13	16.13
	F	**67.42**	63.99	63.62	6.84	59.81
	P	**69.27**	65.83	69.20	8.75	58.55
	W	52.86	43.97	**54.34**	12.18	29.44
4	C	34.78	28.49	**40.33**	12.95	16.93
	F	**70.13**	67.74	66.40	4.79	62.43
	P	**72.50**	69.58	70.38	7.23	66.07
	W	54.24	47.66	**55.79**	11.24	33.86
5	C	**47.63**	41.40	45.95	9.58	15.80
	F	**66.12**	62.80	62.38	4.57	42.43
	P	**66.12**	64.47	64.47	8.73	52.38
	W	48.18	41.48	**51.02**	12.80	21.12
6	C	31.42	25.61	**33.74**	7.90	10.75
	F	39.34	34.97	**44.42**	6.68	29.98
	P	53.36	50.73	**63.13**	15.09	20.33
	W	41.21	34.67	**41.85**	11.03	15.45
7	C	**72.78**	67.78	**72.78**	11.11	41.11
	F	**58.33**	37.78	48.33	5.56	11.11
	P	63.89	63.89	**69.44**	31.67	15.00
	W	**46.11**	36.67	41.11	15.56	20.00
8	C	56.41	49.91	**56.45**	0.95	21.69
	F	**47.32**	43.51	44.55	6.58	15.15
	P	**64.29**	59.70	61.43	7.62	19.74
	W	35.11	**36.80**	35.11	11.34	11.26
9	C	**86.36**	72.73	**86.36**	27.27	36.36
	F	**50.00**	31.82	**50.00**	4.55	4.55
	P	**63.64**	59.09	68.18	36.36	31.82
	W	**54.55**	36.36	45.45	9.09	18.18

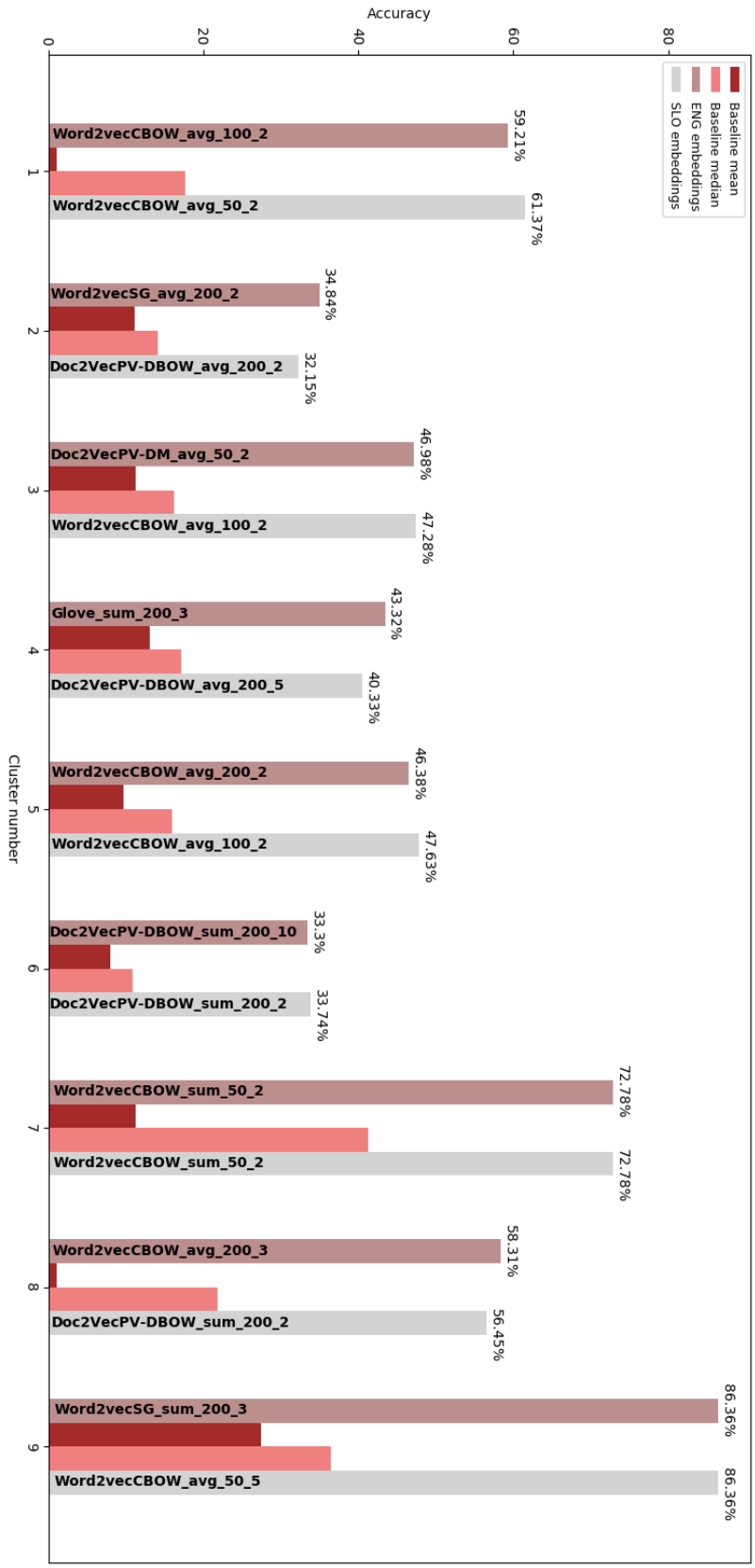

Figure 3.2: Best prediction accuracies for carbohydrates predictions obtained from the embeddings for the English names and Slovene names for each cluster compared to the baseline mean and median for the particular cluster.

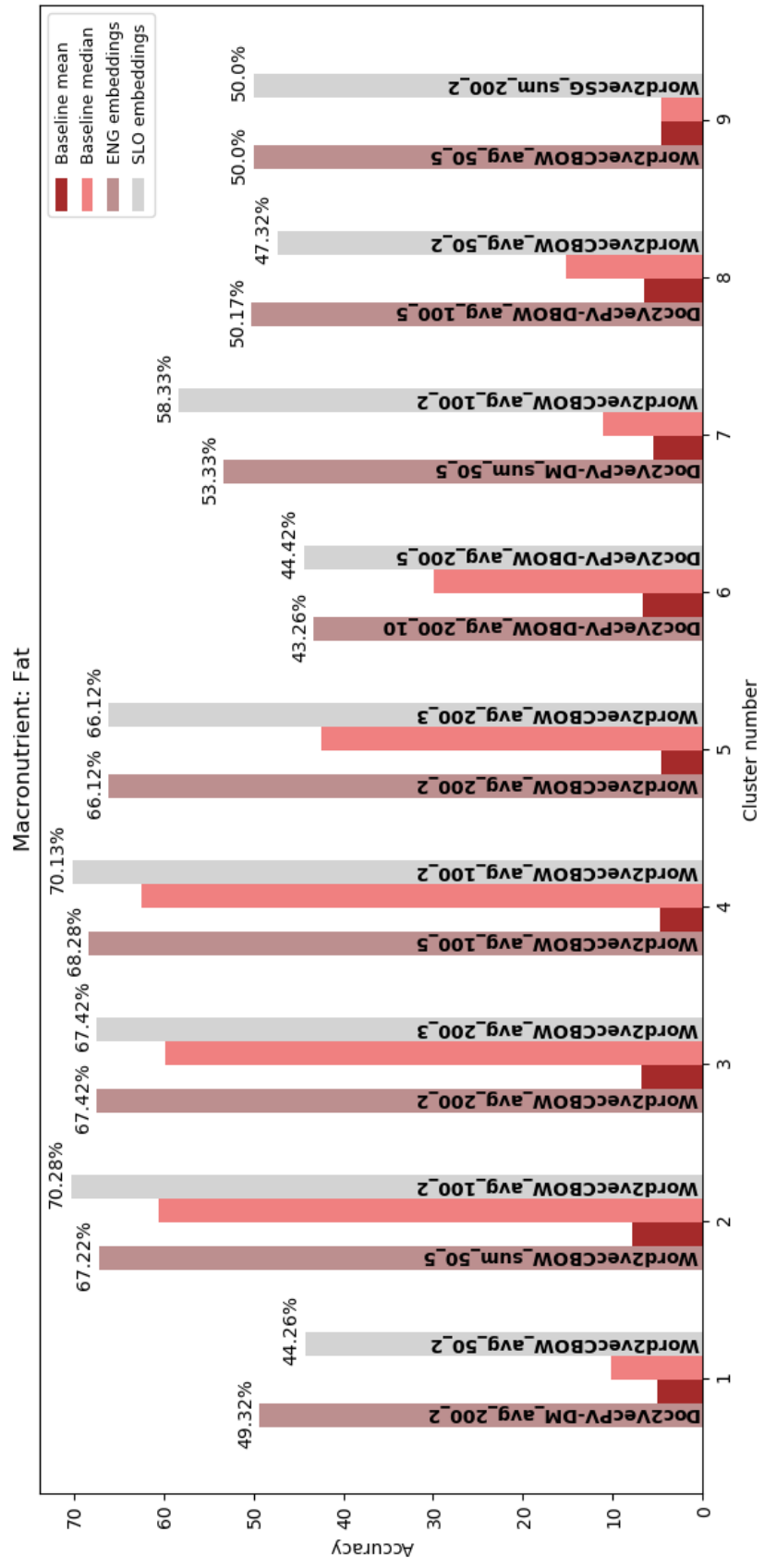

Figure 3.3: Best prediction accuracies for fat predictions obtained from the embeddings for the English names and Slovene names for each cluster compared to the baseline mean and median for the particular cluster.

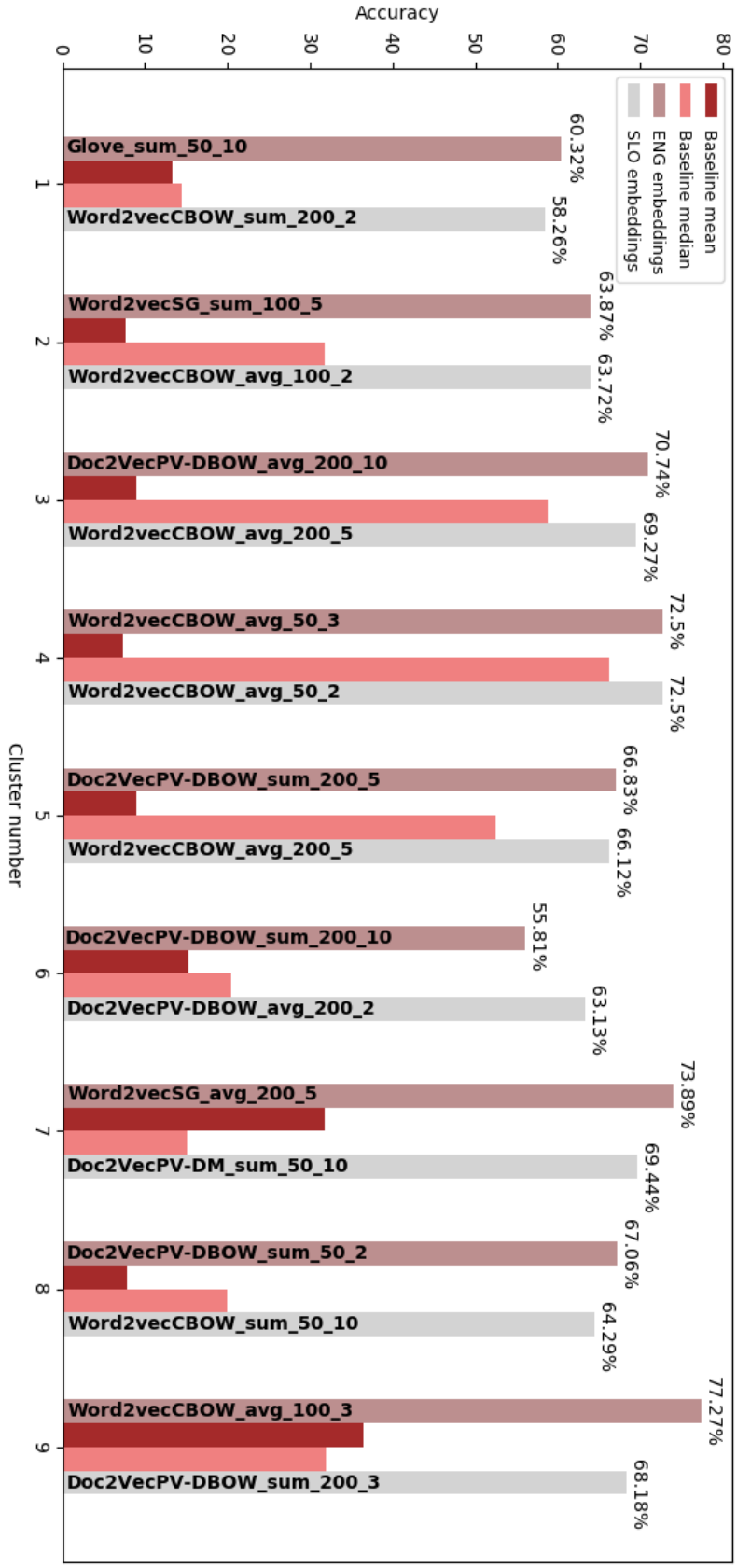

Figure 3.4: Best prediction accuracies for protein predictions obtained from the embeddings for the English names and Slovene names for each cluster compared to the baseline mean and median for the particular cluster.

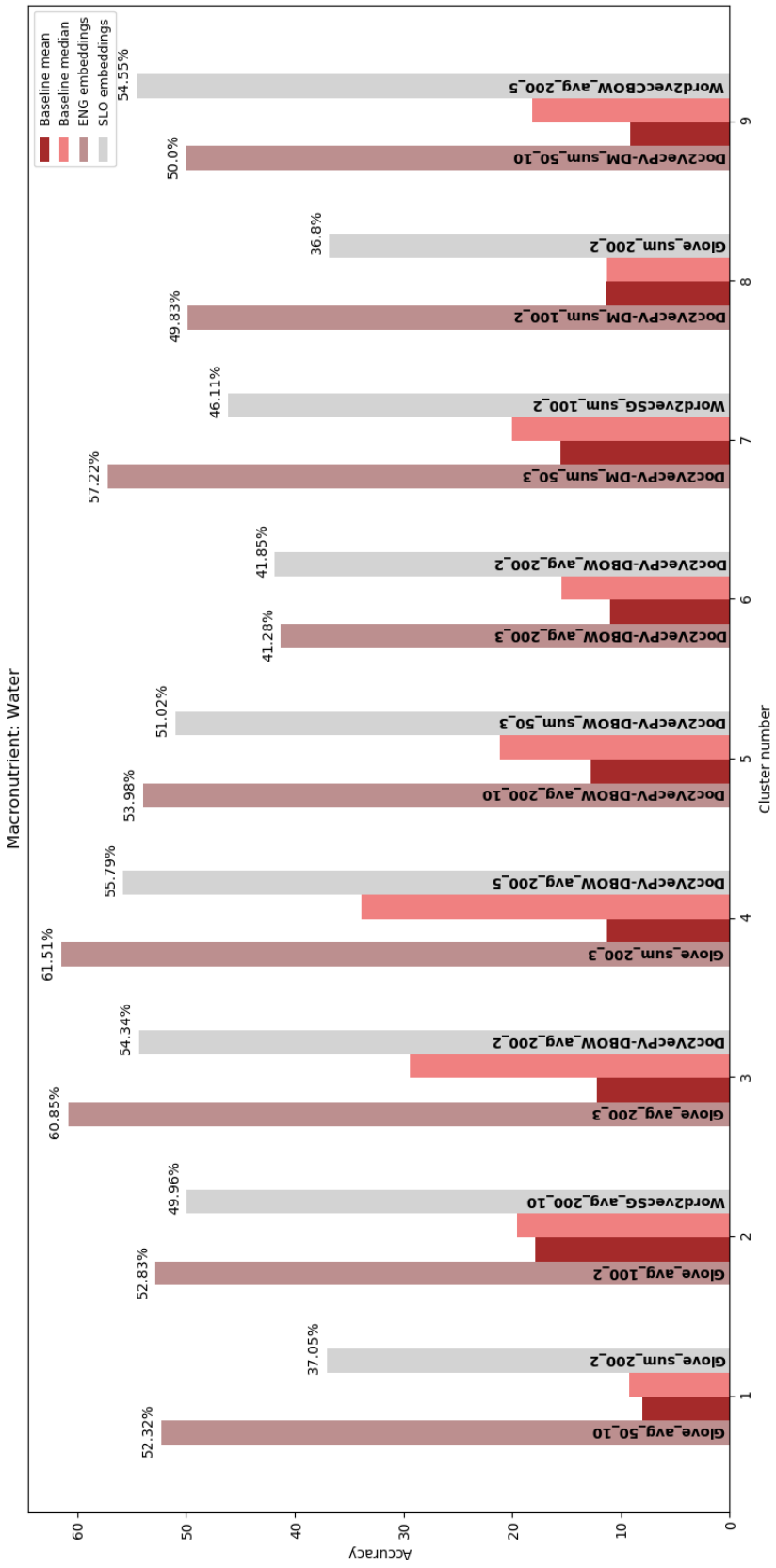

Figure 3.5: Best prediction accuracies for water predictions obtained from the embeddings for the English names and Slovene names for each cluster compared to the baseline mean and median for the particular cluster.

Table 3.7: Embedding and regression algorithms which yielded highest accuracies for each nutrient prediction in each cluster.
Target: C – Carbohydrates, F – Fat, P – Protein, W – Water.
Regression Algorithm: EN – Elastic Net, R – Ridge, L – Lasso, LN – Linear Regression.

Cluster	Target	Embedding Algorithm		Regression Algorithm	
		ENG	SLO	ENG	SLO
1	C	Word2VecCBOW_avg_100_2	Word2VecCBOW_avg_50_2	EN	R
	F	Doc2VecPV-DM_avg_200_2	Word2VecCBOW_avg_50_2	L	R
	P	GloVe_sum_50_10	Word2VecCBOW_sum_200_2	L	R
	W	GloVe_avg_50_10	GloVe_sum_200_2	EN	R
2	C	Word2VecSG_avg_200_2	Doc2VecPV-DBOW_avg_200_2	R	R
	F	Word2VecCBOW_sum_50_5	Word2VecCBOW_avg_100_2	L	R
	P	Word2VecSG_sum_100_5	Word2VecCBOW_avg_100_2	R	R
	W	GloVe_avg_100_2	Word2VecSG_avg_200_10	R	R
3	C	Doc2VecPV-DM_avg_50_2	Word2VecCBOW_avg_100_2	R	EN
	F	Word2VecCBOW_avg_200_2	Word2VecCBOW_avg_200_3	R	R
	P	Doc2VecPV-DBOW_avg_200_10	Word2VecCBOW_avg_200_5	R	R
	W	GloVe_avg_200_3	Doc2VecPV-DBOW_avg_200_2	R	EN
4	C	GloVe_sum_200_3	Doc2VecPV-DBOW_avg_200_5	R	EN
	F	Word2VecCBOW_avg_100_5	Word2VecCBOW_avg_100_2	L	R
	P	Word2VecCBOW_avg_50_3	Word2VecCBOW_avg_50_2	L	R
	W	GloVe_sum_200_3	Doc2VecPV-DBOW_avg_200_5	R	EN
5	C	Word2VecCBOW_avg_200_2	Word2VecCBOW_avg_100_2	R	L
	F	Word2VecCBOW_avg_200_2	Word2VecCBOW_avg_200_3	R	R
	P	Doc2VecPV-DBOW_sum_200_5	Word2VecCBOW_avg_200_5	EN	R
	W	Doc2VecPV-DBOW_avg_200_10	Doc2VecPV-DBOW_sum_50_3	R	L
6	C	Doc2VecPV-DBOW_sum_200_10	Doc2VecPV-DBOW_sum_200_2	R	R
	F	Doc2VecPV-DBOW_avg_200_10	Doc2VecPV-DBOW_avg_200_5	R	R
	P	Doc2VecPV-DBOW_sum_200_10	Doc2VecPV-DBOW_avg_200_2	R	R
	W	Doc2VecPV-DBOW_avg_200_3	Doc2VecPV-DBOW_avg_200_2	R	R
7	C	Word2VecCBOW_sum_50_2	Word2VecCBOW_sum_50_2	LN	LN
	F	Doc2VecPV-DM_sum_50_5	Word2VecCBOW_avg_100_2	EN	LN
	P	Word2VecSG_avg_200_5	Doc2VecPV-DM_sum_50_10	LN	EN
	W	Doc2VecPV-DM_sum_50_3	Word2VecSG_sum_100_2	LN	LN
8	C	Word2VecCBOW_avg_200_3	Doc2VecPV-DBOW_sum_200_2	R	R
	F	Doc2VecPV-DBOW_avg_100_5	Word2VecCBOW_avg_50_2	L	R
	P	Doc2VecPV-DBOW_sum_50_2	Word2VecCBOW_sum_50_10	EN	R
	W	Doc2VecPV-DM_sum_100_2	GloVe_sum_200_2	R	R
9	C	Word2VecSG_sum_200_3	Word2VecCBOW_avg_50_5	L	LN
	F	Word2VecCBOW_avg_50_5	Word2VecSG_sum_200_2	LN	LN
	P	Word2VecCBOW_avg_100_3	Doc2VecPV-DBOW_sum_200_3	LN	L
	W	Doc2VecPV-DM_sum_50_10	Word2VecCBOW_avg_200_5	L	LN

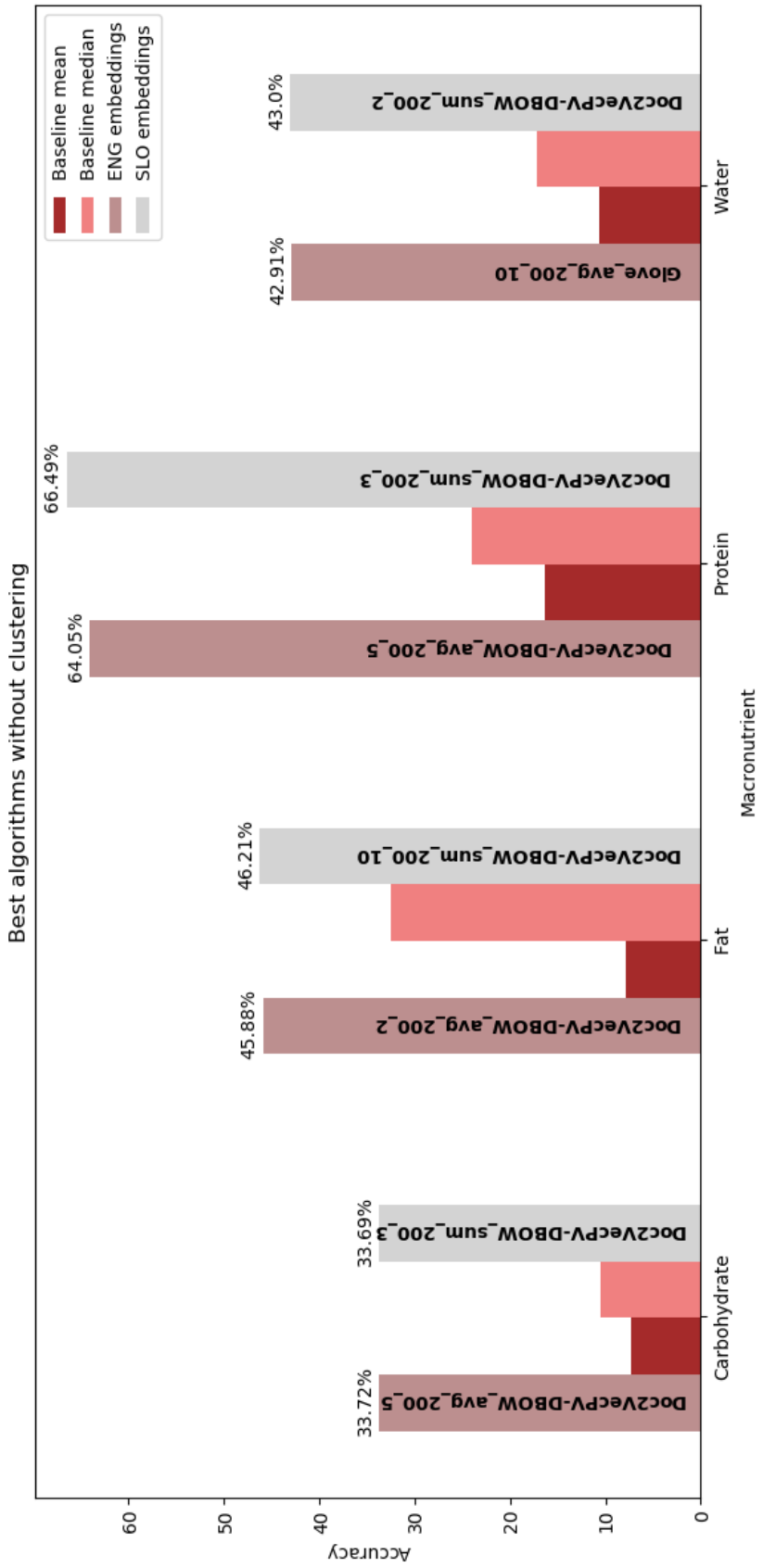

Figure 3.6: Best prediction accuracies for each nutrient obtained from the embeddings for the English and Slovene names compared to the baseline mean and median from the whole dataset.

best results were obtained without clustering the data.

Table 3.8: Embedding and regression algorithms which yielded highest accuracies for each nutrient prediction on the whole dataset (without clustering).
Target: C – Carbohydrates, F – Fat, P – Protein, W – Water.
Regression Algorithm: EN – Elastic Net, R – Ridge, L – Lasso, LN – Linear Regression

Target	Embedding Algorithm		Regression Algorithm	
	ENG	SLO	ENG	SLO
C	Doc2VecPV-DBOW_avg_200_5	Doc2VecPV-DBOW_sum_200_3	R	R
F	Doc2VecPV-DBOW_avg_200_2	Doc2VecPV-DBOW_sum_200_10	L	EN
P	Doc2VecPV-DBOW_avg_200_5	Doc2VecPV-DBOW_sum_200_3	R	R
W	GloVe_avg_200_10	Doc2VecPV-DBOW_sum_200_2	R	L

3.4.5 Discussion

From these results, it is worth arguing that modeling ML techniques on food data previously clustered based on FoodEx2 codes would yield better results than predicting on the whole dataset. If we compare the performances of the three embedding algorithms, it is hard to argue if one outperformed the others, or if one under-performed compared to the other two. This outcome is due to the fact that we are dealing with fairly short textual descriptions.
Given the fact that the results with the clustering are better than the results without, and we rely so strongly on having the FoodEx2 codes in order to cluster the foods, the availability of the FoodEx2 codes is of big importance and therefore a limitation of the methodology. For this purpose, we can rely on a method such as StandFood [51], which is a natural language processing methodology developed for classifying and describing foods according to FoodEx2. When this limitation is surpassed, the application of our method can be fully automated.
As mentioned previously it was decided to include water content as it is an instant indicator of whether the food is dry or not (example: cranberries vs. dried cranberries). From further exploration of the dataset we determined that there are foods which are traditional Slovenian food/dishes or the name of the food is typical only for Slovenia. While the English name includes a description of what the food is. For example: “Mlinci” and “Mlinci (Slovenian pasta tatters prepared from thin dried dough)”, “Prežganka” and “Soup made by roasting and browning flour”, “Jota” and “Yota (istrian stew - beans, sauerkraut, potatoes, bacon, spare rib)”, “Rizi bizi” and “Rice with peas, stewed”. As we can see a lot of these descriptions included in the English names but not in the Slovene names contain words that are crucial for water content in foods: “dried, soup, stewed”.
From a theoretical viewpoint this methodology considers the benefits of using RL as the base of a predictive study, and proves that dense real-valued vectors can capture enough semantics even from a short text description (without including the needed details for the task in question – in our case, measurements or exact ingredients) in order to be considered in a predictive study for complicated and value-sensitive task such as predicting nutrient content. This study offers a fertile ground for further exploration of RL and considering more complex embedding algorithms – using transformers [24], [147] and fine tuning them for this task. From a managerial viewpoint the application of this methodology opens up many possibilities for facilitating and easing the process of calculating nutrient content,

which is crucial for dietary assessment, dietary recommendations, dietary guidelines, nutrient tracking, and other such tasks which are key tools for doctors, health professionals, dietitians, nutritional experts, policy makers, professional sport coaches, athletes, fitness professionals, etc.

Chapter 4

Exploring Knowledge Domain Bias on a Prediction Task

In this chapter we present an extension of the ML/DM pipeline presented in Chapter 3. The pipeline is extended in order to test how the domain bias affects the prediction results, by incorporating two different external semantic resources from the Food and Nutrition domain. The chapter begins with the problem definition, what led to the decision of doing this study, then we present the related work as an addition to what is mentioned in Chapter 2. Next, we describe the methodology, the two external domain semantic resources, the data used in the experiments, the experimental setup and the results obtained from the experiments. At the end, we conclude with a discussion of the results.

4.1 Problem Definition

Nutrient content can vary a lot from one complex food to another, even though they have roughly the same type of ingredients, which complicates the nutrient tracking and calculating, and makes the possibility of predicting nutrient content extremely unlikely. In Chapter 3, we proposed an approach – a ML/DM pipeline, published in [39] (called P-NUT), for predicting nutrient values of a food item considering learned vector representations of text describing the food item. The ML/DM pipeline consists of three parts: RL part – learning vector representations from short text descriptions of recipes; unsupervised ML – introducing domain knowledge for obtaining separate clusters of data; and supervised ML – obtaining predictions for the nutrient values of the recipes.

When it comes to nutrient content, recipes and foods in general can be very unbalanced. In a dataset containing a large variety of foods, one nutrient can have values from 0 grams to 100 grams per 100 grams of the certain food, for example the content of fat can go from "fat-free" foods to "fat-based" foods (ex. different kinds of nut butters), therefore, a general model for prediction will not be efficient in nutrient prediction. For this reason, the domain knowledge incorporated in the unsupervised ML part of the methodology is very much needed for obtaining a good prediction model. In Chapter 3, we introduced clustering as a method to separate foods in order to obtain clusters (groups) of foods with similar characteristics. Subsequently, on these separate clusters we predict the nutrients with applying supervised ML. Prior to our proposed ML/DM pipeline [39], predicting nutrients as a task was viewed as a process of calculating or estimating nutrients from measurements and exact ingredients [110]–[112], which is a multi-step procedure that involves multiple demanding calculations that can be done only when all the ingredients and measurements

are available. When this data are not available, this procedure gets more complicated [110], [111], and the problem can only be solved by borrowing nutrient values from other similar foods and/or products.

In this study, we focus on sensitivity analysis of the unsupervised ML part, i.e. how introducing different types of domain knowledge affects the results from the supervised ML part. The two criteria that we used to explore the food and nutrition domain knowledge bias for predicting nutrients are – the FoodEx2 classification system [148], and the FSA traffic light system [149]. These two systems have domain knowledge inferred in them, are constructed and developed with food and nutrition domain experts, give us different insights on how to classify and separate foods, and are two of the world's most established systems for classifying foods.

4.2 Related Work

To the best of our knowledge, P-NUT [39], presented in Chapter 3, is the first methodology for predicting nutritional content of foods/recipes using only short text description. There has been some work involving ML done in this direction, mainly involving image recognition: employing different deep learning models for accurate food identification and classification from food images [124], dietary assessment through food image analysis [125], calculating calorie intake from food images [126], [127]. All this work is in the direction of predicting total calories, and strongly relies on textual data retrieved from the Web. There are numerous mobile and web applications for tracking nutrient intake [138], [139]. Systems like these are used for achieving dietary goals, allergy management or simply maintaining a healthy balanced diet. The biggest downside is the fact that they require manual input of details about the meal/food.

In Chapter 2, we explained the concept behind RL and the RL methods that can be used for words and paragraphs/documents, as well as graphs.
To explore how different external semantic resources affect the results from the prediction task in the presented ML/DM pipeline in Chapter 3 we used Word2Vec [19], [20] and GloVe [21], to generate word embeddings, Doc2Vec [22] to generate paragraph/phrase embeddings, and Poincaré [71] to generate graph embeddings. The first semantic resource of question in this study is the FoodEx2 classification system [49], which we described in details in Chapter 2, Subsection 2.1.1. Because this information, i.e., the FoodEx2 codes, is not always available in the recipe, or any other datasets containing food and nutrition-related data, there is a need of a method for obtaining these codes, and this method is StandFood [51], which we described in Chapter 2, Subsection 2.1.2. We use the StandFood method to obtain the matches of the food names, i.e. recipe names with the FoodEx2 classification system.

4.3 Methodology

To explore how including different domain specific semantic resources affects the nutrient prediction task, we propose an extension to the ML/DM pipeline, i.e. the P-NUT methodology as presented in Figure 4.1.

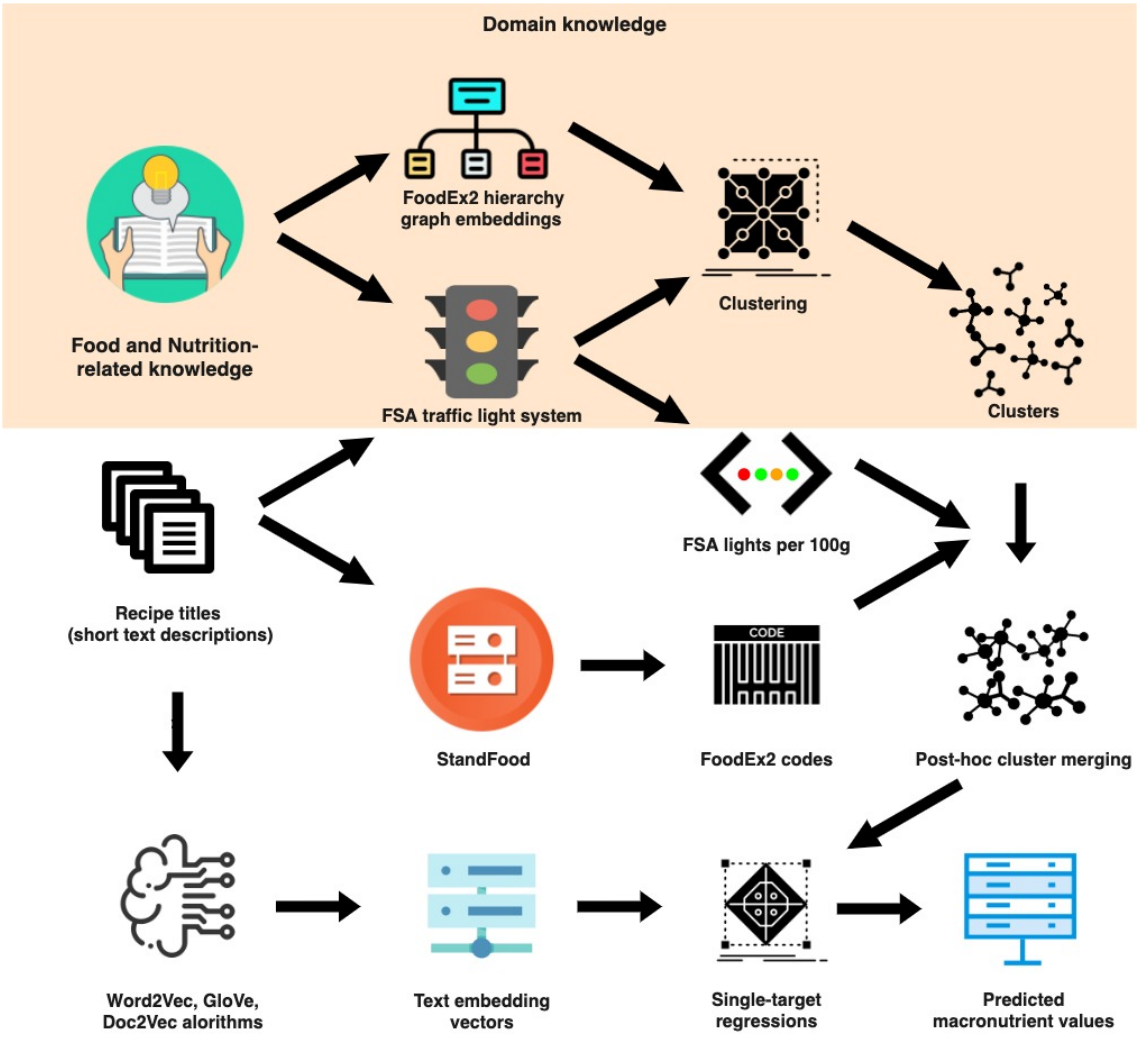

Figure 4.1: Inferring domain knowledge in P-NUT.

4.3.1 Domain knowledge criteria

1. FoodEx2 classification system – FoodEx2 is the second version of FoodEx [148], which is a standardized system for food classification and description developed by the European Food Safety Authority (EFSA), it has domain knowledge embedded in it and it contains descriptions of a vast set of individual food items combined in food groups and more broad food categories in a hierarchy that exhibits a parent-child relationship. These FoodEx2 codes already contain domain knowledge, and based on them food items are grouped in food groups and broader food categories in the FoodEx2 hierarchy.

2. FSA traffic light system – The Food Standards Agency (FSA) introduced the traffic light system [149] in order to determine how healthy a recipe/food product is. The FSA traffic light system gives independent expert scientific dietary advice in order to help individuals make healthier choices quickly and easily. The main goal of the FSA traffic light system is for the food industry to incorporate this traffic light system and implement it on the food labels, which would quickly give consumers indications about the nutritional content of the food product, and therefore make a more informed decision about their diet.
   There are three colors in this system – green, orange (amber), and red, which follow a hierarchy of "healthiness". These colors show at a glance if the food has low, medium or high amounts of fat, saturated fat, sugars and salt.

4.3.2 Tolerance for nutrient values

The task we are exploring is prediction of nutrient values. Nutrient values which are expressed in grams (or milligrams, micro-grams for certain groups), can have tolerances defined by international legalizations and regulations. As mentioned in Chapter 3, these tolerance levels are defined by The European Commission Health and Consumers Directorate General in 2012 in a guidance document [46], with the aim to provide advised recommendations for calculation of the acceptable differences between quantities of nutrients on the label declarations of food products and the ones established in Regulation EU 1169/2011 [47]. These tolerance levels are different for different nutrients.

Table 4.1: Tolerated differences in nutrition content in foods besides food supplements.

Nutrient	Quantity per 100 g	Tolerances (allowed deviations in quantity)
Salt	$<$ 1.25 g per 100 g	$\pm$0.375 g
	$\geq$ 1.25 g per 100 g	$\pm$20%
Saturates	$<$ 4 g per 100 g	$\pm$0.8 g
	$\geq$ 4 g per 100 g	$\pm$20%
Fat	$<$ 10 g per 100 g	$\pm$1.5 g
	10-40 g per 100 g	$\pm$20%
	$>$ 40 g per 100 g	$\pm$8 g
Protein, Sugars	$<$ 10 g per 100 g	$\pm$2 g
	10-40 g per 100 g	$\pm$20%
	$>$ 40 g per 100 g	$\pm$8 g

From the tolerance levels stated in [46], for our particular case we used the tolerance levels for the nutrition declaration of foods that do not include food supplements. The ones of importance for this study are given in Table 4.1. In the table, the allowed deviations

are presented for each of the nutrients we are predicting, depending on their quantity in 100 grams of the food in matter. Including these tolerance levels in the evaluation process of our ML/DM pipeline for determining the accuracy of the predicted nutrient values, we are embedding domain knowledge in the evaluation process.

4.4 Results and Discussion

This section explains the evaluation process of the presented methodology – the experimental setup and the results obtained, as well as a discussion about the outcome of the experiments.

4.4.1 Data

Food- and nutrition-related data comes in many different forms and formats: data about the nutritional composition of food (food composition data); data about actual and recommended food consumption on local, national and even international level (food consumption data); recipe data; and many other kinds of data (e.g. branded data). When we talk about recipe data we usually mean text descriptions of composite foods, which are readily available on user-based recipe websites. Recipe data can contain: recipe description – which is usually just a short textual description of the recipe. The dataset used for the evaluation of the ML/DM pipeline in Chapter 3 was collected in Slovenia for the aims of the EFSA EU Menu project [148], comprising data about simple food products, and recipe dishes. For each food item it contained the following information: its name in Slovene and English, its FoodEx2 code, and its nutrient values for: energy, water, fat, carbohydrates, and proteins, recipe instructions – which is a longer text with a description of the cooking method and preparation instructions, and list of ingredients – the recipe's ingredients alongside their quantities.

For this study we use the dataset Recipe1M, available online [150], which is a large-scale, structured corpus of over one million cooking recipes and 13 million food images. However, from those one million recipes in our focus are the ones that contain details like nutrients – a total of 51,235 recipes. The following information is available for each of the 51,235 recipes:

1. Recipe title – a short textual description of the recipe;
2. Ingredients – list of ingredients;
3. Recipe instruction – long textual description of step-by-step instructions for preparing the recipe;
4. Nutrient content of ingredients – quantity in grams of fat, protein, saturates, sodium, and sugar per 100 grams of the ingredient for each ingredient;
5. Quantity of each ingredient;
6. Units of measurement per each ingredient – according to the household measurement system (cup, tablespoon, teaspoon, etc.);
7. Weight in grams per each ingredient;
8. Nutrient content – quantity in grams of fat, protein, salt, saturates, and sugars per 100 grams of the recipe;

9. FSA traffic light labels per 100 grams – for each of the four nutrients (fat, salt, saturates and sugars) one of the three labels from the FSA traffic light system is assigned:
   (a) red – high content;
   (b) orange – medium content;
   (c) green – low content.

In Table 4.2, a few example data instances from the dataset are given.

Table 4.2: Dataset structure and example data instances

Recipe title	**FSA lights per100g**			
	fat	**salt**	**saturates**	**sugars**
Salt Free, Low Cholesterol Sugar Cookies Recipe	red	orange	orange	orange
Pacific Wasabi Sauce for Grilled Tuna	red	orange	red	green
Barbeque Brisket Rub	orange	red	green	green
Frozen Banana Smoothie	green	green	green	red

4.4.2 Data pre-processing

The Recipe1M dataset contains detailed information about each recipe. The data of interest here are: the recipe titles, the FSA traffic light labels for fat, salt, saturates, and sugars, and the quantities per 100 grams of the recipe of the five nutrients for prediction. First, the text descriptions are tokenized, beforehand the punctuation signs and numbers that represent quantities are removed, whereas the percentage values (e.g.: of fat, of sugar, of cocoa, etc.) which contain valuable information concerning the nutrient content, and stop words which add meaning to the description, are kept. Next, the tokenized words are lemmatized [144]. The titles/descriptions of the recipes after this pre-processing are ready to undergo the next steps.

4.4.3 Experimental setup

After the data pre-processing, the next step is to apply the algorithms for generating embeddings. We generate the vector representations of the recipe titles in two different ways with three different algorithms:

1. Word embeddings – generating vector representations with the Word2Vec and GloVe algorithms for each word of the description and merging the separate word embeddings into sentence embedding by summing or averaging the separate vectors.

2. Paragraph embeddings – generating paragraph/sentence embeddings for the whole description with the Doc2Vec algorithm.

For the algorithm implementation we use the *Gensim* [145] library in Python, and the *Word2Vec* and *Doc2Vec* packages for the algorithms respectively. The GloVe algorithm is implemented with the scripts and instructions provided by the authors [151]. We run the three algorithms for different values for the parameters of each. For the vector dimensions the values chosen are 50, 100 and 200, because, as mentioned in Chapter 3, they are among the frequently used dimensions for textual embeddings for these algorithms, and are efficient in means of execution time and computational power. The values for the sliding

window (which is a number that indicates the maximum distance between the current and predicted word within a sentence) chosen are 2, 3, and 5, because we are dealing with fairly short textual descriptions – the maximum number of word per recipe title is 19. For the Word2Vec and the Doc2Vec algorithms the two existing architecture are considered, CBOW and SG for Word2Vec, PV-DM and PV-DBOW for Doc2Vec. For the two word embedding algorithms – Word2Vec and GloVe we also train different models with the two heuristics for merging the word embeddings (sum and average). As for the Doc2Vec algorithm the non-concatenative mode is used (separate models for the sum and the average) as the model with using the concatenation of context vectors rather than sum/average the result would be a much-larger model. Therefore, with the Word2Vec and the Doc2Vec algorithm 48 are trained, while with the GloVe a total of 24 models were trained.

Independently of this process, the data are clustered following the two criteria presented – the FoodEx2 clustering system and the FSA traffic light system. Since we do not have the FoodEx2 codes available, they are obtained beforehand with applying the StandFood [51] method on the dataset.

For the FoodEx2 clustering we are using the method presented in [74], where the FoodEx2 codes (based on their graph embeddings) are clustered into 230 clusters. This clustering is already done in [74]. The authors use the silhouette method to determine the number of clusters before using the clustering algorithm (PAM in their case). By matching the FoodEx2 codes of the clusters and the FoodEx2 codes of our dataset instances obtained by StandFood, the instances in our dataset are clustered, i.e. a cluster number from 1 to 230 is assigned to each recipe. After this initial clustering, because there are some empty clusters, the post-hoc cluster merging is performed, where we merge the clusters following a bottom-up approach. For the Recipe1M dataset, the parents on the forth level in the FoodEx2 hierarchy are chosen, and with this we obtain 47 clusters. After a close observation, 16 clusters containing very few elements are removed. One criteria that is chosen to be satisfied as well is – no cluster has less then 200 instances. As we have 200 dimensional vectors, i.e. 200 features for each input instance, therefore to avoid having more features than data points 200 is chosen as the minimal number of instances per cluster. The end result is 31 clusters.

For the FSA traffic light system, the clusters are generated by obtaining all the possible permutations of the three colors possible, i.e. calculating permutations with repetitions:

$$ {}_nP_r = n^r \tag{4.1} $$

Where $n = 3$ are the three colors in the FSA traffic light system – red, orange, and green, and $r = 4$ are the four nutrients for which we have these colors available – fat, salt, saturates, and sugars. Therefore, we have 81 possible combinations, i.e. 81 clusters. After clustering or, better said, dividing the recipes from the dataset according to these 81 possible combinations, 66 clusters of recipes are obtained, the rest – 15 combinations are not present in the dataset. Out of the 66 clusters, 10 clusters had very few examples – less then 200, which, as stated previously, is contradictory to our study. Thus, the final result is 55 clusters according to the FSA traffic light system. The process is described in Algorithm 4.1.

The next step is the actual predictive modeling, i.e. the supervised ML part – applying single-target regressions. This is done according to the following setup:

1. Select regression algorithms – Linear regression, Ridge regression, Lasso regression,

Algorithm 4.1: Unsupervised learning part when using FSA clustering.

Data: Dataset of n recipes represented with their FSA traffic light codes
Result: Dataset of n recipes grouped in k clusters

Get P – a set with all 81 possible permutations with repetitions of the FSA labels: "red", "orange", and "green";
Form a cluster for each permutation $p \in P$;
for *Each recipe* r **do**
  Identify in which cluster r belongs by its combination of FSA labels;
end
for *Each cluster* c **do**
  Get the number of instances $Num(c)$;
  if $Num(c) < 200$ **then**
    Remove cluster;
  else
    Keep cluster;
  end
end

and ElasticNet regression (using the Scikit-learn library in Python [146]).

2. Select parameter ranges for each algorithm and perform hyper-parameter tuning – Ranges and values are a priori given for all the parameters for all the regression algorithms. From all the combinations, the best parameters for the model training are selected with GridSearchCV (using the Scikit-learn library in Python [146]). This is done for each cluster separately.

3. Apply k-fold cross-validation to estimate the prediction error – We train models for each cluster using each of the selected regression algorithms. The models are trained with the previously selected best parameters for each cluster and then evaluated with cross-validation. For comparison of the regressors, the matched sample approach is chosen, i.e. using the same data in each iteration.

4. Apply tolerance levels and calculate accuracy – The accuracy is calculated according to the tolerance levels in 4.1. If a_i is the actual value of the i^th instance from the test set on a certain iteration of the k-fold cross-validation, and p_i is the predicted value of the same, i^th, instance of the test set, then:

$$d_i = |a_i - p_i| \tag{4.2}$$

d_i is the absolute difference between the two set values. We define a binary variable

that is assigned a positive value if the predicted value is in the tolerance level.

$$
\begin{aligned}
&allowed = 1, \text{ if } : \\
Salt &: \begin{cases} a_i < 1.25 \ \& \ d_i < 0.375 \\ a_i \geq 1.25 \ \& \ d_i \leq 0.2 \times a_i \end{cases} \\
Saturates &: \begin{cases} a_i < 4 \ \& \ d_i < 0.8 \\ a_i \geq 4 \ \& \ d_i \leq 0.2 \times a_i \end{cases} \\
Fat &: \begin{cases} a_i < 10 \ \& \ d_i < 1.5 \\ 10 \leq a_i < 40 \ \& \ d_i \leq 0.2 \times a_i \\ a_i \geq 40 \ \& \ d_i \leq 8 \end{cases} \\
Protein, Sugar &: \begin{cases} a_i < 10 \ \& \ d_i < 2 \\ 10 \leq a_i < 40 \ \& \ d_i \leq 0.2 \times a_i \\ a_i \geq 40 \ \& \ d_i \leq 8 \end{cases}
\end{aligned}
\tag{4.3}
$$

At the end we calculate the accuracy as the ratio of predicted values that are in the "allowed" range, i.e. tolerance level:

$$Accuracy = \frac{\sum_{i=1}^{n} allowed}{n} \tag{4.4}$$

Where n is the number of instances in the test set. The accuracy percentage is calculated for the baseline mean and baseline median as well – the percentage of baseline values (means and medians from each cluster) that falls in the tolerance level range. In this case – a_i is the actual value of the i^{th} instance from the test set on a certain iteration of the k-fold cross-validation, and instead of p_i in Equation 4.2 we have:

$$
b = \begin{cases} \frac{\sum_{i=1}^{m} x_i}{m}, \text{the baseline is the mean} \\ \frac{X_{[\frac{(m+1)}{2}]} + X_{[\frac{(m+1)}{2}]}}{2}, \text{the baseline is the median} \end{cases}
\tag{4.5}
$$

Where m is the number of instances in the train set, and X is the train set sorted in ascending order. The accuracy percentages are calculated for each fold in each cluster, and at the end for each cluster we calculate an average of the percentages from each fold.

4.4.4 Evaluation outcomes

In Figure 4.2, a graph that represents the highest accuracy percentages obtained with the FoodEx2 clustering method is presented, as well as the baseline mean and baseline median accuracy percentages for the cluster in question. In Figure 4.3, the same for the FSA traffic light clustering method is presented. In the graphs, for each target nutrient, we give the best result obtained with the embedding vectors and compare them with the baseline mean and median for the particular cluster. In the graphs, the embedding algorithm that yields the best results alongside with the parameters and heuristic is given as:

$$
E\_h\_d\_w = \begin{cases} h \in [sum, average], \text{is the chosen heuristic for merging the word embeddings} \\ din[50, 100, 200], \text{is the dimension of the embedding vectors} \\ win[2, 3, 5, 10], \text{is the sliding window parameter} \end{cases}
\tag{4.6}
$$

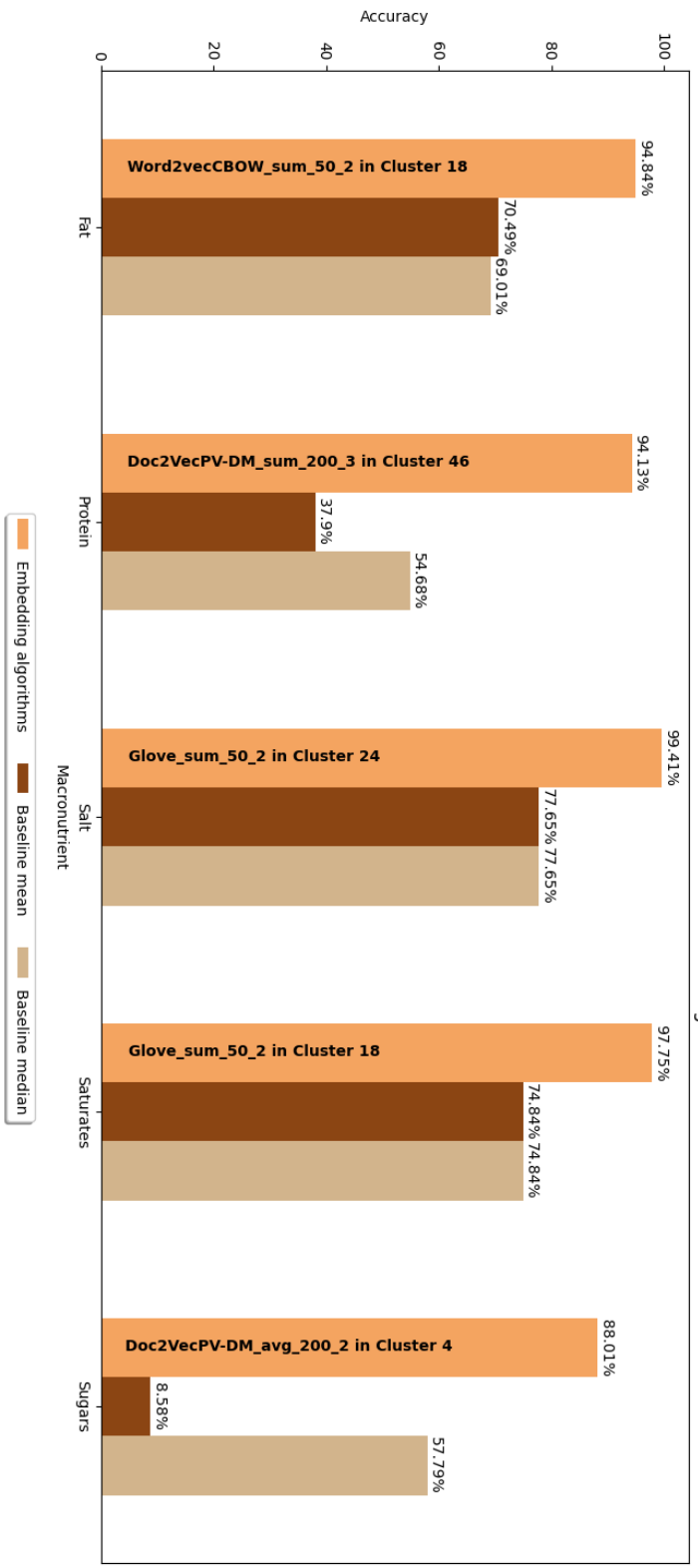

Figure 4.2: Highest accuracy percentages obtained with the FoodEx2 clustering method compared to the baseline mean and baseline median.

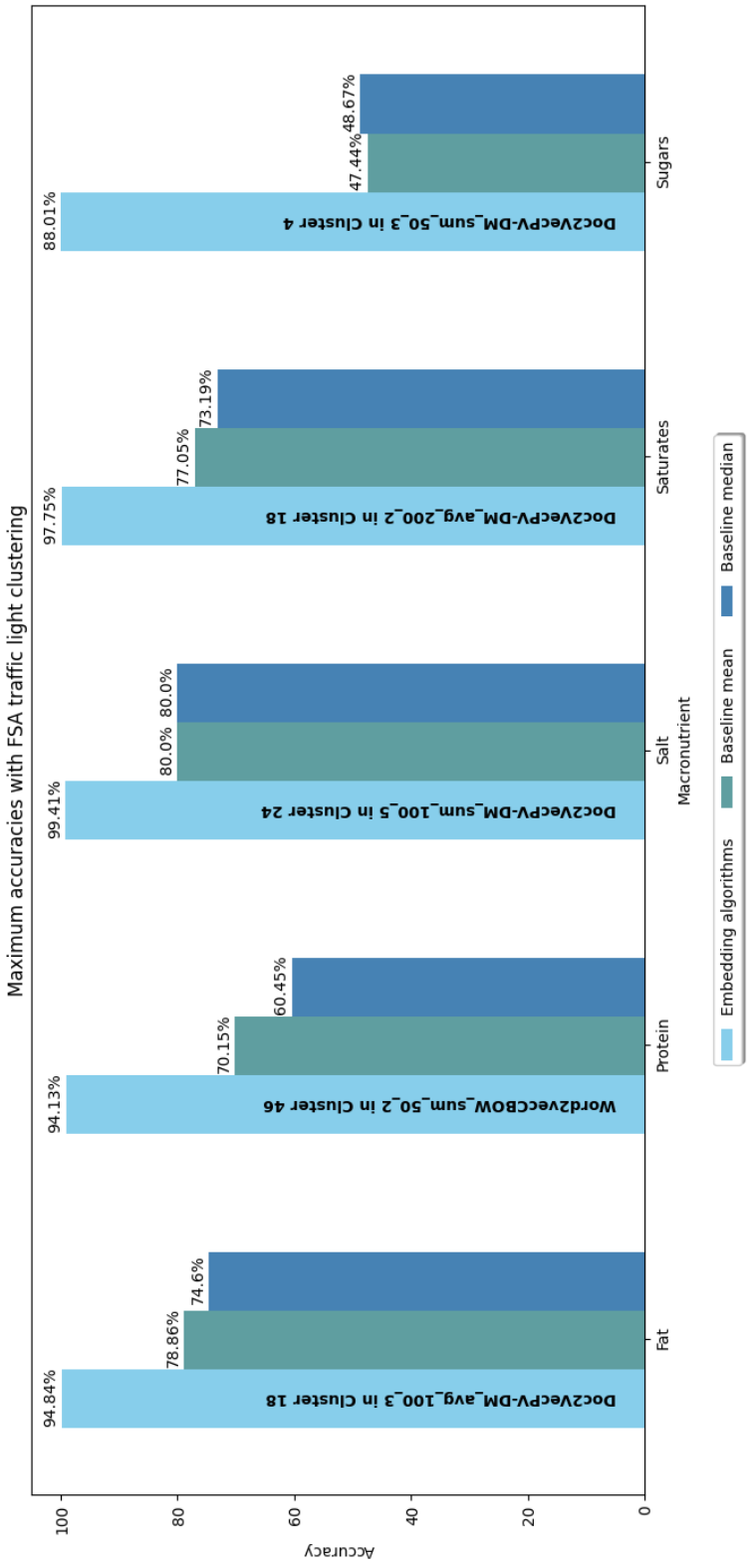

Figure 4.3: Highest accuracy percentages obtained with the FSA traffic light clustering method compared to the baseline mean and baseline median.

Where E is the embedding algorithm (Word2Vec, GloVe or Doc2Vec). We can see that which embedding algorithm yields the best results changes, but in all cases the predictions with the embedding vectors give better results than the baseline methods. From these graphs we can see that both of the clustering methods give much better results than the baseline mean and baseline median. For demonstration purposes in the graphs only the highest obtained accuracies are presented. The range of the percentage of accuracy for the FoodEx2 clustering approach across the 31 clusters is 68% – 99% and for the FSA traffic light clustering approach across the 55 clusters is 71% – 99%. At last, in Figure 4.4, we compare the highest percentages of accuracy obtained by both domain knowledge clustering approaches. As we can see they obtain very similar results, with a little favor for the FSA traffic light approach.

4.4.5 Discussion

When closer observing the clusters we can state that the benefits to the FSA traffic light clustering approach are that it is based strictly on the quantity of the nutrients, therefore we can have food items that belong to very different food groups, but have similar nutrient labels (contents). Whereas the FoodEx2 clustering approach puts together food items that belong to a similar food group, for example – types of fruit-based products, types of meat based product, egg dishes, etc. From these results, it is worth arguing that modeling ML techniques on food data with previously considering domain knowledge yields better results than predicting on the whole dataset. If we compare the performances of the three embedding algorithms, it is hard to argue if one outperformed the others, or if one underperformed compared to the other two. This outcome is due to the fact that we are dealing with fairly short textual descriptions. On the other hand, if we compare the results of the two approaches we chose for inferring domain knowledge in the dataset, we can say that both performed very similarly, with a slight favour to the FSA traffic light system, simply because of the fact that it is purely based on nutrient values, which is the task in matter for our prediction models.

In regard to the easiness of applying – the clustering based on the FSA traffic light system takes the win, as it is based on the nutrient values of the recipes. Therefore, if we want to replicate this on another dataset/s, the only data needed are the nutrient values for fat, salt, saturated fat and sugars, to obtain the four FSA lights. For the FoodEx2 clustering on the other hand, we need the FoodEx2 codes for the recipes, which, more often than not, are not available in recipe datasets. Even with the FoodEx2 codes available, we still need their graph embeddings, and the post-hoc cluster merging.

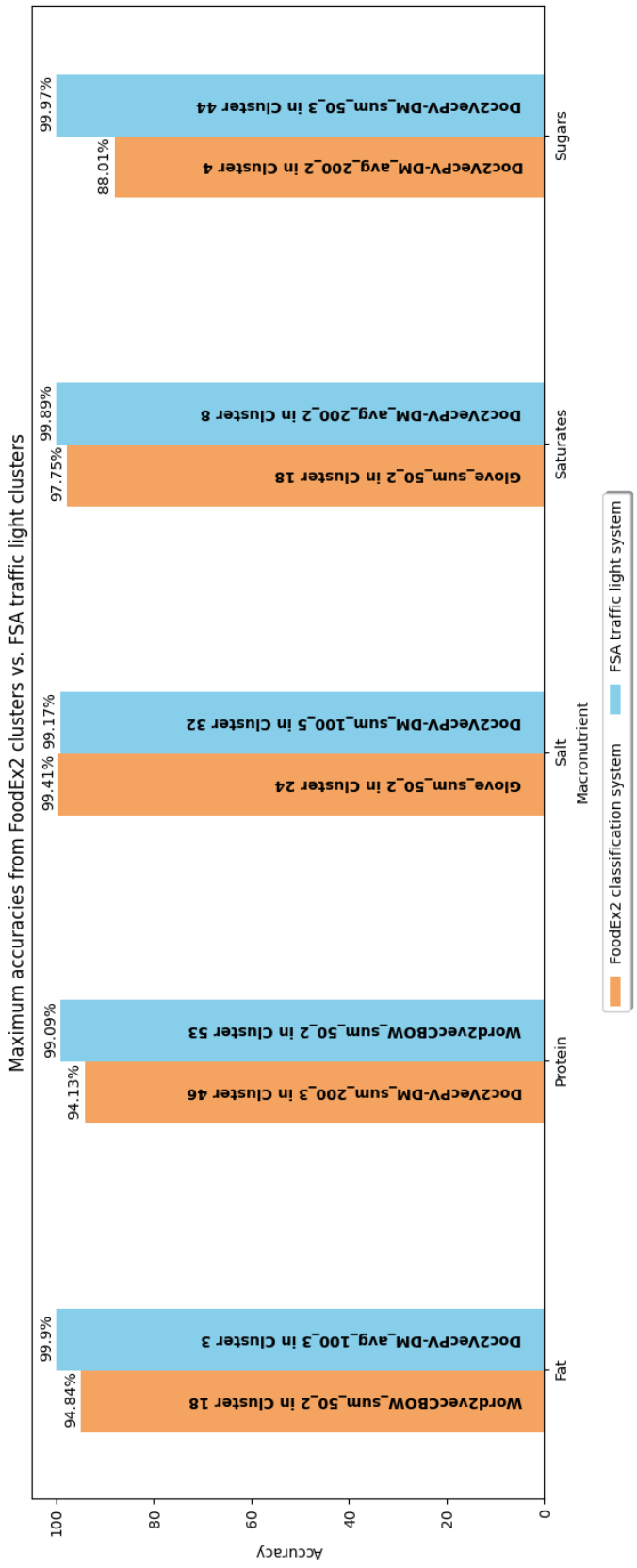

Figure 4.4: Comparing the highest accuracies obtained for each nutrient prediction with the FoodEx2 clustering and the FSA traffic light clustering.

Chapter 5

Domain Heuristic Fusion of Multi-word Embeddings for Nutrient Value Prediction

In this chapter, we present an extension of the P-NUT ML/DM pipeline presented in Chapter 3. The pipeline is extended in order to explore the effect on the prediction results when instead of using conventional heuristics for merging word/sentence embeddings, we use a heuristic that has domain knowledge embedded in it. We begin the chapter with the problem definition, the key point that guided us into doing this study, followed by additional related work besides what is already presented in Chapter 2, a detailed description of the methodology, the data used in the experiments, the experimental setup and the results obtained from the experiments. At the end, we conclude with a discussion of the results.

5.1 Problem Definition

Our proposed machine learning pipeline from Chapter 3 deals with nutrient prediction based on learned vector representations from word and paragraph embedding algorithms on short text descriptions – recipe names. In this study, we explored how the prediction results change when instead of using the vector representations of the recipe description, we use the embeddings of the list of ingredients. The nutrient content of one food – recipe depends on its ingredients and the quantities of those ingredients, therefore the text of the ingredients in contrast to the text in the short recipe description contains more relevant information for our task. Therefore, we define a domain-specific heuristic for merging the embeddings of the ingredients which combines the quantities of each ingredient as well in order to use them as features in machine learning models for nutrient prediction.

Calculating nutrient content is a very demanding and important task, and even though two foods can have roughly the same ingredients, their nutrient content can vary significantly. This makes nutrient tracking and calculating very challenging, and predicting nutrient content very complicated. Nutrient content calculation is usually a process of estimating and calculating the nutrient quantities from measurements and exact ingredients [110], [112], [152], and before our proposed ML/DM pipeline – P-NUT [39] (presented in Chapter 3) it has not been viewed as a prediction task. The instructions for calculating nutrient content from measurements and ingredients are pretty demanding, the detailed

procedure for calculation of the nutrient content of a multi-ingredient food has a few major steps: selection or development of an appropriate recipe, data collection for the nutrient content of the ingredients, correction of the ingredient nutrient levels for weight of edible portions, adjustment of the content of each ingredient for effects of preparation, summation of ingredient composition, final weight (or volume) adjustment, and determination of the yield and final volumes. And this is when all the ingredients and measurements are available, when the data for the ingredients are not available, then data for the uncooked ingredients is used with appropriate yield factors applied to adjust for weight changes and retention factors for nutrient losses or gains during cooking [153].

While in our proposed ML/DM pipeline we used the vector representation of the short text descriptions of the food products as input features to the ML algorithms, in this study we are working with recipe data and we propose using the embeddings of the lists of ingredients for each recipe as the input features. Each list of ingredients is a list of simple or complex foods, which are multi-word strings and not sentences, for example, the recipe name is "No bake oatmeal cookies" and the list of ingredients is as follows: "sugars, granulated; cocoa, dry powder, unsweetened; milk, fluid, 1% fat, without added vitamin a and vitamin d; butter, without salt; vanilla extract; peanut butter, smooth style, without salt; oats". This means their embeddings are non-contextualized, and in order to merge the vector representations of the separate ingredients we propose a new domain-specific heuristic which combines the quantities of the ingredients as well.

This idea came about from the mere fact that when working with big recipe datasets there can be multiple recipes with the same name, or even multiple recipes with the same ingredient list. When this happens, we are inherently beginning with an error – with poorly constructed representation vectors (input features). Namely, the RL part of the presented ML/DM pipeline would generate the same vector representation for two recipes if they have the same name (if we consider just the recipe descriptions) or the same list of ingredients (if we consider the list of ingredients) but have different quantities of the ingredients, and therefore different nutrient content. This entails that we want the ML models to predict two different values for the target nutrient value, given the same input, i.e. map one point from the feature space to two points from the performance space (Figure 5.1a).

On the other hand, if the quantities of the ingredients (which are crucial for calculating the nutrient content) are incorporated when generating the embeddings, the two recipes with the same ingredient list and different quantities will have two different vectors, therefore will be mapped to two different points in the feature space, making it logical for the ML models to predict two different values for the target nutrient i.e., map them to two different points in the performance space (Figure 5.1b). With using this heuristic, the results from this study show how domain knowledge can lead to better results when considering a prediction task in the Food and Nutrition domain.

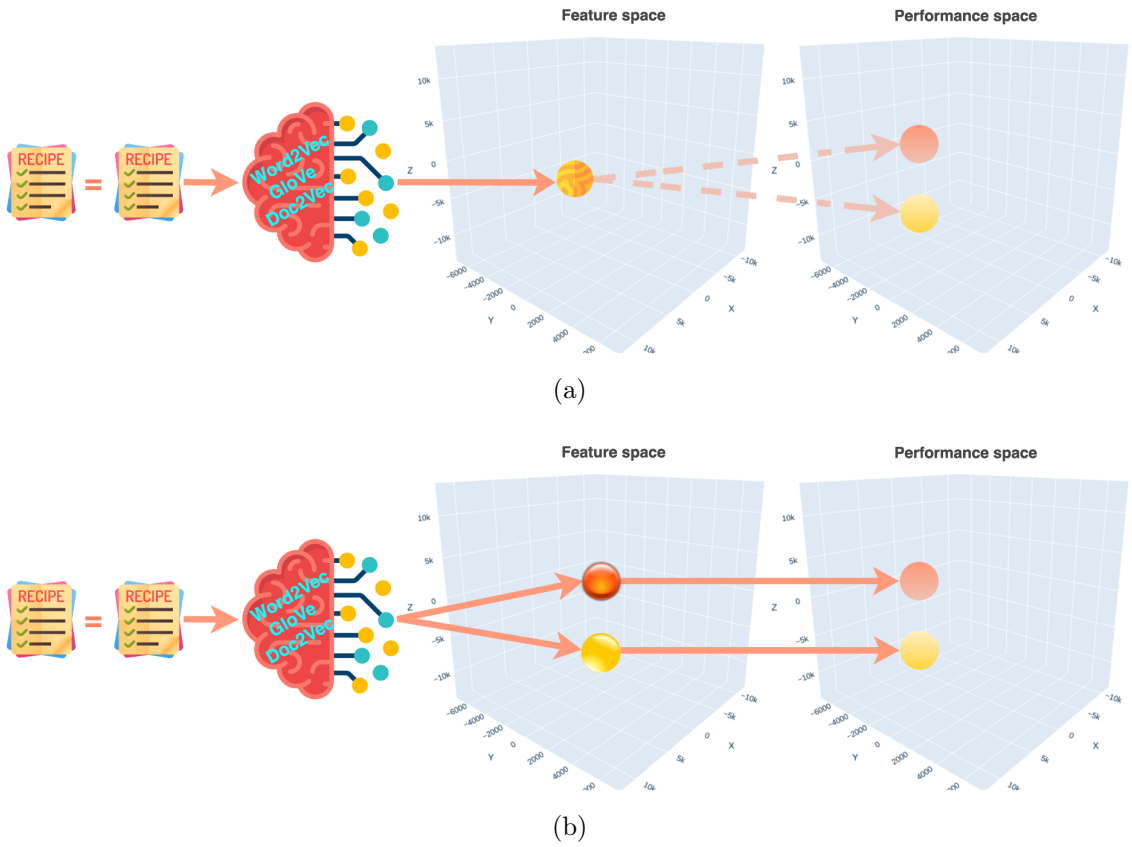

Figure 5.1: From feature space to performance space: (a) without taking into account the differences in ingredients; (b) with taking into account the differences in ingredients.

5.2 Related Work

Given the historical success of text embeddings in NLP, when evaluating our ML/DM pipeline, we use the most well-known word/paragraph embedding algorithms – Word2Vec [19], GloVe [21], and Doc2Vec [22]. These embedding algorithms are non-contextualized or context-independent, i.e. there is just one vector (numeric) representation for each word (i.e., Word2vec and GloVe) or each chunk of words/paragraph (i.e., Doc2vec). Or to put it in other words – different senses of the word (if there are any) are combined into one single vector. Non-contextualized word embeddings due to their ability to be used in resource and memory capacity limited settings are used in many NLP tasks today. In recent years, especially in the biomedical domain, context-dependent embedding algorithms [154]–[156] have emerged as superior to the aforementioned non-contextualized ones. These algorithms generate multiple vector representations for the same word, based on the context in which the word is used. In the two evaluations of our proposed ML/DM pipeline [39], [40] (presented in Chapters 3 and 4) we dealt with short text descriptions of recipes – multi-word strings that do not represent a complete sentence, since looking at them from a semantic point of view, they do not have the complete necessary structure to form a sentence, i.e. a subject, a verb, and an object. In this study we are dealing, again, with multi-word strings, or in many cases even one-word strings, such chunks of text cannot be treated as sentences when generating vector representations, therefore using context-dependent embedding algorithms is not of use here, so we opted for non-contextualized embeddings, which yield only one vector for each word.

5.3 Methodology

5.3.1 Domain-specific embeddings

In Figure 5.2, a flowchart of the methodology is presented.

Obtaining the domain-specific embeddings for the list of recipe ingredients is a two-step process (described in Algorithms 5.1 and 5.2):

1. Generating single ingredient embeddings – obtaining multi-word embeddings for each ingredient from the ingredient list of a recipe. If we have:

$$recipe_k \in \{recipe_1, ..., recipe_i\} \tag{5.1}$$

as the recipe we are generating a domain-specific embedding for, where $k \in \{1, l\}$, and l is the number of recipes in the dataset, then:

$$ingredients_k = \{I_1, ..., I_m\} \tag{5.2}$$

as the list of ingredients for $recipe_k$. And we have I_i as the i^{th} ingredient of the list of ingredients:

$$I_i = \{word_1, word_a, .., word_n\} \tag{5.3}$$

Then the single ingredient embedding is obtained in two ways:

- Utilizing sum and average as heuristics for merging the vector representations for each word in the ingredient obtained with word embedding algorithms.

$$E[word_a] = [x_{a1}, x_{a2}, ...x_{ad}] \tag{5.4}$$

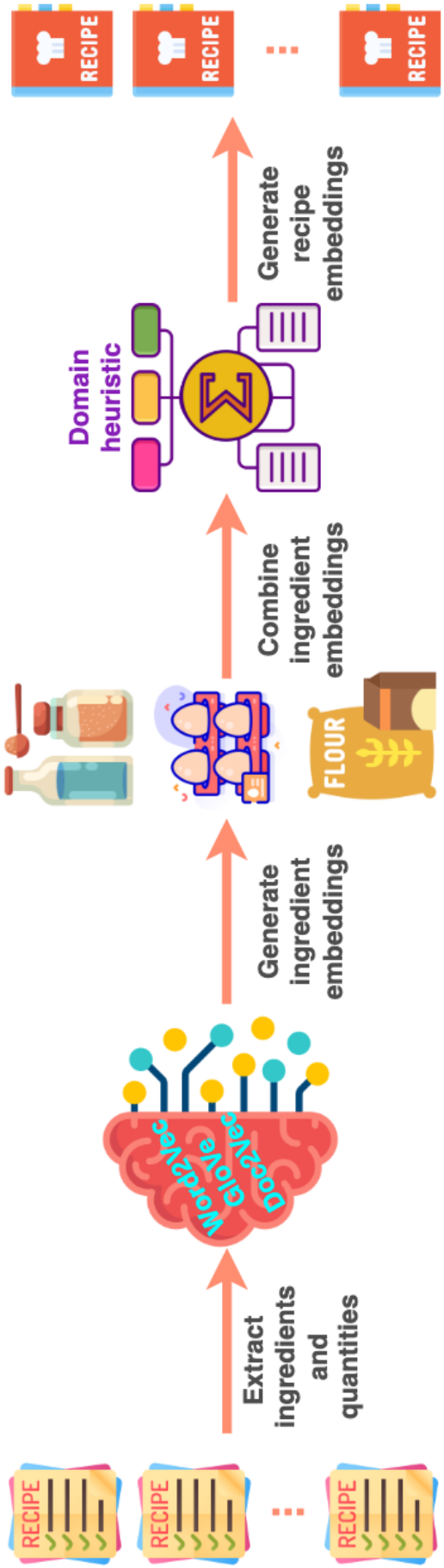

Figure 5.2: Flowchart of the presented approach.

Algorithm 5.1: RL part of the methodology.

Data: Dataset of n recipes with their ingredient lists
Result: Datasets of embedding vectors for the ingredients in the ingredient lists of all n recipes

for *each recipe* r_k *with list of ingredients* $ingredients_k$ **do**
  for *each ingredient* $I_i \in ingredients_k$ **do**
    for *each token* $word_a$ *of* I_i **do**
      Generate $E_{Word2Vec}[word_a] = [x_{a1}, x_{a2}, ..., x_{ad}]$;
      Generate $E_{GloVe}[word_a] = [x_{a1}, x_{a2}, ..., x_{ad}]$;
    end

    for *each dimension of the embedding vectors* $d \in [50, 100, 200]$ **do**
      for *each value of the sliding window parameter* $s \in [2, 3, 5, 10]$ **do**
        for *each metric for merging the word embeddings* $m \in [sum, avg]$ **do**
          for *each Word2Vec architecture* $a \in [SG, CBOW]$ **do**
            Generate $Word2vec$ embedding of I_i:
            if $m == sum$ **then**
              $E_{Word2Vec_{a-m-s-d}}[I_i] = [\frac{x_{11}+...+xn1}{n}, ..., \frac{x_{ad}+...+x_{nd}}{n}]$;
            else
              $E_{Word2Vec_{a-m-s-d}}[I_i] = [x_{11} + ... + x_{n1}, ..., x_{ad} + ... + x_{nd}]$;
            end
          end
          Generate $GloVe$ embedding of I_i:
          if $m == sum$ **then**
            $E_{GloVe_{m-s-d}}[I_i] = [\frac{x_{11}+...+xn1}{n}, ..., \frac{x_{ad}+...+x_{nd}}{n}]$;
          else
            $E_{GloVe_{m-s-d}}[I_i] = [x_{11} + ... + x_{n1}, ..., x_{ad} + ... + x_{nd}]$;
          end
        end
        for *each Doc2Vec*
        $a \in [PV{-}DM_{sum}, PV{-}DM_{avg}, PV{-}DBOW_{sum}, PV{-}DBOW_{avg}]$
        do
          $E_{Doc2Vec_{a-m-s-d}}[I_i] = [x_1, ..., x_d]$
        end
      end
    end
  end
end

Algorithm 5.2: Representation learning part of the methodology.

Data: Datasets of embedding vectors for the ingredients in the ingredient lists of all n recipes
Result: 120 datasets of embedding vectors for the n recipes

for *each recipe* r_k *with list of ingredients* $ingredients_k$ **do**
  Get weights $weights_k$ for $recipe_k$ and normalize them $weights_{100g_k}$;
  Initialize $E_{Word2Vec_{a-m-s-d}}[recipe_k] = 0$;
  Initialize $E_{GloVe_{m-s-d}}[recipe_k] = 0$;
  Initialize $E_{Doc2Vec_{a-m-s-d}}[recipe_k] = 0$;
  for *each ingredient* $I_i \in ingredients_k$ **do**
    for *each dimension of the embedding vectors* $d \in [50, 100, 200]$ **do**
      for *each value of the sliding window parameter* $s \in [2, 3, 5, 10]$ **do**
        for *each metric for merging the word embeddings* $m \in [sum, avg]$ **do**
          for *each Word2Vec architecture* $a \in [SG, CBOW]$ **do**
            $E_{Word2Vec_{a-m-s-d}}[recipe_k] += w_{100g_i} \times E_{Word2Vec_{a-m-s-d}}[I_i]$
          end
          $E_{GloVe_{m-s-d}}[recipe_k] += w_{100g_i} \times E_{GloVe_{m-s-d}}[I_i]$
        end
        for *each Doc2Vec architecture* $a \in [PV-DM_{sum}, PV-DM_{avg}, PV-DBOW_{sum}, PV-DBOW_{avg}]$ **do**
          $E_{Doc2Vec_{a-m-s-d}}[recipe_k] += w_{100g_i} \times E_{Doc2Vec_{a-m-s-d}}[I_i]$
        end
      end
    end
  end
end

Where $E[word]$ is the vector representation (embedding) of a separate word, $a \in \{1, ..., n\}$, and d is the dimension of the word vectors. Then the vector representation of the ingredient will be obtained as:

$$E[I_i] = [\frac{x_{11} + ... + x_{n1}}{n}, \frac{x_{12} + ... + x_{n2}}{n}, ..., \frac{x_{1d} + ... + x_{nd}}{n}] \tag{5.5}$$

an average from the vectors of the words from which it consists of, or:

$$E[I_i] = [x_{11} + ... + x_{n1}, x_{12} + ... + x_{n2}, ..., x_{1d} + ... + x_{nd}] \tag{5.6}$$

by summing the vector representations of the words it consists of.

- Utilizing the vector representations for the full multi-word strings considered as paragraphs, obtained with a paragraph embedding algorithm. The same applies as in Equations (5.1), (5.2), and (5.3), then:

$$E[I_i] = [x_{11} + ... + x_{n1}, x_{12} + ... + x_{n2}, ..., x_{1d} + ... + x_{nd}] \tag{5.7}$$

$E[I_i]$ is the multi-word string vector representation for the ingredient I_i. Where d is the predefined dimension of the vectors.

2. Embeddings on recipe level – Here we define a domain-specific heuristic for merging non-contextualized multi-word embeddings. If the same applies as in Equation 5.1 and Equation 5.2, and:

$$weights_k = [w_1, w_2, ..., w_m] \tag{5.8}$$

is the list of the weights expressed in grams for each ingredient for $recipe_k$, calculated on the whole recipe. On the grounds that the nutrient values of a food are by rule given per 100 grams, and the fact that we are making the predictions for the nutrients on 100 grams, the weights for each ingredient on 100 grams of the recipe are calculated:

$$weights_{100g_k} = [w_{100g_1}, w_{100g_2}, ..., w_{100g_m}] \tag{5.9}$$

Then in order to obtain the domain-specific embedding for the whole recipe, we implement the following heuristic for combining the single ingredient embeddings:

$$E_{recipe_k} = w_{100g_1} \times E[I_1] + ... + w_{100g_m} \times E[I_m] \tag{5.10}$$

5.4 Results and Discussion

5.4.1 Data

In the experiments for the evaluation of the ML/DM pipeline in [39] (presented in Chapter 3) we used food consumption data containing nutritional information about food items. For exploring the bias of the domain knowledge over the prediction task, in [40] (presented in Chapter 4) we evaluated the extended ML/DM pipeline on the Recipe1M dataset, available online [150], which is a large-scale structured corpus of over one million cooking recipes and 13 million food images. For the evaluation in this study, we used the Recipe1M dataset, as well, and out of those one million recipes in our focus are the ones that contain details like nutrient content – a total of 51,235 recipes. For each of the 51,235 recipes, the following information is available:

- Recipe title – a short textual description of the recipe;
- Ingredients – list of ingredients;

- Recipe instruction – long textual description of step-by-step instructions for preparing the recipe;
- Nutrient content of ingredients – quantity in grams of fat, protein, saturates, sodium, and sugar per whole weight of the ingredient for each ingredient;
- Quantity of each ingredient;
- Units of measurement per each ingredient – according to the household measurement system (cup, tablespoon, teaspoon, etc.);
- Weight in grams per each ingredient for the whole recipe;
- Nutrient content – quantity in grams of fat, protein, salt, saturates, and sugars per 100 grams for the whole recipe;

In Table 5.1, we give an example data instance from the dataset – a recipe with it's title, list of ingredients, and nutrient values per 100 grams of the recipe. For the same recipe in Table 5.2 the nutrient values per total weight of the ingredients in the recipe.

Table 5.1: Example instance from the Recipe1M dataset.

Recipe title	**No Bake Oatmeal Cookies**					
Ingredients	"sugars, granulated", "cocoa, dry powder, unsweetened", "milk, fluid, 1% fat, without added vitamin a and vitamin d", "butter, without salt", "vanilla extract", "peanut butter, smooth style, without salt", "oats"					
Nutrients per 100 grams	**Energy**	**Fat**	**Protein**	**Salt**	**Saturated fat**	**Sugars**
	378.64	35.40	3.81	0.06	21.01	8.59

5.4.2 Data pre-processing

The Recipe1M dataset contains detailed information about each recipe. In this study, the data of interest are: the list of ingredients, the weights of each ingredient, and the quantities per 100 grams of the recipe for the five nutrients of concern. Before generating the single ingredient embeddings, for each ingredient the text undergoes some basic NLP pre-processing: tokenization, normalization, noise removal and lemmatization.

5.4.3 Experimental setup

The experimental setup after the data pre-processing is as follows:

1. Generate embeddings on ingredient level – for each ingredient using the Word2Vec, GLoVe and Doc2Vec algorithms [20]–[22], [145] multi-word non-contextualized embeddings are generated. The implementation of the embedding algorithms is with the *Gensim* [145] library in Python – for Word2Vec and Doc2Vec using the appropriate packages, and with the scripts and instructions provided by the authors in [151] for the GloVe algorithm. Same as in Chapter 3 and 4, several runs are executed for the three algorithms with different values for the parameters of each. The selected vector dimensions are 50, 100 and 200, because, as mentioned in Chapters 3 and 4, they are among the frequently used dimensions for textual embeddings for these algorithms, and are efficient in means of execution time and computational power. For the sliding window (the number that indicates the maximum distance between

Table 5.2: nutrient values per total weight of the ingredients in the recipe given in Table 5.1.

Ingredient	**Nutrient values**					
	Energy	**Fat**	**Protein**	**Saturated fat**	**Sugars**	**Sodium**
sugars, granulated	1536.00	0.00	0.00	0.00	402.240	0.00
butter, without salt	19584.00	2211.84	23.04	1376.44	1.92	384.0
cocoa, dry powder, unsweetened	16.33	0.98	1.40	0.57	0.13	1.50
milk, fluid, 1% fat, without added vitamin a and vitamin d	1224.00	28.44	98.64	18.54	152.28	1284.00
peanut butter, smooth style, without salt	1528.00	131.52	56.88	26.43	26.88	40.00
vanilla extract	12.00	0.00	0.00	0.00	0.53	0.00
oats	1821.00	32.28	79.00	5.69	0.00	9.00

the current and predicted word within a sentence) the values chosen are 2, 3, 5, and 10, as we are, again, dealing with short textual descriptions that do not form a sentence and the maximum number of word per ingredient is 23. For the Word2Vec and the Doc2Vec algorithms the two existing architecture are considered, CBOW and SG for Word2Vec, PV-DM and PV-DBOW for Doc2Vec. Additionally, we train different models using the two methods for merging the word embeddings (sum and average) for the two word embedding algorithms, Word2Vec and GloVe. Regarding the Doc2Vec algorithm, the non-concatenative mode is employed (separate models for the sum and the average), as the model would be substantially larger if using concatenated vectors in place of sum/average. As a result, 48 models are trained using the Word2Vec and Doc2Vec algorithms, while 24 models are trained using the GloVe algorithm.

2. Generate embeddings on recipe level – use the above-defined domain heuristic (Equation 5.10) to merge the embedding on single ingredient level.

3. Apply single-target regression algorithms – building models for predicting the five given nutrients: fat, protein, salt, saturates, and sugars. First, we select the regression algorithms (Linear regression, Ridge regression, Lasso regression, and ElasticNet regression [146], all described in Chapter 2, Section 2.3.1). Then, hyper-parameter tuning is performed in order to select the best parameters for the model training; after which the values are generated with k-fold cross-validation. For comparison of the regressors the matched sample approach is chosen, i.e., using the same data in

each iteration.

4. Calculate domain-specific measure of accuracy – we define the accuracy according to the appropriate tolerance levels for each nutrient, which are defined by international legalizations and regulations. The European Commission Health and Consumers Directorate General in 2012 published a guidance document [46], with the aim to provide advised recommendations for calculation of the acceptable differences between quantities of nutrients on the label declarations of food products and the ones established in Regulation EU 1169/2011 [47]. These tolerances for the food product labels are important as it is impossible for foods to contain the exact levels of nutrients that are presented on the labels, as a consequence of the natural variations of foods, as well as the variations occurring during production and the storage process. The accuracy is calculated following the same process described in [39], [40].

5.4.4 Evaluation outcomes

In order to evaluate the performance of our domain-specific merging heuristic, we repeat the same experiments twice:

1. Merge the single-ingredient embeddings with the domain-specific heuristic, and perform the predictive modeling, obtain the results, and calculate the defined accuracy.

2. Merge the single-ingredient embedding with conventional merging heuristics – sum and average, according to the merging heuristic used when obtaining the single-ingredient embeddings. In other words:

   - The embedding on recipe level is obtained by calculating an average of the embeddings on single-ingredient level when they are obtained using Equation 5.4.
   - The embedding on recipe level is obtained by summing the single-ingredient embeddings when they are obtained using Equation 5.5.

   After obtaining the embedding on recipe level, the same steps are applied – performing the single-target regressions, obtaining the predictions and calculating the defined accuracy, of course, using the same experimental setup described earlier.

The results from the evaluation show that using the domain-specific heuristic yields higher accuracy percentages than the conventional merging techniques. In Table 5.3, the results from the evaluation are presented. For presentation purposes, for each embedding algorithm we included only the maximal and minimal accuracies achieved for each nutrient (without the details of vector dimension, sliding window, and regression algorithm), and we also calculated the mean accuracy – calculated for each nutrient from all the accuracy percentages for the certain embedding algorithm (all possible vector dimensions, sliding windows, and regression algorithms). From the results it is evident that the ML models that use the embeddings merged with the domain heuristic as input features outperform the models which use the embeddings merged with conventional merging heuristics as features. The differences between the two approaches in the accuracies are rather big. We can note that the prediction accuracies for salt content are rather lower than the other nutrients, but this is due to the fact that the salt content in 100 grams from the food/recipe are rather low compared to the other nutrients (most often less than 1 gram). It is also apparent that the prediction results for protein values are the best out of all the nutrients, and that the Doc2Vec embedding algorithm outperforms the Word2Vec and GloVe algorithms.

Table 5.3: Results from the evaluation on Recipe1M.

Measure		Algorithm					
		With domain heuristic			Without domain heuristic		
		Word2Vec	GloVe	Doc2Vec	Word2Vec	GloVe	Doc2Vec
Maximal accuracy (in %)	Fat	76.66	75.62	91.65	21.45	22.16	20.29
	Proteins	90.57	88.79	97.98	55.47	54.54	52.67
	Sugars	73.38	76.35	88.14	25.73	25.81	22.97
	Saturates	72.78	73.66	95.95	24.00	24.71	20.58
	Salt	43.34	41.79	52.35	36.28	33.10	19.43
Minimal accuracy (in %)	Fat	18.65	18.65	28.00	8.34	8.34	8.35
	Proteins	56.60	56.60	56.57	27.20	27.20	27.22
	Sugars	17.03	17.03	28.54	7.90	7.90	7.90
	Saturates	30.52	30.52	45.27	8.34	8.34	8.35
	Salt	19.24	19.24	45.27	9.44	9.44	9.44
Mean accuracy (in %)	Fat	37.96	48.03	68.99	12.54	13.22	14.70
	Proteins	68.02	78.99	86.12	37.19	36.85	39.54
	Sugars	38.58	47.13	56.05	13.61	14.03	15.07
	Saturates	60.10	67.37	78.32	12.84	13.18	14.11
	Salt	26.25	27.49	31.01	16.07	15.65	18.37

For further interpretation of the results, we used the Principal Component Analysis (PCA) [157] as a reduction technique to visualize some of the obtained embeddings. With closer inspection of the results, we observed that the maximal accuracies in most cases were obtained with the Doc2Vec algorithm when using the following parameters:

- Vector dimension: 100,
- Sliding window: 2,
- Architecture: PV-DBOW,
- Merging technique: sum.

All of the visualizations presented are using the embeddings obtained from the Doc2Vec algorithm and these parameters. First, we visualized the embeddings obtained with the domain-specific heuristic for 20 different recipes, with the same list of ingredients but same or different quantities. The visualization is presented in Figure 5.3. From the visualization we can see how they group in the embedding space – there is a big chunk of recipes in the middle and then 5 of them very far from each other. To understand why this is happening we searched for the nutrient values of these recipes, which are given in Table 5.4. We can see that the reduced embeddings for the recipe with title "The Best No Bake Cookies" and the recipe with title "No Bake Oatmeal Cookies" are overlapping, and from the table we can see that they have almost identical nutrient values. Next, we can see that the reduced embedding for the recipe with title "Laura's House Famous Mud Pie" is far apart from the rest 19, and from the table we can see that it is because its sugar content is significantly higher than the rest. Then the reduced embeddings for the recipes with titles "No Bake Cookies" and "No-Bake Cookies" are placed far apart although they have almost identical titles (except the hyphen), however from the table we can see that this is logical because of their different nutrient values in fat and sugars.

The same visualization is made for the embeddings on recipe-level obtained with the conventional merging heuristics (presented in Figure 5.4). From the figure we can see a very different placement of the embeddings in the space. For comparison purposes, only the names of the previously analyzed recipe embeddings are included. We can see that the embeddings for the recipes with titles "The Best No Bake Cookies" and "No Bake Oatmeal Cookies" are very far apart even though they have almost identical nutrient values, while the embeddings for the "No Bake Oatmeal Cookies" recipe and "Chocolate No-Bake Cookies" are overlapping, even though we can clearly see from their nutrient content (presented in Table 5.4) that they have very different nutrient values – the difference in each nutrient is very considerable (fat difference is 2 times, protein difference is 3 times, saturates difference is almost 3 times, and sugar difference is more than 4 times). The next thing we can notice is how the recipes "Laura's House Famous Mud Pie" and "The Best No Bake Cookies" are placed very close together, when, judging from their nutrient content (presented in Table 5.4), they have very different nutrient values (the first recipe has 2 times less fat than the second, more than 6 times the amount of sugar, and almost 3 times less saturates). This just goes to show how important is the chosen heuristic when merging word embeddings. Even though all these recipes have the same identical ingredient list, they differ significantly in nutrient content. This depicts the differences in the feature space, to capture this difference in the performance space, we present the results from the predictions for the same recipes in Table 5.5. From these results, we can see that the difference between the actual values of the nutrients and the predicted values of the nutrients when using the domain heuristic is smaller than the difference between the actual values of the nutrients

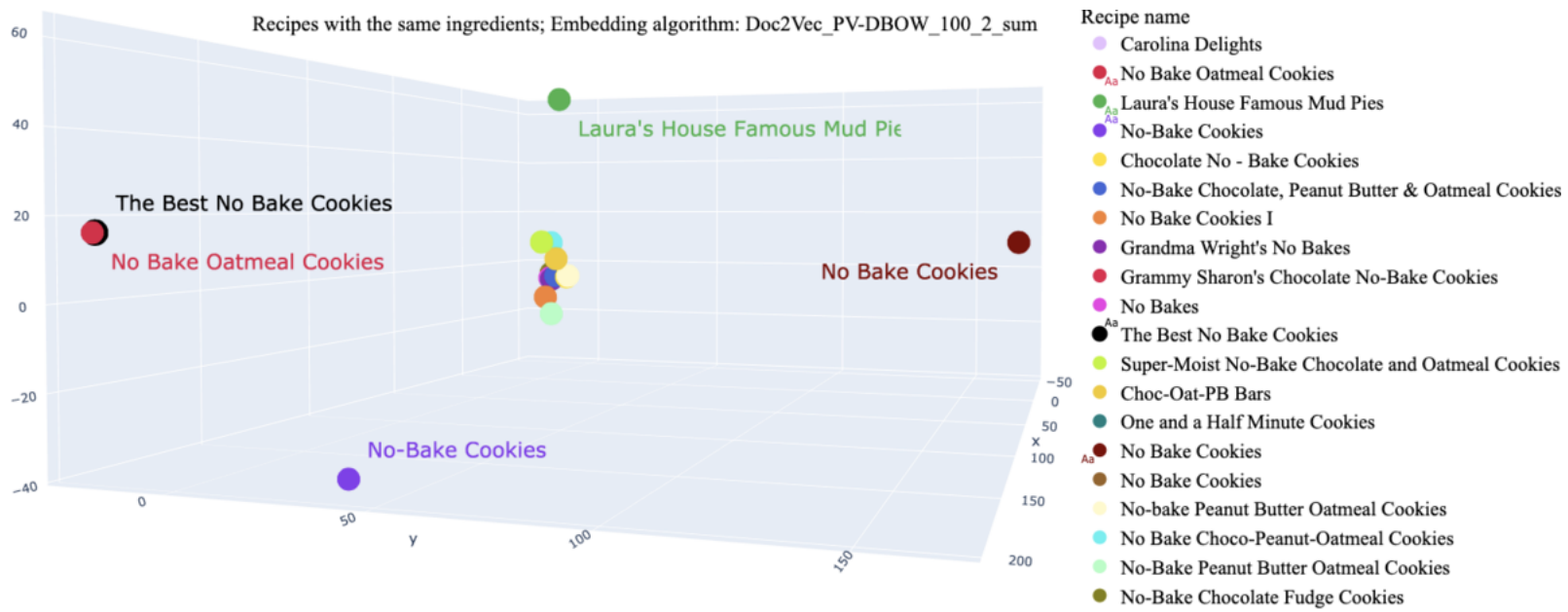

Figure 5.3: Visual representation of the vector representations when using the domain-specific heuristic of a group of recipes with the same ingredients but different nutritional values (The nutritional values of each recipe represented in this picture are given in Table 5.4).

Table 5.4: Nutrient values for recipes with the same ingredient list, but same or different quantities.

Recipe title	Ingredients	Nutrients per 100 grams					
		Energy	**Fat**	**Protein**	**Salt**	**Saturates**	**Sugars**
The Best No Bake Cookies	"cocoa, dry powder, unsweetened","milk, fluid, 1% fat, without added vitamin a and vitamin d","oats","peanut butter, smooth style, without salt","sugars, granulated","butter, without salt","vanilla extract"	378.64	35.40	3.81	0.06	21.01	8.59
No Bake Oatmeal Cookies		378.44	35.37	3.83	0.06	21.00	8.58
Laura's House Famous Mud Pies		385.10	15.58	4.45	0.02	8.05	50.80
No Bake Cookies		323.39	15.26	14.74	0.03	6.41	19.38
No-Bake Cookies		317.04	22.93	13.34	0.06	5.53	11.53
Chocolate No–Bake Cookies		397.65	15.39	9.43	0.02	6.28	33.02

and the predicted values of the nutrients without using the domain heuristic.

From the list of 20 recipes in these visualizations, we can see that there are a lot of recipes with very similar names, if not with the same name. Given this observation, we dug deeper, and gathered a list of recipes in the Recipe1M dataset that have the same identical name – "No Bake Cookies", there were some recipes that included punctuation sign(s), but we omitted those. There were 7 recipes in total with the name "No Bake Cookies". We did the same two visualizations for the embeddings on recipe-level obtained with the domain-specific heuristic (Figure 5.5) and with the conventional merging heuristics (Figure 5.6). For each recipe we included the five nutrient values of concern next to the reduced embedding point. We can see that when using the domain heuristic, the embeddings placed close together are for recipes that have very similar nutrient values for the five nutrients (i.e., which further help the prediction task), while when using the conventional merging heuristic, the embeddings placed close together (i.e., making the prediction task difficult), or in this case overlapping (the black and grey) have very different nutrient content (the recipe represented with the grey marker has almost double the amount of sugar). This not only proves that the domain-specific merging heuristics is a better approach when predicting nutrient values, but also how limited is the information included in these short recipe descriptions – having the same name, or even the same ingredient list, does not by any means state that two recipes have the same or even comparable nutrient content.

5.4.5 Discussion

The results from this study, following our two previous related studies [39], [40] indicate the impact of integrating domain-driven knowledge into a ML pipeline using a nowadays common NLP tool – word embeddings. When put-into-effect our task-specific embedding merging heuristic yields high accuracy results for a domain sensitive assignment such as – nutrient prediction. We must point out that the Recipe1M dataset is very thorough, and can be regarded as an "outlier" dataset, since many datasets of such kind, i.e., recipe datasets, do not contain such exhaustive information. Particularly talking about the "weight per ingredient in grams" data, which in our case is crucial. The most typical data that can be found crosswise recipe datasets would be just a list of ingredients. The list of ingredients usually includes data about the quantity of each ingredient, the unit in which said ingredient is measured and the ingredient itself, which is commonly combined, or if we put it into data terms – is a coherent string. An important fact that should be noted here is – we are talking about recipe data, that is data that is usually retrieved from the Web; thus, these units are most certainly expressed in food/cooking household measurements. Therefore, in order to utilize this data, and draw the information that is required, a few NLP techniques must be put into work. First, named entity recognition (NER) – to segment the strings into quantity and unit, which means we need rules of what represents a quantity, a resource with all the possible units, i.e., the common food household measurements, and lastly a resource to identify the ingredients/food item [158], [159]. Then, the quantities need to be normalized – after the NER, the units (household measurements and/or others) need to be converted in grams, in order to have the same unit all across the dataset. This can be done using conversion tables [160].

Even more, we need to map the extracted unit to the proper unit in the conversion table, which can be viewed as simple string matching, but since there are multiple ways of writing a single household measure unit, it can be viewed as a slightly more complex task than string matching – for example: mapping strings based on lexical similarity [161]. Another option is to put these conversions in a dictionary and create a conversion dictionary, which

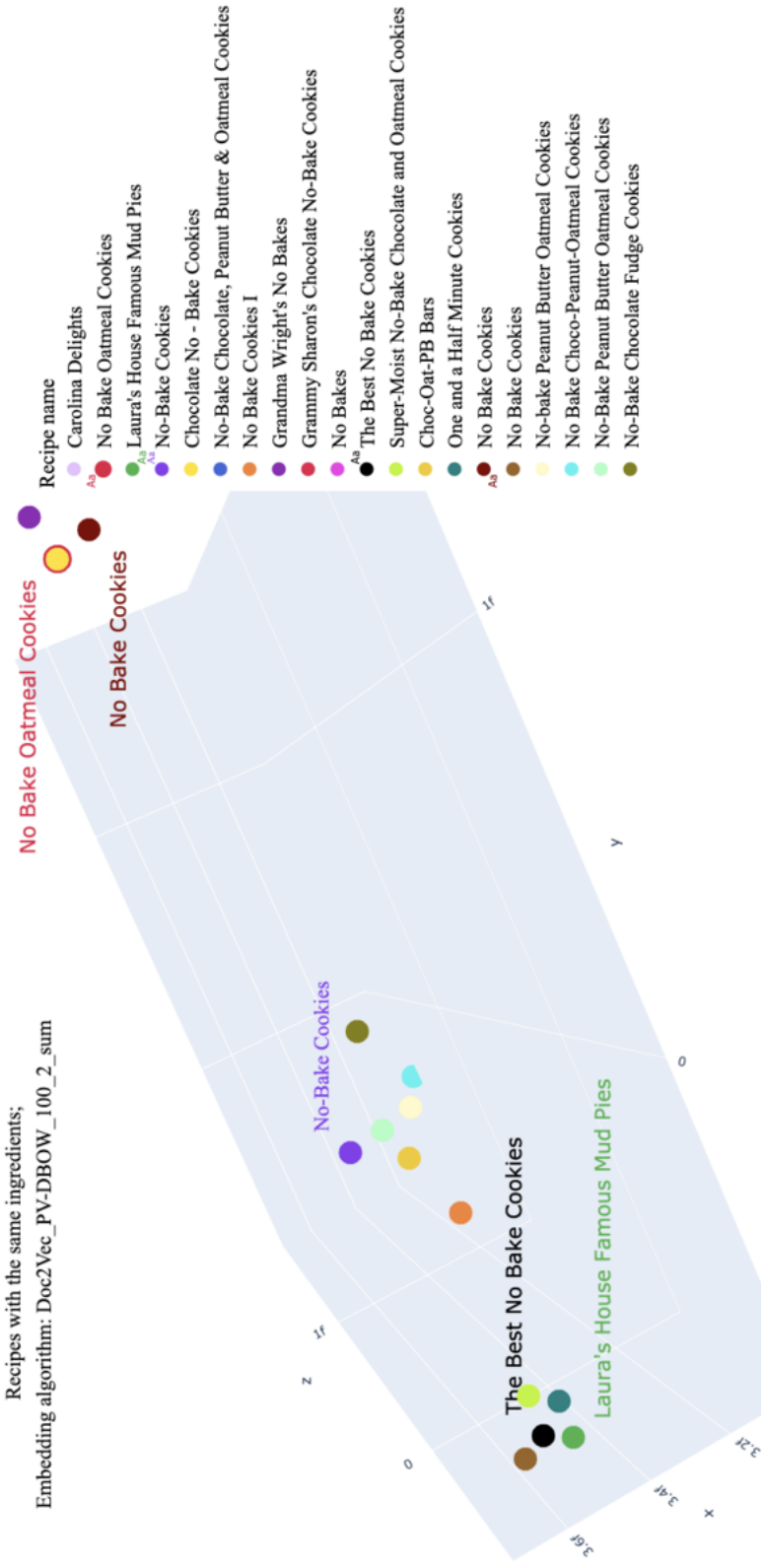

Figure 5.4: Visual representation of the vector representations without the domain-specific heuristic of a group of recipes with the same ingredients.

Table 5.5: Differences in performance space (A – Actual value, DH – Predicted value when using the domain heuristic, No DH – Predicted values without using the domain heuristic).

Recipe title			The Best No Bake Cookies	No Bake Oatmeal Cookies	Laura's House Famous Mud Pies	No Bake Cookies	No-Bake Cookies	Chocolate No–Bake Cookies
Nutrients per 100 grams	**Fat**	**A**	35.4	35.37	15.58	15.26	22.93	15.39
		DH	32.28	32.92	14.38	15.32	25.54	14.47
		No DH	23.11	12.44	29.33	16.78	16.78	8.79
	Protein	**A**	3.81	3.83	4.45	14.74	13.34	9.43
		DH	3.59	4.38	4.40	12.29	9.12	7.26
		No DH	8.96	19.67	2.34	12.11	7.34	15.67
	Salt	**A**	0.06	0.06	0.02	0.03	0.06	0.02
		DH	0.03	0.03	0.15	0.06	0.30	0.39
		No DH	1.78	0.30	0.20	0.28	0.20	0.38
	Saturates	**A**	21.01	21.00	8.05	6.41	5.53	6.28
		DH	18.21	18.23	7.21	6.11	6.79	5.74
		No DH	10.53	10.43	12.56	2.56	11.67	15.89
	Sugars	**A**	8.59	8.58	50.80	19.38	11.53	33.02
		DH	10.88	10.88	50.62	22.36	11.30	34.46
		No DH	27.45	18.99	24.75	2.33	21.74	27.85

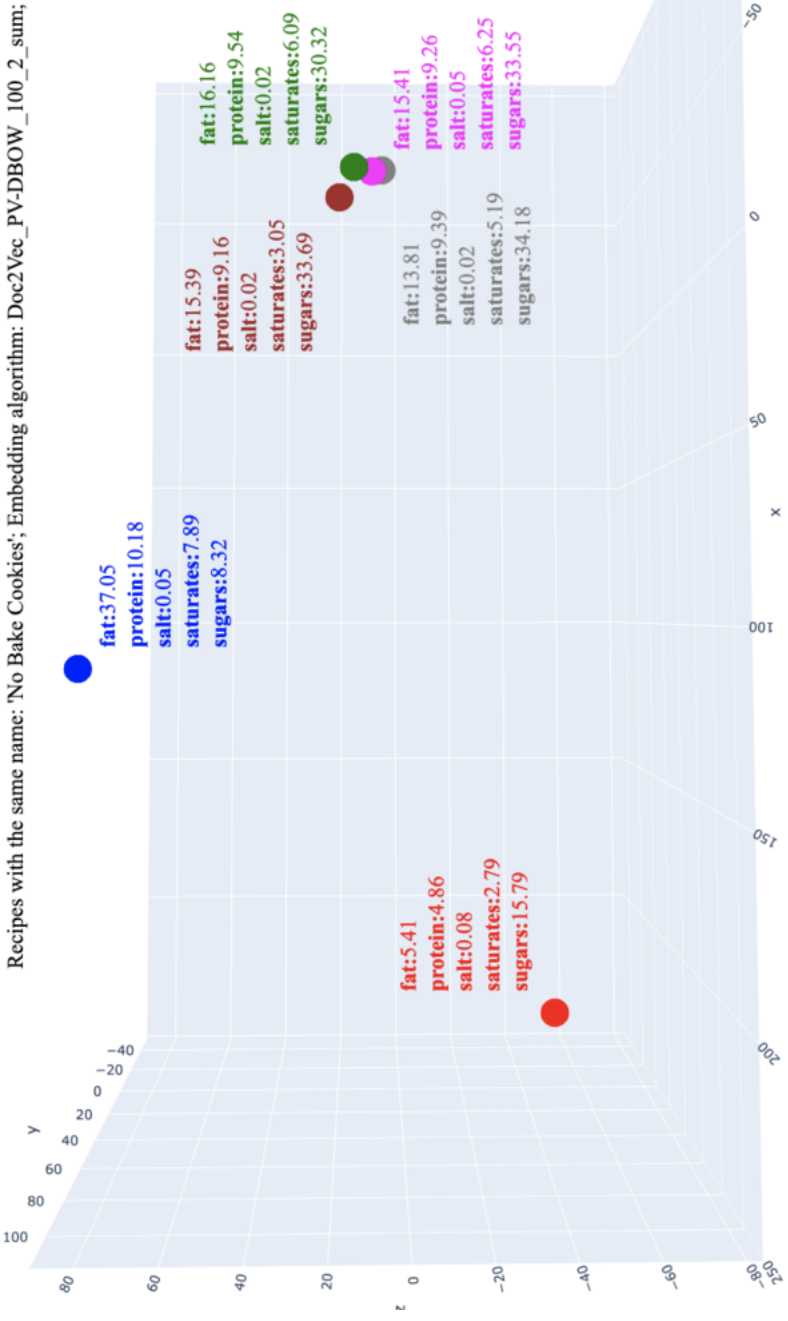

Figure 5.5: Visual representation of the vector representations with the domain-specific heuristic of a group of recipes with the same name.

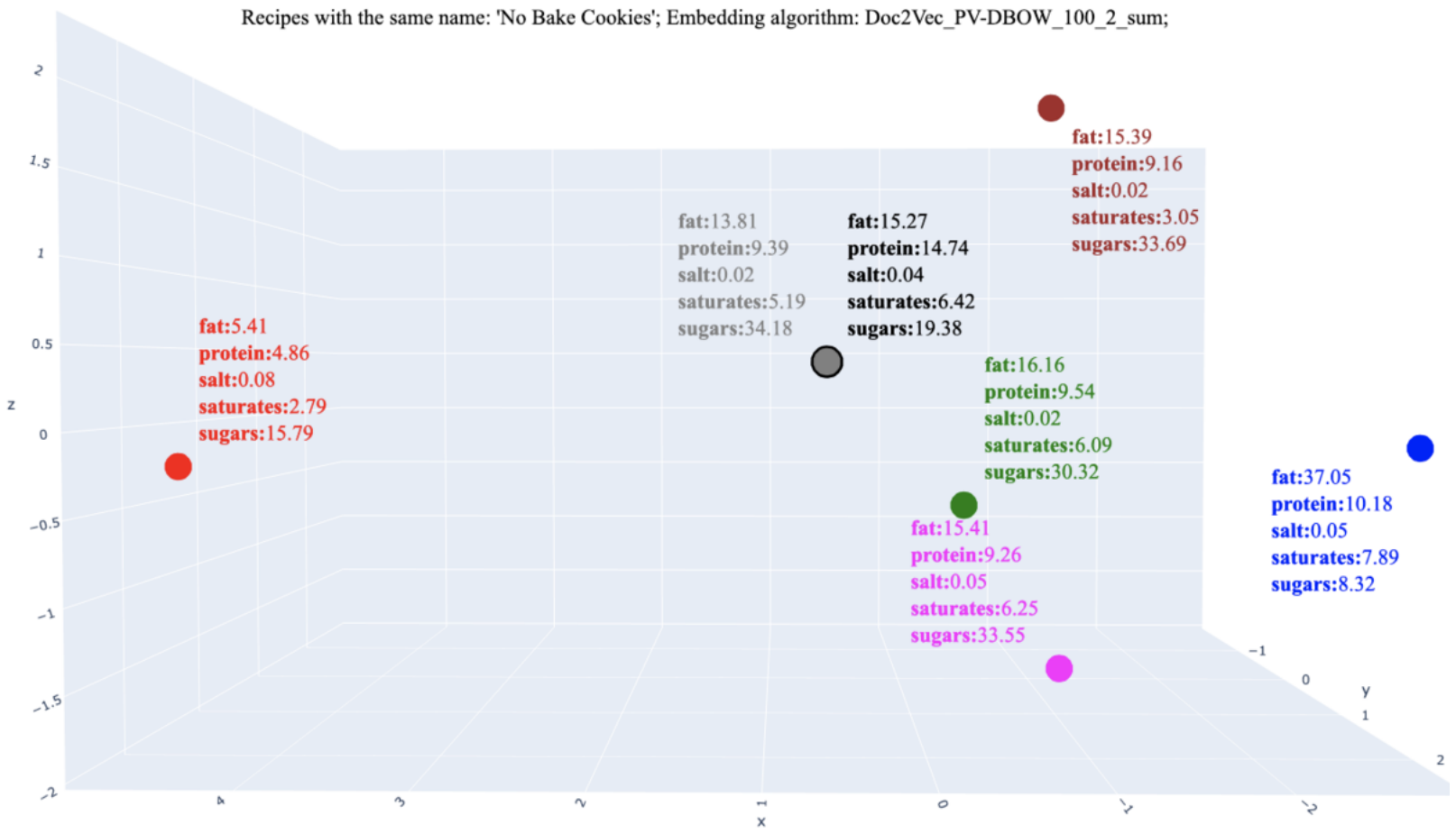

Figure 5.6: Visual representation of the vector representations without the domain-specific heuristic of a group of recipes with the same name.

should still be constructed and approved by a domain expert. Needless to say, here, the lack of domain semantic resources in the Food and Nutrition domain comes to the fore.
Another very important note to be mentioned is that recipe dataset more often than not, does not include nutrient values for each ingredient, or for the recipe itself as a whole. As a matter of fact it is very rare for a recipe dataset to include them. Therefore, to obtain the nutrient values for each ingredient when the recipe nutrient values are not available, we should resort to mapping the recipe data (the ingredients) to a FCDB. This can be viewed as a lexical [161] or semantic similarity problem [162], [163].

Chapter 6

Predefined Domain-Specific Embeddings of Food Concepts and Recipes: A Case Study on Heterogeneous Recipe Datasets

In this chapter, we present a case study – applying the RL pipeline presented in Chapter 5 on heterogeneous recipe datasets and generating two corpora of predefined embeddings. We start with the problem definition, followed by additional related work specific to this study. Next, we go into detail about the methodology, especially the data normalization techniques used – to extract the information needed, and the data mapping process. Following the description of the methodology we present the evaluation results from applying the ML/DM pipeline on six heterogeneous recipe datasets, to which we will refer as: Indian recipes, Recipe1M, Epicurious, Salad recipes, Yumlly28K, Recipe box. At the end we conclude with a discussion of the results.

6.1 Problem Definition

In this era of social media, internet content containing recipe data is in abundance. However, well-structured datasets containing recipe data, that can be used for research purposes or reused for some application scenario that requires structure, are very rare, almost non-existent. Recipe datasets are most often than not collected and extracted from food-focused online social networking services, websites and mobile apps that provide recipes to users, which are written from other such users. There are plenty of such services, to name a few – Allrecipes [164], Yummly [165], Epicurious [166]. These recipes are usually written with little to no structure, using both standardized and non-standardized units of measurement. Recipe1M is the only publicly available recipe dataset that has the necessary data to apply the presented ML/DM pipeline. The goal of this study is to generate predefined embeddings for recipe data learned using heterogeneous, multi-lingual datasets.

Text embeddings have gained a lot of traction and are an essential and standard component of many NLP studies. The principal use has been "transfer learning", where using very big datasets of raw, unlabeled data (from different sources, e.g. web scrapping) we first learn embeddings, and then use these pre-trained i.e. predefined embeddings as inputs of a model in a supervised task (shown in the flowchart of a classical ML pipeline in Figure 1.1).

In [167], the authors demonstrate the ability to use transfer learning for text data. Transfer learning has been shown to perform exceptionally well for imaging tasks [168], [169] because of pre-trained computer vision models [170]–[172], all of which are pre-trained on the ImageNet database [173].

In the means of domains, work like this has been done in the medical domain, for example cui2vec [174] – which are a predefined set of embeddings for 108,477 medical concepts. In [175] the authors present Med2Vec – a simple, robust algorithm that learns code and visit representations by using huge EHR datasets with over a million visits. It enables us to interpret the representations that have been positively validated by clinical specialists. In the Food and Nutrition domain, however, there is a lack of this kind of studies. Existing work that focuses on textual content to learn recipe representations is presented in [29], [30]. They take advantage of the instructions and ingredients associated with recipes. Several other studies consider the images [31]–[34] and they typically concentrate on the recipe-image retrieval task and attempt to align images and text together in a shared space, resulting in information loss for both modalities. In [176], the authors focus on learning recipe representations using multi-modal information extracted from images, text, and relations, focusing on cuisine category classification and region prediction, based on the recipe categories, meaning the textual data included is only through the recipe category.

The task of predicting nutrient values in this context still remains unexplored. Moreover, recipe embedding based on the quantities of the ingredients has never been introduced prior to our work in [41]. In this study, we collect recipe data from six heterogeneous recipe datasets: Indian recipes, Recipe1M, Epicurious, Salad recipes, Yumlly28K, Recipe box. The datasets are first brought to the same format, through the processes of named entity recognition (NER) and data mapping. We use dictionary-based NER for extracting the measurement units and rule-based NER for extracting the quantities of the ingredients. The data mapping process is an ensemble of multiple approaches, which we use for mapping the ingredients to a Food Composition Database (FCDB) from the United States Department of Agriculture (USDA) [177] called FoodData Central USDA to obtain the nutrient values.

The end product from this study are two predefined embedding corpora – for ingredient vector representations and recipe vector representations, as well as four domain dictionaries constructed with and approved by a domain expert: a dictionary for units of measurement, for converting units of measurement, for names of branded foods, and a dictionary for redundant words specific to recipe data. Training embeddings tailored for a specific task is a very time consuming process, therefore the corpora of predefined embeddings can be used for research purposes as well as for application purposes transferring them to other tasks.

6.2 Related Work

In this section we present a additional related work to the one presented in Chapter 2 that is more tightly connected to this study in particular.

6.2.1 USDA Food Composition Database

FoodData Central is the The Food Composition Database (FCDB) from the United States Department of Agriculture (USDA) and it is a comprehensive, research-focused data system that offers links to sources of information about agriculture, food, dietary supplements, and other topics in addition to increased data on nutrients and other food components. Researchers, policy makers, academics and educators, nutrition and health professionals, product creators, and others can all use FoodData Central to their advantage. Over time, both the food supply and scientific knowledge of the connections between dietary intake and health have changed. In order to fulfill the demands of a wide range of users, including researchers, policymakers, nutrition and health experts, and product manufacturers, the USDA's food composition data tools are also evolving. The need for open and easily accessible information about the nutrients and other ingredients in foods and food products has significantly increased in recent years due to the rapidly expanding pace of change in the food supply and the growing diversity of uses for food data. In order to analyze, gather, and present dietary profile data in a way that is rigorously scientific, a new methodology is needed. FoodData Central is still the embodiment of this novel strategy. From the FoodData Central USDA FDCB four datasets are of our importance:

1. Foundation Foods – contains values determined from assessments of nutrients on individual samples of commodity/commodity-derived minimally processed foods with insights into variability, as well as significant underlying metadata. The metadata includes quantity of samples, sampling location, collection date, analytical techniques employed, and, if pertinent, agricultural data like genotype and production techniques. The increased breadth and transparency of Foundation Foods' data can shed light on a variety of factors that affect the variability of nutrient and food component profiles, offering useful insights. The long-term objective of Foundation Foods is to increase the variety of staple foods, ingredients, and the research that supports them. The data are based on analytically derived values.

2. Nutrient Database for Dietary Studies (FNDDS) – Data on nutrients and portion weights for foods and beverages reported in What We Eat in America, the dietary intake section of the National Health and Nutrition Examination Survey (NHANES) [178]. Analysis of food intakes reported in NHANES and numerous other nutritional research studies is made easier by FNDDS data.

3. Branded Foods – Data from labels of national and international branded foods collected by a public-private partnership. The data are based on food label information. Through standardized data presentation, USDA supports this data type. Research studies, efforts to regulate food labels, and product development all make use of Branded Foods data in various ways.

4. Standard Reference (SR) Legacy – Historic data on food components including nutrients derived from analyses, calculations, and published literature. The data are based on SR originally available via the USDA National Nutrient Database (NNDB), and for many years, the main food composition data type in the US has been Standard Reference Legacy. It offers a thorough listing of values for food nutrients, including nutrients gleaned via analyses, imputations, and the written record. This data type has supported a wide range of public policy efforts, research investigations, diet planning and education activities, and has provided the values for the majority of other governmental and private food composition databases. The last version of this data type, SR Legacy, is made available in April 2018.

6.2.2 Domain dictionaries

In order to perform extraction of the needed information from the text in the recipe datasets, there is a need of domain dictionaries – specifically, a dictionary with units of measurements (from the International System of Units (SI) and household measurements). Unfortunately, there were no available dictionaries that would be useful for our case, therefore we proceeded with assembling the needed resources with the assistance of a domain expert – a nutritionist.

6.2.2.1 Units of measurement dictionary

This dictionary is constructed with the help of a domain expert – a nutritionist, and it includes possible units of measurement for different food items that can be found in recipe data:

1. Units of measurement for quantity from the International System of Units (SI) – these measurements are official and defined in [179], [180], and are established and maintained by the General Conference on Weights and Measures (CGPM) [181]. Example: grams, milligrams, etc.

2. Units of measurement for quantity from the Household Measurement System – which is not an official system for measuring units, but it is most commonly used in everyday life. Example: tablespoon, teaspoon, cup, etc.

3. Units of measurement for quantity from the Apothecaries' system, which originated as the system of weights and measures for dispensing and prescribing medications, is a historical system of mass and volume units that were used by physicians and apothecaries for medical prescriptions and also sometimes by scientists [182]–[184]. Example: pint, dram, scruple, etc.

The dictionary of Units of Measurement contains 144 instances.

6.2.2.2 Redundant words dictionary

This dictionary includes a list of words and phrases that can be considered as redundant in this scenario. These can be words that do not bring any weight or value to the nutritional content of the ingredient in question, as well as additional explanation that the person that wrote the recipe included in order to further explain either the ingredient itself or the cooking process. This dictionary contains 415 words and phrases in total. In Table 6.1, examples are presented where the redundant word/phrase is shown in bold.

6.2.2.3 Branded foods dictionary

This dictionary includes popular food brand names that can often replace the word for the actual food item they produce. This dictionary is extracted from the branded food dataset from the USDA FCDB. Example of such food items are: "M&Ms", "Cola", "Stevia", "Oreo", "Nescafe", "Dr. Peppers", "7Up", etc.

6.2.2.4 Conversion dictionary

This dictionary is constructed using the above-mentioned systems of measurements, multiple online conversion calculators, and the opinion and suggestions of an expert. The purpose is to convert all the quantities in one unit of measurement – grams. This dictionary contains conversions for all instances contained in the units of measurement dictionary.

Table 6.1: Examples with redundant words and/or phrases.

Ingredient
1 lb large shallots, **bulbs separated if necessary and each bulb halved lengthwise**
8 slices **store bought** bread**, each sliced diagonally and pressed-down in the middle**
12 oz **best available quality** salmon**, cut into paper thin pieces and arranged in rows in a container, separated, and left in the refrigerator for half an hour**
5 oz **coarsely chopped** gherkins, or **any type of** pickled cucumbers

6.3 Methodology

The methodology used for obtaining the ingredient embeddings corpus is presented in Figure 6.1a, and the methodology used for obtaining the recipe embeddings corpus is presented in Figure 6.1b. The generating ingredient embeddings, and combining the ingredient embeddings are already defined in previous chapters (Chapter 3 and Chapter 4). Here, we are going to focus on the data normalization part.

6.3.1 Data

For this study, all publicly available recipe datasets were explored, and five others, besides Recipe1M, that had the potential to be transformed into the necessary format were selected. The selected datasets including Recipe1M are:

1. Recipe1M – containing 51,500 recipes and the following data for each:
   - recipe title – a short textual description of the recipe;
   - structured list of ingredients;
   - recipe instruction – long textual description of step-by-step instructions for preparing the recipe;
   - nutrient content of ingredients – quantity in grams of fat, protein, saturates, sodium, and sugar per 100 grams of the ingredient for each ingredient;
   - quantity of each ingredient;
   - units of measurement per each ingredient – according to the household measurement system (cup, tablespoon, teaspoon, etc.);
   - weight in grams per each ingredient;
   - nutrient content – quantity in grams of fat, protein, salt, saturates, and sugars per 100 grams of the recipe;
   - FSA traffic light labels per 100 grams – for each of the four nutrients (fat, salt, saturates and sugars) one of the three labels from the FSA traffic light system is assigned:
     - red – high content;
     - orange – medium content;
     - green – low content.
2. Indian recipes – contains 6,871 recipes and the following data for each:

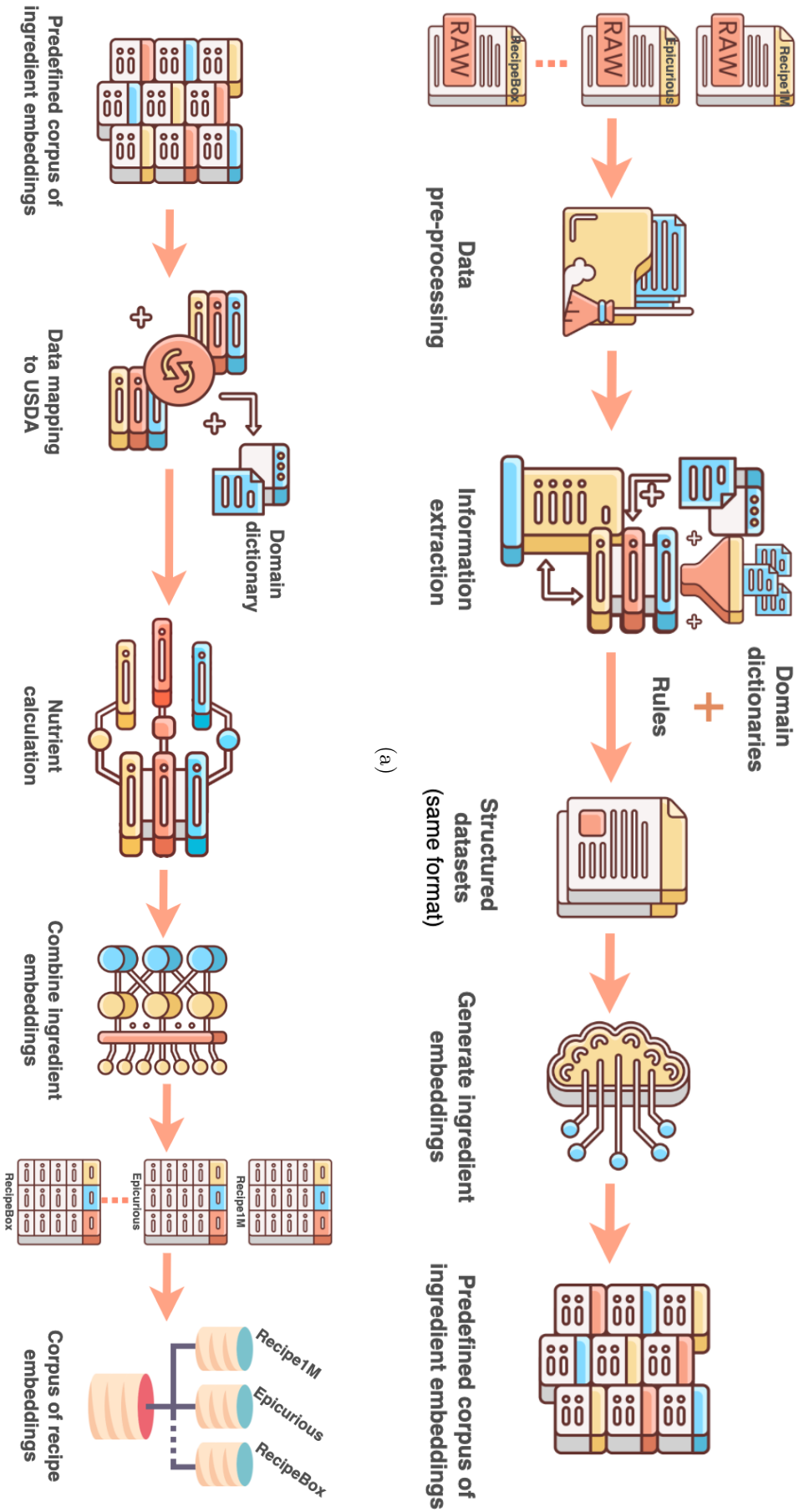

Figure 6.1: Flowchart of the methodology for obtaining the predefined corpus of: (a) ingredient embeddings; (b) recipe embeddings.

- recipe url;
- raw continuous text with ingredients, quantities and units of measurement in Hindi;
- servings;
- course;
- type of diet;
- recipe instruction – long textual description of step-by-step instructions for preparing the recipe.

3. Epicurious – contains 20,103 recipes and the following details for each:
   - recipe title – a short textual description of the recipe;
   - recipe url;
   - recipe instruction – long textual description of step-by-step instructions for preparing the recipe;
   - raw continuous text with ingredients, quantities and units of measurement;
   - calorie content;
   - nutritional values for protein, fat and sodium;
   - categories – keywords connected to the recipe;
4. Salad recipes – contains 82,243 recipes and the following details for each:
   - recipe title – a short textual description of the recipe;
   - recipe instruction – long textual description of step-by-step instructions for preparing the recipe;
   - cuisine;
   - raw continuous text with ingredients, quantities and units of measurement;
   - preparation time;
   - number of portions;
   - tags – keywords connected to the recipe;
5. Yummly28k – contains 27,639 recipes and the following data for each:
   - recipe title – a short textual description of the recipe;
   - raw continuous text with ingredients, quantities and units of measurement in English or German;
   - recipe image;
   - course information (example: breakfast, lunch, etc.);
   - detailed nutrient information – nutrient values for 94 nutrients;
6. RecipeBox – contains 39,802 recipes and the following data for each:
   - recipe title – a short textual description of the recipe;
   - raw continuous text with ingredients, quantities and units of measurement;
   - recipe instruction – long textual description of step-by-step instructions for preparing the recipe;
   - recipe url;
   - recipe image url;

6.3.2 Data normalization

After collecting the datasets, they need to be transformed into the needed format in order to apply the ML/DM pipeline. This process is separated into two parts:

1. Extracting the needed information:
    - Dictionary-based NER – for extracting the unit/s of measurement.
    - Rule-based NER – for extracting the quantity of the ingredient.
2. Data mapping – for mapping the ingredient (food item) to the USDA FCDB.

6.3.2.1 Extracting information from unstructured recipe data

Before any step is taken towards transforming the datasets into the needed format, it is important to note that two of the datasets contain data that is not in English – Indian recipes and Yummly28K. This means that we are dealing with multilingual data. In order to obtain the translation of the data, the Google Translate API in Python [185] is used. The results obtained in this way are reliable because we are dealing with one sentence text, that carries little to no context.

Now, let us define the ideal format that is needed for applying the ML/DM pipeline. Preferably, we would have dataset D with n number of instances/recipes and m number of features representing those instances. In those m features out of our importance is that we have the following:

1. Name of recipe;
2. Nutritional information about the whole recipe (on 100 grams);
3. List with names of ingredients;
4. List with quantities in grams for each ingredient (ideally on 100 grams);

Unfortunately, none of the datasets come in this format, only Recipe1M has the name of the ingredients and the quantities separately. The other five dataset all have raw continuous text with ingredients, quantities and units. Therefore, the first step is for each recipe in each dataset to separate the names of the ingredients, the quantities and the units into three separate lists. This process would be just a simple task of text splitting, i.e. string splitting, which can be performed using a simple function/s in Python [186] and search patterns using regular expression [187], [188], if this text is following some kind of a structure. An example of how this would work is given in Table 6.2.

Table 6.2: Example of well-structured not separated ingredient list in a recipe dataset.

Example	2 cups flour, whole wheat; 15 grams baking powder; 300 milliliters 3.5% milk; 2 tablespoons cocoa powder.
List of quantities	**[**2, 15, 300, 2**]**
List of units	**[**cups, grams, milliliters, tablespoons**]**
List of names of ingredients	**[**flour whole wheat, baking powder, 3.5% milk, cocoa powder**]**

When inspecting the datasets, more specifically the raw continuous text that contains the list of ingredients, quantities and units, it is more than evident that this technique will

not be enough in this case, in view of the fact that the text is not following any pattern or rule. The text of our interest in all of the recipe datasets did not have any kind of structure. There were a few things that should be taken into account:

1. The measurement units:
    - Can be of a different type:
        - from the the International System of Units (SI), i.e. the metric system (e.g. grams, milliliters)
        - a household measurement (e.g. cup, tablespoon, teaspoon)
        - an approximate measurement (e.g. splash, pinch, dash, handful, thumb sized, smidgen)
        - specific for a single food or a group of foods (e.g. "clove of garlic", "head of lettuce", "a cinnamonstick", "a stick of butter", "sprig of rosemary", "1 sausage link")
    - Can be written in different ways:
        - with full text (e.g. grams, tablespoon)
        - different type of abbreviations (e.g. for gram – g, gr; for tablespoon – tb, tbs, tbsp; for pound – lb, lbs, lbss)
    - Can be non-existent (e.g. "3 eggs", "2 onions"
2. The quantity:
    - Can be written in many different ways:
        - with numbers (e.g. "250 ml orange juice")
        - with words (e.g. "two apples")
        - a combination of a number and words (e.g. "4 and a half tbsp sugar", "Two 0.16 ounces packets instant noodles")
        - as indefinite articles (e.g. "a cup of milk")
    - Can be non existent (e.g. "salt and pepper")
    - Can be "user-defined" (e.g. "cinnamon, to taste")
3. The text representing the ingredient:
    - Can be a simple food (e.g. "2 1/2 pounds cherries")
    - Can be a branded food (e.g. "1 (40 grams) packet M&M's chocolate candies)
    - Can have an additional explanation about the food itself (e.g. "1 lb large shallots (8), bulbs separated if necessary and each bulb halved lengthwise", "$\frac{1}{2}$ pound skinless chicken breast (about 1 chicken breast, cut into $\frac{1}{2}$-inch dices)")
    - Can be something that is not food but it is written in the ingredient list as it is needed for the recipe (e.g. "1 sheet of parchment paper")

The first attempt is to formulate all these cases and put together rules for string splitting, including rules for extracting the quantity and extracting the unit of measurement.

1. Extracting the measurement unit/s – In order to extract the unit of measurement from the text, a dictionary is constructed, defined in Subsection 6.2.2 – therefore, every entity or entities from that dictionary found in the text of matter are defined and extracted as a unit of measurement.

2. Extracting the quantity/quantities – In order to extract the quantity from the text, we define certain rules and patterns:

   - If there are digit occurrences in the following format:
     - $D_{integer}$ $\frac{D_{integer}}{D_{integer}}$,

     Where D can be one or multi-digit integer number. This is summed into one number.
   - If there are digit occurrences in the following format:
     - $D_{integer}$ $D_{integer}$,
     - $D_{integer}$ D_{float},
     - D_{float} $D_{integer}$,
     - $\frac{D_{integer}}{D_{integer}}$ $\frac{D_{integer}}{D_{integer}}$,
     - $\frac{D_{integer}}{D_{integer}}$ $D_{integer}$,
     - D_{float} D_{float},

     where $D_{integer}$ can be a one- or a multi-digit integer number, and D_{float} is a two- or multi-digit float number. If there is an occurrence like this, the numbers from the occurrence are multiplied.

3. Extracting the food item – In order to extract the food item on the text that remains after the extraction of the measurement unit/s and quantity/quantities, we use the dictionary with redundant words, and remove any occurrences of any of the words belonging in the dictionary. Everything that is left is the food item.

There are a few special cases where some words can be considered as measurement units in some cases and be omitted in others. When either one of the following: "jar", "can", "packet", "package", "box", "bottle", and "container" appears in the text representing an ingredient, there are two scenarios:

1. If it is proceeded with a measurement unit from the defined dictionary (Subsection 6.2.2), it is omitted (e.g. "1 (500 grams) container Greek yogurt").

2. If it is found in a text where no measurement unit from the defined dictionary is extracted, it is considered a unit of measurement (e.g. "2 packets of Stevia").

For some food items there are specific units of measurement that correspond only to them. These words are extracted as units of measurement only when they appear in a pair with the food item/s they are correlated with. These word pairs are:

- "clove" and "garlic";
- "stick" and "butter"/"margarine"/"cinnamon"/"carrot"/"celery";
- "sprig" and "rosemary"/"thyme"/"mint"/"parsley";
- "link" and "sausage";
- "stalk" and "celery"/"green onion"/"spring onion"/"broccoli"/"kale"/"cauliflower";
- "sheet" and "gelatin";
- "cube" and "stock"/"butter","margarine";
- "head" and "cabbage"/"lettuce"/"cauliflower"/"broccoli";

Each of these, of course, have different weights for the different food items, this is taken into account in the conversion dictionary. Therefore, we can see that the process of extracting the unit/s of measurement is a hybrid process of NER with dictionary and additional rules. These rules, when applied, should result in separated lists of units of measurement (one or more), quantities (one or more) and food item.

However, after defining these steps, we stumbled upon more difficulties with the process of extracting the needed information:, to be more specific – text designated for one ingredient in the list of ingredients can contain more than one ingredient (concatenated with the conjunction "plus", "and" or "with"), contain multiple options for one ingredient (concatenated with the conjunction "or"), contain more than one quantity for one ingredient (concatenated with the preposition "to"), or a combination of two or more of the aforementioned. The solutions that were deemed fit for these cases are:

1. Containing conjunction "and" or "with":
    - If the condition: $(count(\text{"}and\text{"}) \geq 1 \vee count(\text{"}with\text{"}) \geq 1) \wedge count(\text{"}plus\text{"}) == 0$ is satisfied and there are two units of measurement extracted on each side of the conjunction, the conjunction "and" is replaced with "plus" and is treated as such forward (e.g. "2 4 oz packages of salmon and 5 tablespoons of lemon juice").
    - Any other cases with the conjunction "and" are dismissed (e.g. "1 lb large shallots (8), bulbs separated if necessary and each bulb halved lengthwise").
2. Containing conjunction "plus":
    - If $count(\text{"}plus\text{"}) \geq 1$ – split on "plus" and separate text to as many ingredients as the count of "plus" in the text (e.g. "1/4 cup finely chopped sweet gherkins plus 2 tablespoons pickled juice from the jar, plus 12 whole gherkins").
3. Containing conjunction "or":
    - If $count(\text{"}or\text{"}) > 1$ – split on "or" and separate text to two parts, but keep the first part only;
    - If $count(\text{"}or\text{"}) > 1$ – check the position of the extracted unit/s of measurement, split into as many parts as the number of "or" and:
        - If there is one unit of measurement extracted, take the part with the unit of measurement.
        - If there is more than one unit of measurement, all or none of them with assigned quantity, and there is no "plus" in the ingredient, take the part with the first unit of measurement.
        - If there is more than one unit of measurement, some with unassigned quantity, and there is no "plus" in the ingredient, take the part with the unit of measurement with assigned quantity.
4. Containing preposition "to":
    - If "to" is in a specific pattern involving integer, and/or float numbers, and/or digits separated with "/" (e.g.: "2 14 1/2- to 15-ounce cans diced tomatoes in juice"), the first part of the pattern when split by the word "to" is kept and appended to the rest of the text.

Despite these cases we determined that there are other cases with occurrences of two ingredients expressed within text designated for one, to be more specific, text containing any of the following phrases: "mixed with", "mixed in", "beaten with", "dissolved in", "sauteed in", "diluted", "combined", or the verb "add". This is resolved by treating and replacing these phrases with "plus". In Table 6.3, some examples are given.

Table 6.3: Examples with phrases depicting more than one ingredient.

Ingredient
5 to 6 anchovies **add** 1 teaspoon red pepper flakes
1 large egg **beaten with** pinch of salt
1 teaspoon corn starch **dissolved in** 2 tablespoons milk
1 (500 grams) container Greek yogurt **mixed with** 2 packets of Stevia

Another problem that is reoccurring is the appearance of more than one quantity for one ingredient, meaning having information that is extra in the text. This is occurring in multiple scenarios and for each of these scenarios a fitting solution is applied:

1. If we extract two different patterns for quantity and one unit of measurement after the second pattern and there is no text between the two patterns that express quantity – The second pattern is kept as the quantity, and the first is removed from the text.
2. If we extract two different patterns that express quantity and one unit of measurement after the first pattern, immediately followed by the second pattern – The first pattern is kept as the quantity and the second pattern is removed.
3. If we extract two different patterns that express quantity, and two different units of measurement and there is no text between the two patterns except the unit of measurement connected with the first pattern – The first pattern is kept as the quantity of the ingredient.
4. If we extract two different patterns that express quantity, and there is no unit of measurement extracted and there is no text between the two patterns – The first pattern is kept as the quantity of the ingredient.

Before applying any rules, the text is pre-processed. For the pre-processing, a few rules are followed:

1. Brackets are removed.
2. The text is tokenized.
3. The text is lemmatized.
4. Redundant words from the constructed dictionary are removed.
5. Words from the branded food dictionary are identified, and no further pre-processing is done on them.
6. All special characters are removed with the exception of:

- "%" – if it is preceded with a number.
- "&" – it is replaced with "and".
- "+" – it is replaced with "plus".
- "/" – if it is in between digits.
- "-" – is replaced with "to" if it is between digits.

7. Words depicting numbers are changed with the number/s they represent (special cases are added for "half a", "half of", "half from", which are replaced with "1/2").

8. If the text starts with "a/an" followed by a unit of measurement, the "a/an" is replaced with "1".

9. If the text starts with "a/an" and there is no unit of measurement present and there is no pattern depicting quantity present in the text, the "a/an" is replaced with "1".

10. If there is a dot – "." or a comma – "," present after the extracted quantity and measurement and there is no "and/or" in the text, everything after the dot or comma is removed.

After all of this is applied to all the instances from the datasets we have six new datasets in the format needed for generating the embeddings – the RL part of the methodology presented in Chapter 5.

6.3.2.2 Data mapping to USDA FCDB

In order to apply the supervised ML part, we first need to determine which nutrient values we want to predict. We can see from the descriptions of the datasets, given in Subsection 6.3.1, that only three datasets contain nutrient values, and all the same nutrient values. For the purpose of obtaining the nutrient values for all datasets, we went the route of mapping the data to a FCDB, and the FCDB of choice is the USDA FCDB [177], which made sense not only because the datasets come mostly from websites that are based in the USA, but also for the fact that it is the biggest integrated data system that provides expanded nutrient profile data and links to related agricultural and experimental research. This database is used for two goals:

1. Obtaining the quantities in grams that are missing in the datasets – These quantities are missing because there is no unit of measurement present, just the food item and/or a number (e.g.: "2 eggs", "3 apples", "1 standard size chicken breast", etc.).

2. Obtaining nutrient values for each ingredient in order to calculate the nutrient values of the recipe.

Before the data mapping process, a new dataset is constructed, consisting of all the unique ingredients extracted from all the six recipe datasets. In total this dataset contains 71,641 instances. The two aforementioned goals were both achieved with mapping the ingredients to a food item in the USDA database. We have dealt with this type of mapping in previous studies [40], [51], [161], [189] where we used a lexical similarity approach involving Part of Speech (POS) tagging and probability theory, described in detail in [51], [161] and mentioned in Subsection 2.1.2 of Chapter 2. This method is limited because of its nature to base the matching on intersection of nouns, which the POS tagging method may not capture. More-over, the instance may not contain any nouns or the POS tagging method may not tag the tokens (words) correctly (i.e. the nouns may be tagged as other POS, for example "orange" may be tagged as a adjective instead of a noun). In this study,

it is of real importance to obtain matches for as many as possible ingredients, therefore we could not rely only on this method. We do incorporate this method in the mapping – we include the nouns, verbs, adjectives, and numbers and reformulate the similarity as shown in Algorithm 6.1.

For performing the mapping from the USDA FCDB we extracted the food items from all four datasets mentioned in Subsection 6.2.1. The initial merged dataset contained 298,318 items, after removing duplicates, the final merged dataset contains 290,519 food items. This dataset is then submitted to some of the processes described in Subsection 6.3.2.1:

1. Each of the food items is pre-processed:
    - Everything in brackets is removed.
    - The text is tokenized.
    - The text is lemmatized.
    - Redundant words from the constructed dictionary (described in Subsection 6.2.2) are removed.

The two datasets – with the unique ingredients from the six recipe datasets and the pre-processed food items from the four datasets from the USDA FCDB are then inputs to the data mapping procedure, described in Algorithms 6.2 and 6.3. The mapping procedure is an ensemble of multiple similarity measures sorted in order of importance. Two food items – one from USDA and one ingredient from the datasets – are considered a match if one of the following similarity measures is satisfied:

1. Same set of lemmas.
2. Same set of nouns and the lexical similarity function is greater that 0.
3. The intersection of nouns is not an empty set and the lexical similarity function is greater than 0.
4. The intersection of the unions of nouns, verbs, adjectives and numbers of the two food items is not an empty set and the lexical similarity function is greater than 0.
5. The Levenshtein distance [190] of the concatenated lemmas of both food items is smaller than the length of the concatenated lemmas of the ingredient that is being matched.

Algorithm 6.1: Lexical similarity measure.

Data: Set of nouns, verbs, adjectives and numbers from the ingredient we want to match $Nouns_i, Verbs_i, Adjective_i, Numbers_i$ food item from USDA $(Nouns_j, Verbs_j, Adjective_j, Numbers_j)$

Result: Lexical similarity score

$p_n = \frac{Nouns_i \cap Nouns_j}{Nouns_i \cup Nouns_j}$;

$p_{v+a+num} = \frac{((Verbs_i \cup Adjectives_j \cup Numbers_i) \cap (Verbs_j \cup Adjectives_j \cup Numbers_j))+1}{((Verbs_i \cup Adjectives_j \cup Numbers_i) \cup (Verbs_j \cup Adjectives_j \cup Numbers_j))+2}$;

$similarity_{index} = p_n \times p_{v+a+num}$;

Algorithm 6.2: Pre-process food items from USDA.

Data: Dataset with USDA food items: $FoodItems_{USDA}$

Result: Dataset with pre-processed USDA food items: $FoodItems_{USDA}$

for *Each food item* F_f *in* $FoodItems_{USDA}$ **do**
- Tokenize F_f and get tokens $[t_{f1}, ..., t_{fn}]$;
- Lemmatize $[t_{f1}, ..., t_{fn}]$ and get set of lemmas $\{F_{l1}, ..., F_{ln}\}$;
- Obtain POS tag for the token $\{t_{f1}: POS_{f1}, ..., t_{fn}: POS_{fn}\}$;
- **for** *Each token* t_{fj} *in* $\{t_{f1}\ POS_{f1}, ..., t_{fn}\ POS_{fn}\}$ **do**
  - **if** POS_{fj} *begins with 'NN'* **then**
    - Add t_{fj} to $Nouns_f$;
  - **else**
    - **if** POS_{fj} *begins with 'VB'* **then**
      - Add t_{fj} to $Verbs_f$;
    - **else**
      - **if** POS_{fj} *begins with 'JJ'* **then**
        - Add t_{fj} to $Adjectives_f$;
      - **else**
        - **if** POS_{fj} *begins with 'CD'* **then**
          - Add t_{fj} to $Numbers_f$;
        - **else**
      - **end**
    - **end**
  - **end**
- **end**
- Add $Nouns_f$ to $Nouns_{USDA}$;
- Add $Verbs_f$ to $Verbs_{USDA}$;
- Add $Adjectives_f$ to $Adjectives_{USDA}$;
- Add $Numbers_f$ to $Numbers_{USDA}$;
- Add $\{F_{l1}, ..., F_{ln}\}$ to $Lemmas_{USDA}$

end

Algorithm 6.3: Mapping ingredients to USDA.

Data: Dataset of unique ingredients: $Ingredients_{datasets}$;
Dataset with preprocessed USDA food items: $FoodItems_{USDA}$
Result: Datasets with corresponding matches from USDA: $Matches_{USDA}$

for *Each ingredient* I_i *in* $Ingredients_{datasets}$ **do**
  Tokenize I_i and get tokens $[t_{i1}, ..., t_{im}]$;
  Lemmatize $[t_{i1}, ..., t_{im}]$ and get set of lemmas $L_I = \{I_{l1}, ..., I_{lm}\}$;
  Obtain POS tag for the token $\{t_{f1}: POS_{i1}, ..., t_{in}: POS_{im}\}$;
  for *Each token* t_{ij} *in* $\{t_{i1}\ POS_{i1}, ..., t_{in}\ POS_{im}\}$ **do**
    if POS_{ij} *begins with 'NN'* **then**
      Add t_{ij} to $Nouns_i$;
    else
      if POS_{ij} *begins with 'VB'* **then**
        Add t_{ij} to $Verbs_i$;
      else
        if POS_{ij} *begins with 'JJ'* **then**
          Add t_{ij} to $Adjectives_i$;
        else
          if POS_{ij} *begins with 'CD'* **then**
            Add t_{ij} to $Numbers_i$;
          else
            continue;
          end
        end
      end
    end
  end
  for *Each item from USDA with id,* $Lemmas_{id}$, $Nouns_{id}$, $Verbs_{id}$, $Adjectives_{id}$, *and* $Numbers_{id}$ **do**
    if $Lemmas_{id} == L_I$ **then**
      Append the *id* to $matches_{I_i}$;
    else
      if $Nouns_{id} == Nouns_i \wedge Standfood_{sim}(POS_i, POS_{id}) > 0$ **then**
        Append the *id* to $matches_{I_i}$;
      else
        if $Nouns_{id} \cap Nouns_i \neq \emptyset \wedge Standfood_{sim}(POS_i, POS_{id}) > 0$ **then**
          Append the *id* to $matches_{I_i}$;
        else
          if $\{Nouns_{id} \cup Verbs_{id} \cup Adjectives_{id} \cup Numbers_{id}\} \cap \{Nouns_i \cup Verbs_i \cup Adjectives_i \cup Numbers_i\} \neq \emptyset \wedge Standfood_{sim}(POS_i, POS_{id}) > 0$ **then**
            Append the *id* to $matches_{I_i}$;
          else
            if *Levenshtein*$(Lemmas_{id}, Lemmas_i) <$ *length*$(Lemmas_i)$ **then**
              Append the *id* to $matches_{I_i}$;
            else
            end
          end
        end
      end
    end
  end
end

6.4 Results and Discussion

The results of this study are intertwined with the description of the methodology in Section 6.3, as this methodology is constructed for the needs of bringing these specific six datasets to the format needed for the ML/DM pipeline and the calculation of the recipe embeddings presented in Chapter 5. When or if a new recipe dataset is to be submitted to this methodology, it may require additional pre-processing, if for example, the list of ingredients is given in another language, or the whole list of ingredients is given as a single block of text with a specific delimiter etc. It can also occur that no additional pre-processing is needed and the new recipe dataset can be directly submitted to the methodology.

6.4.1 Generating the predefined corpus of ingredient embeddings

The results from the data normalization process are six harmonized recipe datasets with a structured format, mapped to a FCDB. After the procedure of extracting information from the unstructured recipe datasets, a dataset with 71,641 ingredients is constructed. On this dataset ingredient embeddings are generated using four embedding algorithms and the following parameters:

1. Word2Vec – architectures: CBOW and SG, vector dimension: 50, 100, and 200, sliding window: 2, 3, 5, and 10, and heuristics for combining the individual word embeddings: sum and average.

2. GloVe – vector dimension: 50, 100, and 200, sliding window: 2, 3, 5, and 10, and heuristics for combining the individual word embeddings: sum and average.

3. Doc2Vec – architectures: PV-DM and PV-DBOW, vector dimension: 50, 100, and 200, sliding window: 2, 3, 5, and 10, and heuristics for combining the individual word embeddings: sum and average.

4. BERT – combining the last four layers of the neural network: summing and concatenating.

The result is a predefined corpus of ingredient embeddings.

6.4.2 Generating the predefined corpus of recipe embeddings

After the mapping of the dataset with ingredients to the USDA FCDB, the information about nutrient values from the USDA FCDB is extracted and added alongside each ingredient, that is nutrient values for 100 grams of the ingredient. The missing quantities (as mentioned in Subsection 6.3.2.2) are extracted from the portions sizes (expressed in grams) of the food items from the USDA FCDB. Therefore, every needed data are obtained in order to calculate the recipe embeddings with the domain heuristic.

To generate the recipe embeddings, first we obtain the nutrient values of the recipe per 100 grams, by calculating the quantity of each ingredient per 100 grams of the recipe and scaling the nutrient values accordingly. For our target values we selected the following five nutrients: fat, protein, sugar, saturated fat, and sodium. This choice is obvious since the three datasets that have nutrient values have all or some of these. Using the predefined corpus of ingredient embeddings (presented in Subsection 6.4.1) and the calculated nutrient values with the mapping to the USDA FCDB we generate the recipe embeddings for the six recipe datasets by combining the ingredient embeddings with the domain heuristic

presented in Chapter 5. This way of generating the domain-specific recipe embeddings, with the same dataset of ingredient embeddings, brings them all in the same landscape, i.e. the same feature space. The end result are six datasets of recipe embeddings, which are combined to form the predefined corpus of recipe embeddings, consisting of 219,765 recipe embeddings.

6.4.3 Predictive modeling

After obtaining the embeddings, the next step is the predictive modeling, single-target regressions with six different types of regressions (Linear, Ridge, Lasso, Elastic Net, Decision Tree, Random Forest, and Neural Network regression) for predicting five nutrients (fat, protein, sugar, saturated fat, and sodium) using four different types of embedding algorithms (Word2Vec, GloVe, Doc2Vec, and BERT). In total there were 3,660 models trained: 1,440 Word2Vec, 720 GloVe, 1,440 Doc2Vec, and 60 BERT models. The predictive models are evaluated with the next steps:

1. Hyper-parameter tuning – from the scikit-learn library in Python [146]): GridSearchCV (all parameter combinations) for Linear, Ridge, Lasso, Elastic Net, and Decision Tree regression, and RandomizedSearchCV (sample a given number of candidates from a parameter space) for Random Forest and Neural Network, because they are more complex algorithms and the exhaustive search from GridSearchCV requires considerably much longer execution time.
2. Training with the best parameters from the hyper-parameter tuning.
3. K-fold cross-validation to estimate the prediction error.
4. Calculating the domain-specific accuracy (defined in Chapter 3, Section 3.4), and the baseline mean and median, for comparison.

For comparing the results, we also trained models with the recipe embeddings obtained without the domain heuristic for merging. The detailed tables with the results are presented in Appendix A, while in Table 6.4 a shortened version of the results is given – presenting the average accuracies obtained with the domain heuristic.

6.4.4 Discussion

With this study we achieved harmonization over the meta-data of six different heterogeneous recipe datasets, as well as provide the research community with two corpora of predefined domain-specific embeddings (one with ingredient embeddings and one with recipe embeddings), and four different domain-specific dictionaries, created with the help of a domain expert – a nutritionist:

1. Dictionary for units of measurement.
2. Dictionary for converting units of measurement to grams.
3. Dictionary for branded foods.
4. Dictionary for redundant words specific to text from recipe data.

This study included several stages in order to come from a raw unstructured recipe textual data to a structured dataset, then to a dataset eligible for applying the proposed ML/DM pipeline. The process of data normalization was a demanding task on its own, inducing many steps, defined based on the structure and format of these specific six recipe

Table 6.4: Average accuracies obtained with the embeddings merged with the domain heuristic.

Recipe Dataset	**Target**	**Embedding algorithm**			
		Word2Vec	**GloVe**	**Doc2Vec**	**BERT**
Recipe1M	Fat	67.86	62.96	62.53	77.64
	Protein	86.22	83.95	82.60	89.32
	Saturated fat	85.35	81.69	81.90	90.95
	Sugars	60.76	61.50	55.62	85.64
	Sodium	91.00	90.50	89.60	92.06
Indian recipes	Fat	67.77	67.14	66.08	74.42
	Protein	81.40	81.07	83.16	86.61
	Saturated fat	85.85	85.56	85.02	87.69
	Sugars	59.88	59.44	60.98	75.86
	Sodium	92.88	92.88	92.72	92.78
Epicurious	Fat	54.04	52.77	47.12	59.32
	Protein	78.84	75.88	66.66	79.82
	Saturated fat	86.05	86.03	84.34	86.93
	Sugars	61.96	55.75	45.00	60.85
	Sodium	90.52	91.12	90.56	89.98
Salad recipes	Fat	44.98	44.02	40.56	46.60
	Protein	69.79	66.96	59.23	70.66
	Saturated fat	83.81	83.46	82.44	84.56
	Sugars	50.38	45.67	38.43	51.86
	Sodium	88.06	89.95	90.76	89.92
Yummly28K	Fat	53.47	51.6	44.79	57.365
	Protein	76.51	73.69	60.52	77.26
	Saturated fat	85.69	85.26	83.54	86.58
	Sugars	64.17	58.54	44.60	65.31
	Sodium	89.64	88.25	87.26	82.92
Recipe box	Fat	59.27	56.51	51.17	67.43
	Protein	80.68	83.79	78.30	70.94
	Saturated fat	83.79	83.09	80.41	86.64
	Sugars	59.02	90.84	42.91	64.96
	Sodium	89.88	90.84	90.13	89.67

datasets. Despite it being constructed for these six recipe dataset, it is applicable to new dataset as well, but it may require additional minor rules and/or changes to accommodate the new dataset, such as translation of ingredients (if the dataset comes in another language), separating text for each ingredient (if the ingredient list comes as a whole).

After obtaining the predefined ingredient embedding corpus, we did a manual inspection of the instances, and we could notice that there are some ingredients are very similar and represent the same food (nutritionally), for example: "tomato paste concentrate", and "concentrated tomato paste"; "tomato paste unsalted", and "no-salt tomato paste"; "dried grapes" and "raisins"; "fat-free sweetened condensed milk", and "nonfat sweetened condensed milk". This may be looked as a flaw in terms that there are different embeddings for the same food, but on the other hand it is good to have these versions as recipe data is a very diverse type of data due to the different ways people express themselves, different translations of foods, and foods specific to a certain region or country. We went further and inspected how these instances that were in a sort repetitive were mapped to the USDA FCDB for obtaining the nutrient values. We found out that the data mapping ensemble method employed in this study did map these instances to the same food from the USDA FCDB. For example: "fat free sweetened condensed milk", and "nonfat sweetened condensed milk" were both mapped to "sweetened fat free condensed milk", therefore both instances were assigned the same nutrient values.

Predefined domain-specific embeddings, as discussed previously in Section 6.1, already exist in several domains, as well as the Food and Nutrition domain, but none for recipe data that includes external domain resources, and a domain heuristic for merging. The predefined corpus of ingredient embeddings is currently the biggest existing corpus of food items that are part of recipes – as ingredients. This corpus, as a standalone predefined corpus of embeddings, can easily be used and transferred into other ML tasks and studies beside the task presented here – predicting nutrient values.

The predefined corpus of recipe embeddings is the largest corpus of recipe embeddings, consisting of domain-specific embeddings for 228,158 different recipes, which makes it a powerful resource in the Food and Nutrition domain, and the ML community, as training new embeddings for a specific ML task is a demanding and timely process. These embeddings, as the ingredient corpus can be transferred to other ML tasks and used in other research studies.

We also evaluated the effectiveness of the ML/DM pipeline and the domain heuristic for merging multi-word embeddings, the results showed that the ML pipeline is not dataset-biased and can perform on heterogeneous data. The diversity of the data in these datasets also provides a base for generalizing the prediction models to many different scenarios with a minimum amount of data.

Chapter 7

Generalizing the Knowledge Learned by Predictive Modeling on Heterogeneous Recipe Data

This Chapter presents a study for generalizing prediction models trained on heterogeneous recipe datasets – models trained on one dataset to be used on other/s. This chapter starts with the motivation for this study – Section 7.1, continues with related work specific for this study in Section 7.2. The methodology is defined in Section 7.3 of the chapter, while the evaluation and results are presented in 7.4.

7.1 Problem Definition

In this decade, the changes in AI and ML are actively leaning toward data-centric AI – which is systematical engineering of data for building an AI system. By definition, to build an AI system, you define the problem, obtain the data, choose an algorithm, implement it with code, and train it and test it on your data. The most demanding step in recent years is the improvement of the code for implementing and training the algorithm, while the process of obtaining and preparing the data is almost always a no-brainer and very simple. For many problems there already exists a trained predictive model. Therefore, a shift from model- and architecture-focused to data-focused in AI is more than obvious. The data-centric movement is slowly but surely moving the focus towards improvement of the data in terms of quality.

Our developed ML/DM pipeline presented in Chapter 3 is a predictive modeling pipeline that focuses on the data aspect, and builds models that are both domain and data-driven. In order to test the generalization of the models built from the presented ML/DM pipeline, we decided to conduct a study as an extension to the work presented in Chapter 6. The study involves the newly obtained predefined corpora of ingredient and recipe embeddings including six heterogeneous recipe datasets. In Chapter 6, we presented evaluation results for the presented ML/DM pipeline on heterogeneous recipe datasets, with bringing together their feature spaces, more correctly said – creating a shared feature space, then training a predictive model on one dataset and testing on the same dataset.

In this study, we focus on generalization of the predictive modeling. Predictive models are usually trained and tested on one dataset. Generalization is a term that refers to a predictive model's ability to react to new data, i.e. the model's ability to adapt properly

to previously unseen data, drawn from the same distribution as the one used to create the model. Therefore, to explore the generalizability of a predictive model we explore how a predictive model trained on one dataset performs on other datasets. Using the six recipe datasets and their recipe embeddings from the predefined corpora described in Chapter 6, we train predictive models for predicting five nutrient values: fat, protein, saturated fat, sugars, and sodium. These predictive models are trained separately for each one of the datasets then tested on the rest of the five recipe datasets. As generalization in ML in highly related to the distribution of the data [191], [192], we explore how to define the generalizability of the predictive models (trained separately on each of the six recipe datasets) from their distributions, i.e. the distributions of the recipe embeddings of their instances in the feature space.

7.2 Related Work

With a classical ML pipeline (previously shown in Figure 1.1) we can train predictive models that provide good solutions when the model is tweaked for the problem in hand, but the problem is that they are often over-fitted and cannot be generalized. The main question open here is – how is this done. Unfortunately, there are no studies in the direction of answering this question in the Food and Nutrition domain. We have, however, previously dealt with a matter of similar kind involving time series data in another study [48], where we present a new pipeline for landscape analysis of time-series machine learning datasets that allows us to select a diverse portfolio of benchmark datasets, and it reduces the presence of performance assessment bias through bootstrapping evaluation. We demonstrate the pipeline's ability to identify problems with non-redundancy and representativeness in the benchmark by combining a big multi-domain representation corpus of time-series-specific features and the results of a large empirical study of time-series classification (TSC) benchmark. We pointed out the problems of tailoring and tuning the ML methods on specific regions of the landscape. As a byproduct of the study we provide a collection of datasets that should be taken into consideration while benchmarking novel TSC approaches because they are evenly dispersed across the landscape space. This kind of a generalization is also mentioned in a study for selecting the representative benchmark set for optimization algorithms [193]. However, in many domains, and many types of data, as well as the Food and Nutrition domain, and recipe data, this problem is open, and not dealt with.

7.3 Methodology

For achieving our goal for this study we adhere to the methodology for landscape analysis to define the level of generalization between different datasets presented on the flowchart in Figure 7.1, consisting of the following steps:

1. Pre-process the datasets and bring them to the same format (described in Chapter 6).

2. Generate representations for the instances, and bring them to the same feature space.

3. Conduct predictive modeling – train predictive models on one dataset and test the models on all other datasets.

4. Define and formulate quantitative indicators and calculate the generalizability indexes:

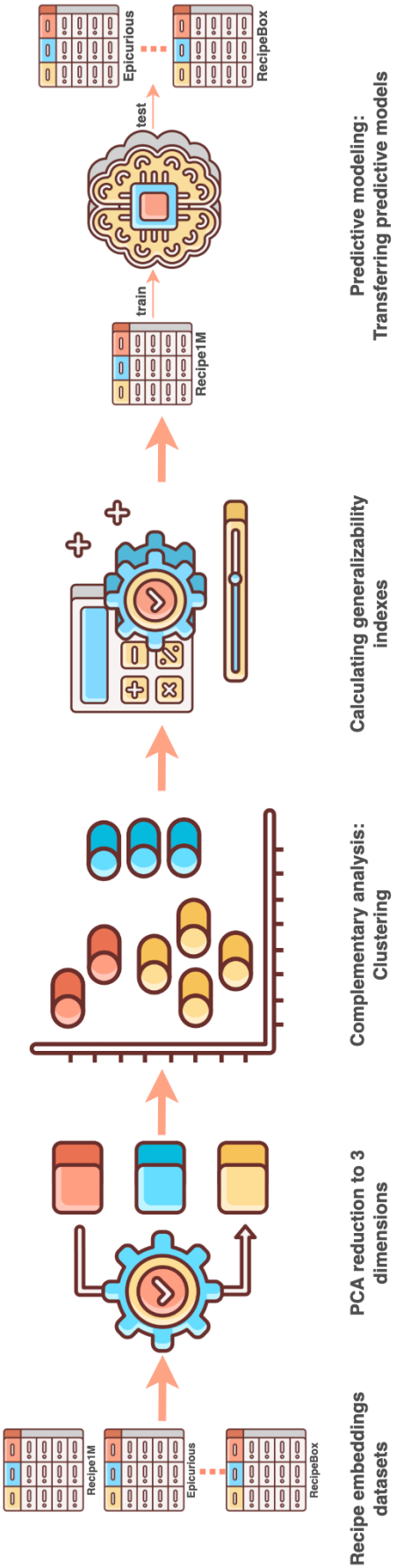

Figure 7.1: Flowchart of the methodology.

- Choose a setting of embedding algorithm with its corresponding parameters from the results from Chapter 6 – designate the settings of embedding algorithm and parameters that yielded the top five accuracies for each dataset and each target and select one that appears in all.
- Bring the recipe embeddings to a three dimensional space with a dimensionality reduction technique – we chose PCA [157] as it is one of the most important methods of dimensionality reduction for visualizing data, and it tries to preserve the global properties (eigenvectors with high variance) while it may lose low-variance deviations between neighbors.
- Cluster the reduced embeddings with a clustering algorithm – we chose k-means [101] as we do not have existing group labels, and we want to group our data.
- Calculate the distribution of instances from each dataset across all clusters – how many instances there are in each cluster from each dataset.
- Calculate a quantitative indicator – we define an index, called **generalizability index**, which is an indicator of how generalizable a predictive model is i.e. how a predictive model trained on one dataset will perform on the dataset in question. These indexes form a matrix, which we call generalizability matrix.

Let us now define the generalizability index:
If we have n datasets with recipe data, then for D_i as the i^{th} recipe dataset from the n datasets, RE_i is the dataset with the recipe embeddings of D_i reduced with the PCA technique to three dimensions, and RE is the dataset with the recipe embeddings of all n datasets, again, reduced with the PCA technique to 3 dimensions. We then cluster the RE dataset with a clustering algorithm in k clusters, and calculate the generalizability index of D_i for D_j (where D_j is the j^{th} dataset out of the n) as:

$$G_{ij} = 1 - \sum_{k=0}^{k=c} \left| \left(\frac{Instances_{k_i}}{Length_i} - \frac{Instances_{k_j}}{Length_j} \right) \right| \tag{7.1}$$

Where $Instances_{k_i}$ and $Instances_{k_j}$ are the number of instances in Cluster k for datasets D_i and D_j respectively (where D_i and D_j are one of the six recipe datasets), and $Length_i$ and $Length_j$ are the number of instances in the whole dataset for datasets D_i and D_j, respectively. The idea behind the generalizability index is the similarity between the number of instances that are distributed across the clusters. The formulation of the index is a symmetrical function, which means that $G_{ij} = G_{ji}$. To define all possible learning scenarios (i.e., when more datasets are available), no matter which dataset is used for training and which for testing, we can define a generalizability matrix such as:

$$\begin{array}{c} \begin{array}{ccccc} D_1 & ... & D_i & ... & D_n \end{array} \\ \begin{bmatrix} G_{11} & ... & G_{1i} & ... & G_{1n} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ G_{i1} & ... & G_{ii} & ... & G_{in} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ G_{n1} & ... & G_{ni} & ... & G_{nn} \end{bmatrix} \end{array} \begin{array}{c} D_1 \\ \vdots \\ \vdots \\ D_i \\ \vdots \\ \vdots \\ D_n \end{array} \tag{7.2}$$

Here, we need to point out that because of the symmetric property of the index definition, it is enough to calculate the upper- or lower-triangular part of this matrix.

7.4 Results and Discussion

1. Data – The starting point are the six recipe datasets, described in Chapter 6, which, again, we will refer to with the following names and abbreviations: Indian recipes, Recipe1M, Recipe box, Yummly28K, Epicurious, Salad recipes.

2. Generating embeddings:

   - Obtaining a combined dataset of ingredients contained in all six recipe datasets – The data contained in these datasets is normalized using multiple dictionary, and rule-based, lexical similarity methods, presented in Chapter 6, Subsection 6.3.2.1. After the data normalization and data mapping procedures, a combined dataset of unique ingredients is formed, containing 71,641 instances. This dataset, considering the types of dataset it is extracted from, consists of diverse food data, covering all food groups.
   - Obtaining ingredient and recipe embeddings – On the composed dataset with heterogeneous ingredients, word and paragraph embedding algorithms are applied, the whole process is explained in detail in Chapter 6. With combining the ingredients in one dataset, and generating embeddings for this dataset, the feature space for the predictive modeling, regardless of which dataset is chosen for training, is the same. After obtaining the ingredient embeddings, by using the domain heuristic described in Chapter 5, the recipe embeddings are calculated.

3. Predictive modeling – In contrast to the studies presented in previous chapters (Chapter 3, 4, 5, and 6), where the training process, i.e. the predictive modeling, is used for obtaining better predictions and researching if and how domain knowledge affects the prediction results, here the training process is aimed in the direction of investigating how transferable are the models depending on the datasets they are trained on, and how to generalize them. The predictive models are trained on each one of the six datasets separately and then the models are tested on the rest of the five datasets separately.
   The specific details for training the predictive models are:

   - Embedding algorithms:
     - Word2Vec – with both available architectures CBOW and SG, dimension size of 50, 100, and 200, "sliding window" of 2, 3, 5, and 10, and the two heuristics for combining the individual word embeddings, sum and average.
     - GloVe – with dimension size of 50, 100 and 200, "sliding window" of 2, 3, 5, and 10, and the two heuristics for combining the individual word embeddings, sum and average.
     - Doc2Vec – with both available architectures and PV-DM and PV-DBOW, dimension size of 50, 100, and 200, "sliding window" of 2, 3, 5, and 10, and the two heuristics for combining the individual word embeddings, sum and average.
     - BERT – with two different ways of combining the last four layers of the neural network, summing them and concatenating them.
   - Regression algorithms:

  - Linear regression;
  - Ridge regression;
  - Lasso regression;
  - ElasticNet regression;
  - Decision Tree regression;
  - Random Forest regression;
  - Neural Network regression.

- Evaluation:
  - Hyper-parameter tuning – we use two approaches for parameter search from the scikit-learn library in Python [146]: GridSearchCV and RandomizedSearchCV. The first one exhaustively considers all parameter combinations, and the second one can sample a given number of candidates from a parameter space with a specified distribution. The first one is used for the first five types of regression, while the second one for the last two, because they are more complex algorithms and the exhaustive search from GridSearchCV requires considerably much longer execution time, since we are training on a lot of combinations for the parameters given for the embedding algorithms.
  - Training the regressors with the best parameters from the hyper-parameter tuning.
  - Testing the trained models on the other five recipe datasets.
  - Calculating the domain-specific accuracy (defined in Chapter 3, Section 3.4, and the baseline mean and median, for comparison.

  In total, for each dataset, there are 3,660 models trained: 1,440 Word2Vec, 720 GloVe, 1,440 Doc2Vec, and 60 BERT models.

4. Clustering the instances of each dataset – From the observation of the results obtained by the predictive modeling in Chapter 6, we chose the embeddings generated with the Wor2Vec embedding algorithm with the following parameters: dimension 100, sliding window 3, architecture $CBOW$ and merging heuristic $average$, for the further experiments. The embeddings for all recipe datasets generated with this setting of parameters with the Word2Vec algorithm then undergo a dimensionality reduction to three dimensions with the dimensionality reduction technique PCA. The distributions of the reduced three-dimensional vectors of the recipe embeddings from the six datasets in the same vector (feature) space are presented for each dataset separately in Figure 7.2. From this figure we can see how each of the dataset is distributed in the feature space, and how different their distributions are. It is also very noticeable how widely distributed the Salad recipe dataset is compared to all the rest.

   After reducing the 100-dimensional vectors to 3-dimensional vectors, the vectors are clustered with the k-means clustering method using the sklearn library in Python [146]. The number of clusters is chosen using the average silhouette method, which assesses the quality of clustering. In other words, it establishes how well each object fits within its cluster. A high average silhouette width indicates a good clustering. For various values of k, the average silhouette method calculates the average silhouette of the observations. The optimal number of clusters k is the one that maximizes the average silhouette over a range of possible values for k [194]. The approach goes as follows:

   - Determine range of the number of cluster, in our case we chose the values from 3 to 12 clusters.

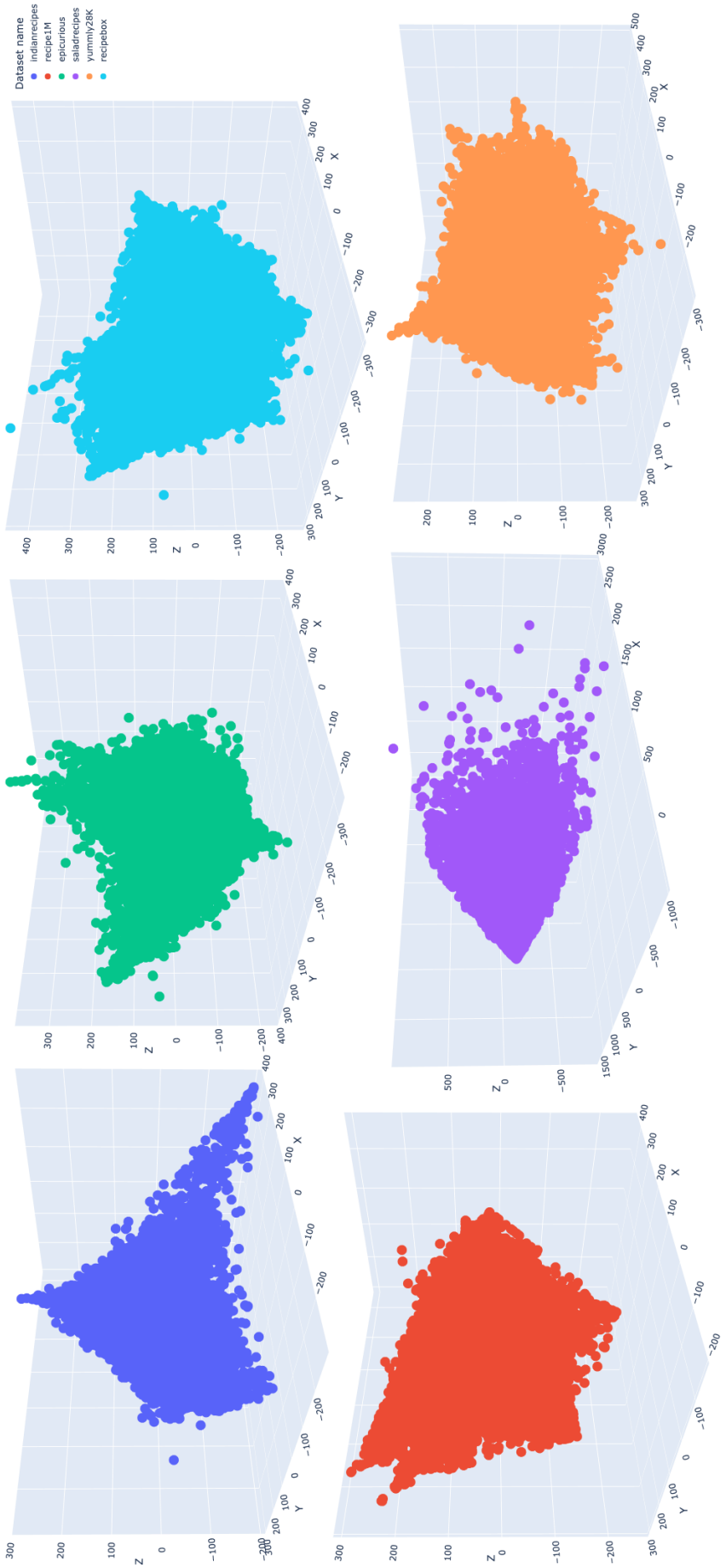

Figure 7.2: Reduced recipe embeddings for all six datasets obtained with the Word2Vec algorithm (architecture: CBOW, dimension: 100, sliding window: 3 merging heuristic: average) presented separately in the same feature space.

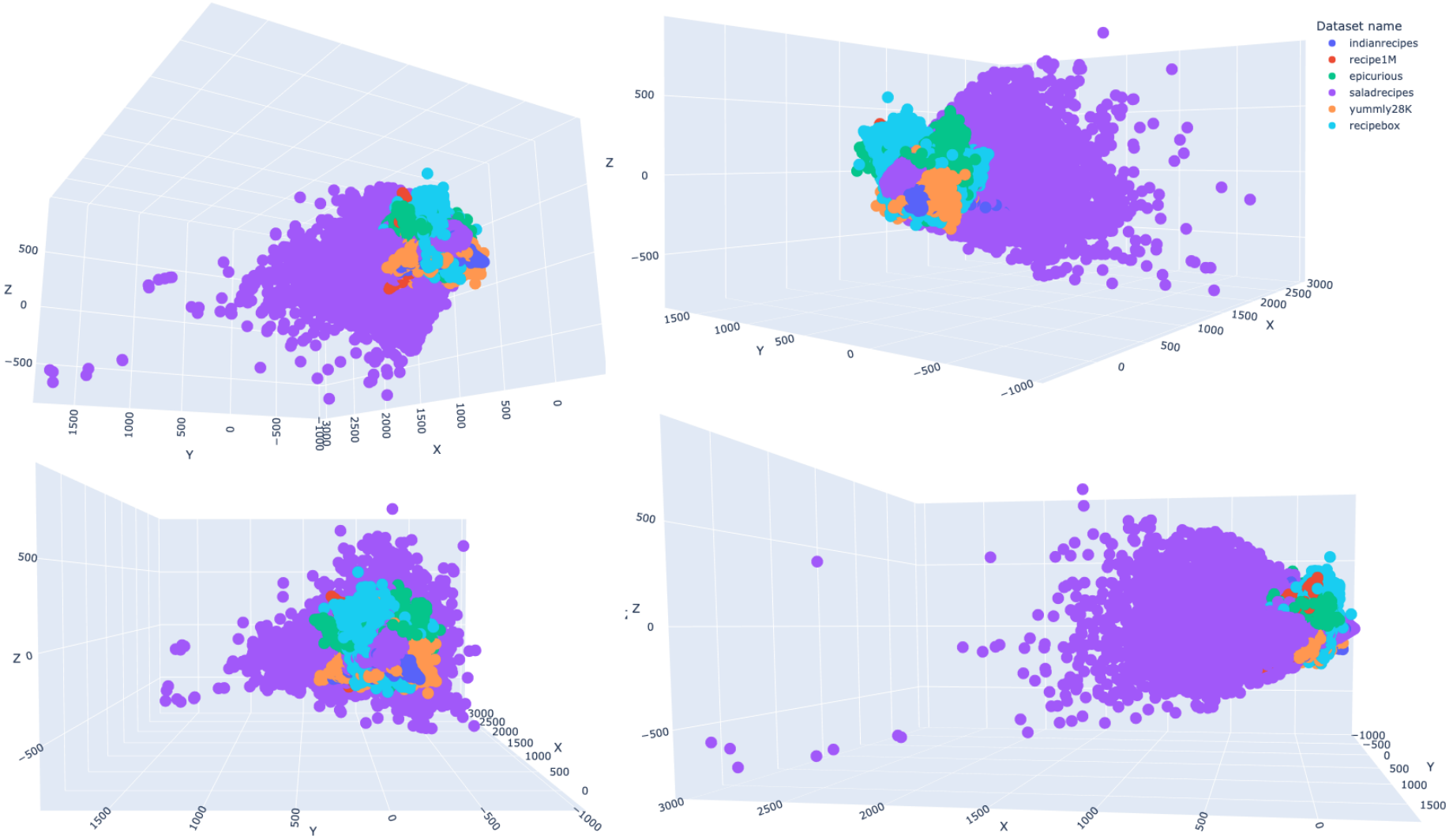

Figure 7.3: Reduced recipe embeddings for all six datasets obtained with the Word2Vec algorithm (architecture: CBOW, dimension: 100, sliding window: 3 merging heuristic: average) presented together in the same feature space.

- Compute k-means clustering algorithm for the different values of the number of clusters k.
- Calculate the average silhouette of observations for each value of the number of clusters k.
- Plot the curve of the average silhouette for all values of k.
- The maximum of the plot is considered as the appropriate number of clusters, in our case 8.

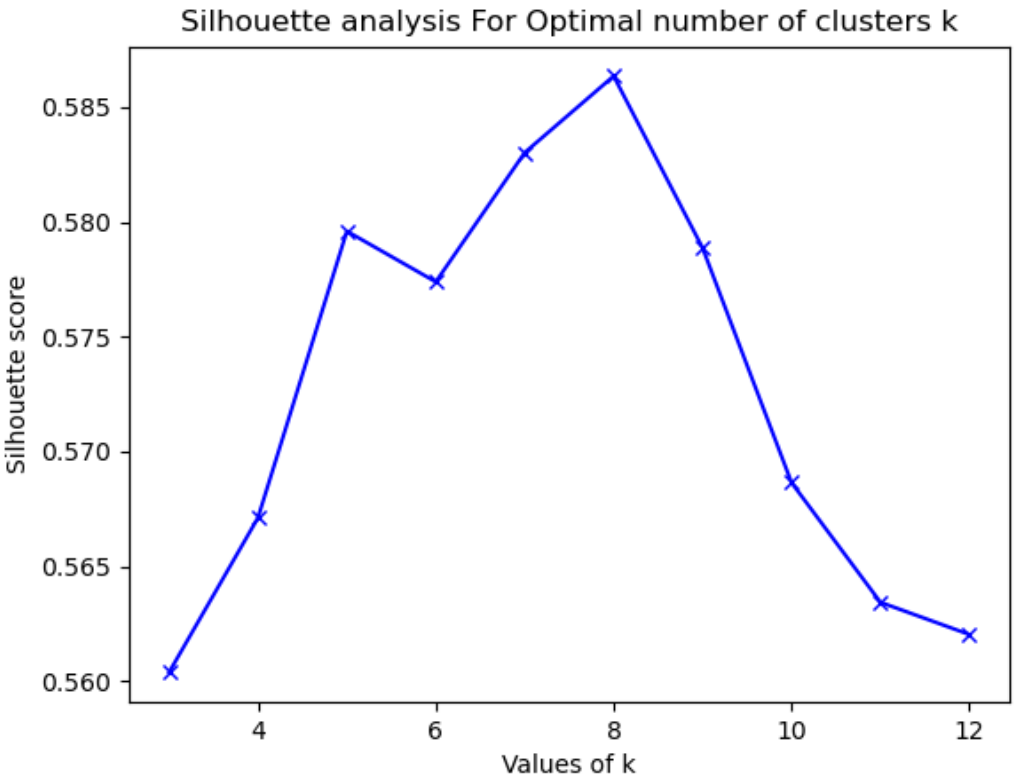

Figure 7.4: Curve of the average silhouette for values of the number of clusters k from 3 to 12.

The curve which is a result from this approach is depicted in Figure 7.4. The value for the silhouette score can be between -1 and 1, the higher the value, the more separated the clusters are – which is what we want. We can see that the maximum for the silhouette score is achieved when the number of Cluster k is 8. We can see that the value of the silhouette score for $k = 8$ is around 0.585, which means the cluster will have good separation [195]. With k=8 as the number of clusters, the reduced recipe embeddings are clustered. The clusters are presented in a 3D graph depicted in Figure 7.5.

5. Finding the distributions of the datasets across clusters – After clustering the instances we calculate the distributions of the instances of each dataset per cluster. In Table 7.1, the number of instances in each cluster is presented, and in Figures 7.6 and 7.7, two heat maps are presented, depicting percentages of each dataset per cluster and number of instances per cluster from each dataset respectively.

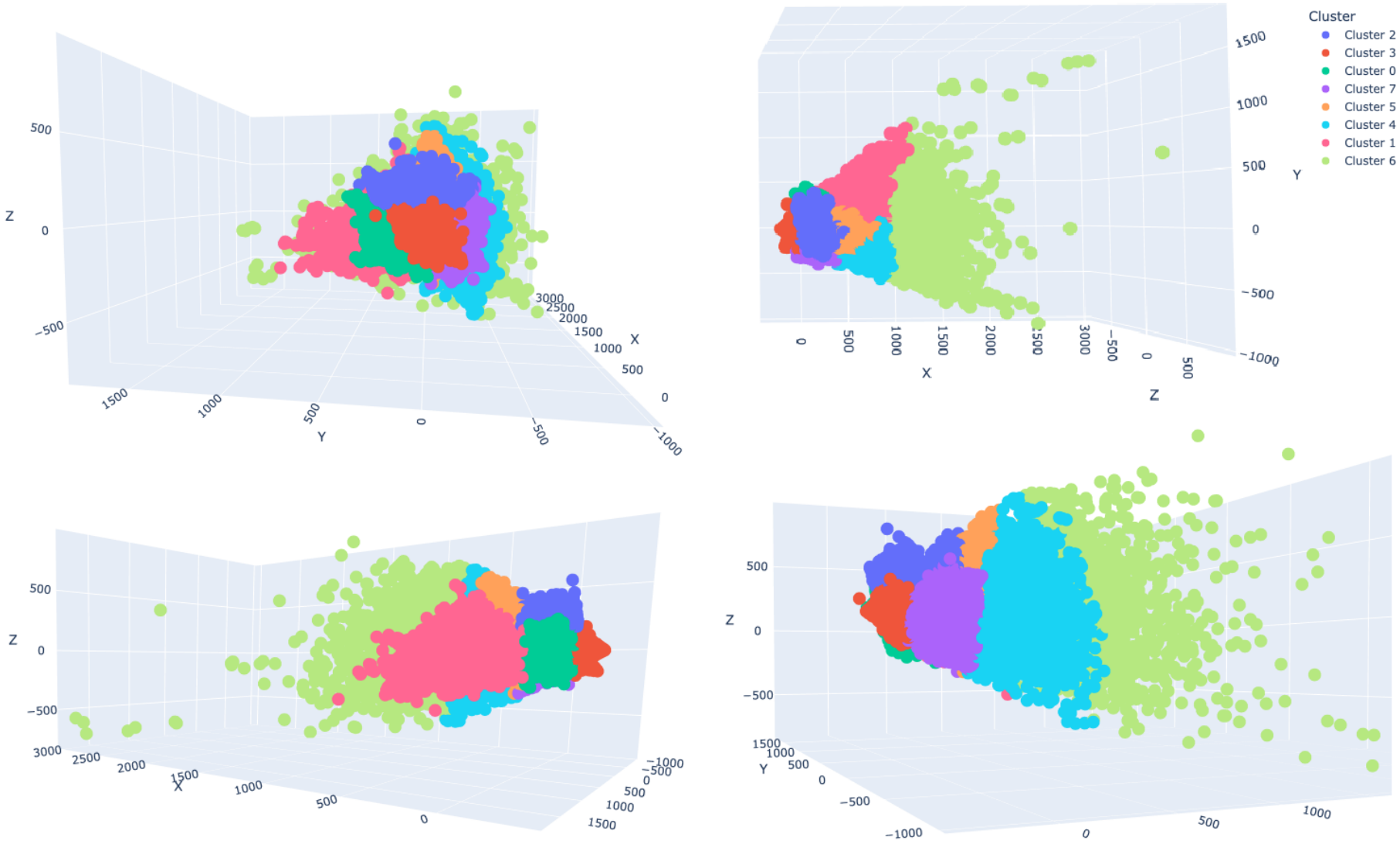

Figure 7.5: Clustering into 8 clusters of the reduced embedding produced with Word2Vec – architecture: CBOW, dimension: 100, sliding window: 3, merging heuristic: average.

Table 7.1: Number of instances in each cluster.

Cluster	0	1	2	3	4	5	6	7
Number of instances	44,576	9,363	33,971	24,751	7,387	32,274	1,966	65,477

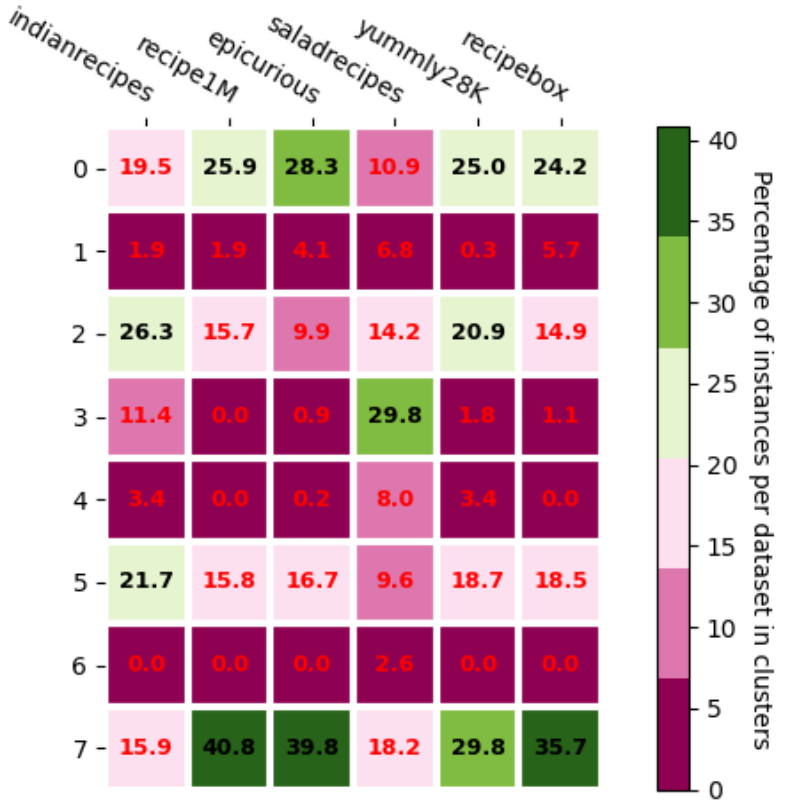

Figure 7.6: Percentage of instances of each dataset per cluster.

Table 7.2: Generalizability matrix.

	Indian recipes	**Recipe1M**	**Epicurious**	**Salad recipes**	**Yumml28K**	**Recipe box**
Indian recipes	1.00	0.38	0.31	0.34	0.61	0.43
Recipe1M	0.38	1.00	0.87	0.08	0.73	0.85
Epicurious	0.31	0.87	1.00	0.07	0.66	0.84
Salad recipes	0.34	0.08	0.07	1.00	0.15	0.19
Yummly28K	0.61	0.73	0.66	0.15	1.00	0.76
Recipe box	0.43	0.85	0.84	0.19	0.76	1.00

6. Calculating the generalizability indexes for each dataset and checking the correlation between the generalizability indexes and the results from the predictive modeling– The next step after finding the distributions of instances per cluster for each dataset is to calculate the generalizability indexes for all datasets according to the distributions of each dataset in each cluster using Equation (7.1), and generate the generalizability matrix (presented in Table 7.2). From the generalizability matrix we can observe the following:

    - For the Indian recipes dataset, the generalizability index is the highest for the

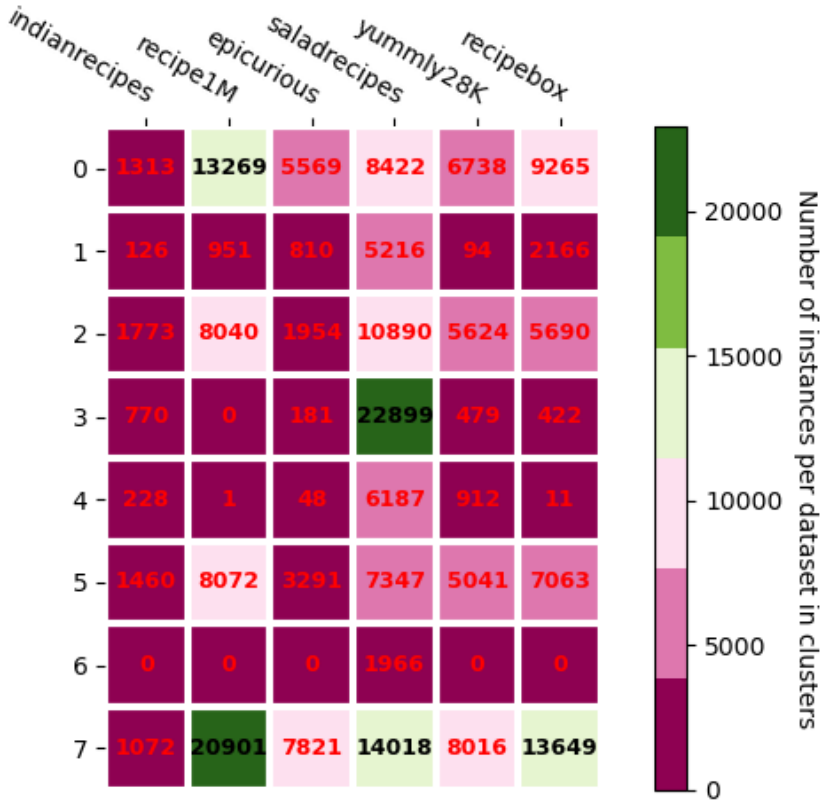

Figure 7.7: Number of instances from each dataset per cluster.

Yummly28K recipes dataset, while the generalizability indexes for the rest of the datasets are lower. From the results in Table 7.3, we can see that the accuracy is constantly the highest for the testing on the Yummly28K recipe dataset, and from the visualizations in Figure 7.8, we can see how the distributions of the Indian recipes dataset and the Yummly28K dataset overlap in contrast to the distributions of the Indian recipe dataset and the Recipe1M, Epicurious, and Recipe box datasets (presented in Figure 7.10).
From Figure 7.9, we can see that the pattern of distribution of the Salad recipes matches the one of the Indian recipes but it overshadows it as the Salad recipes dataset is significantly larger (around 13 times larger) and its distribution is widely spread across the feature space, therefore a model trained on the Indian recipes dataset cannot cover the diversity of instances contained in the Salad recipes dataset.

- For the Recipe1M dataset, the highest generalizability index is achieved for the Epicurious recipe dataset, followed by the Recipe box dataset, and right after is the Yummly28K recipe dataset. From Figure 7.11, we can see how similar the distributions in the feature space of the three datasets are, and from Figure 7.12, we can observe why the Yummly28K dataset has a slightly lower generalizability index – there are two big chunks of instances from the dataset that are not covered by the distribution of the other three datasets.
  When comparing the number of instances of the Recipe1M dataset and the Yummly28K recipe dataset, we can see that the Recipe1M dataset is almost twice the size of the Yummly28K, meaning this happens because the instances

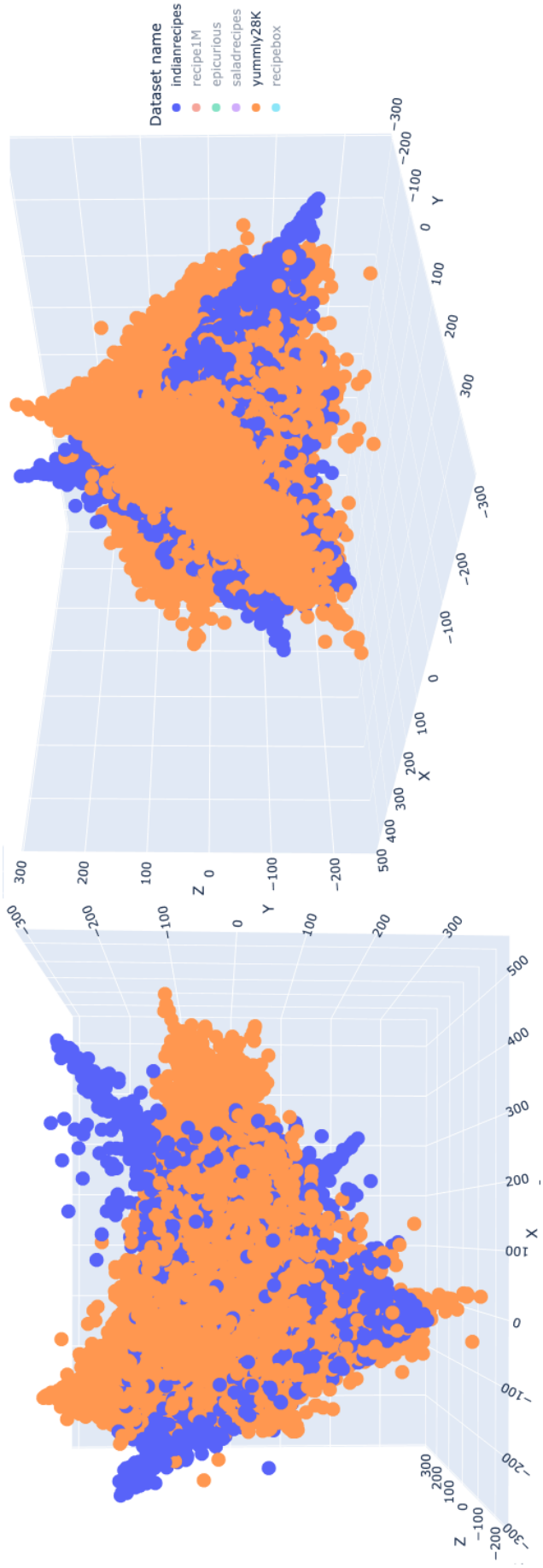

Figure 7.8: Distributions in the feature space of the Indian recipes dataset and the Yummly28K dataset.

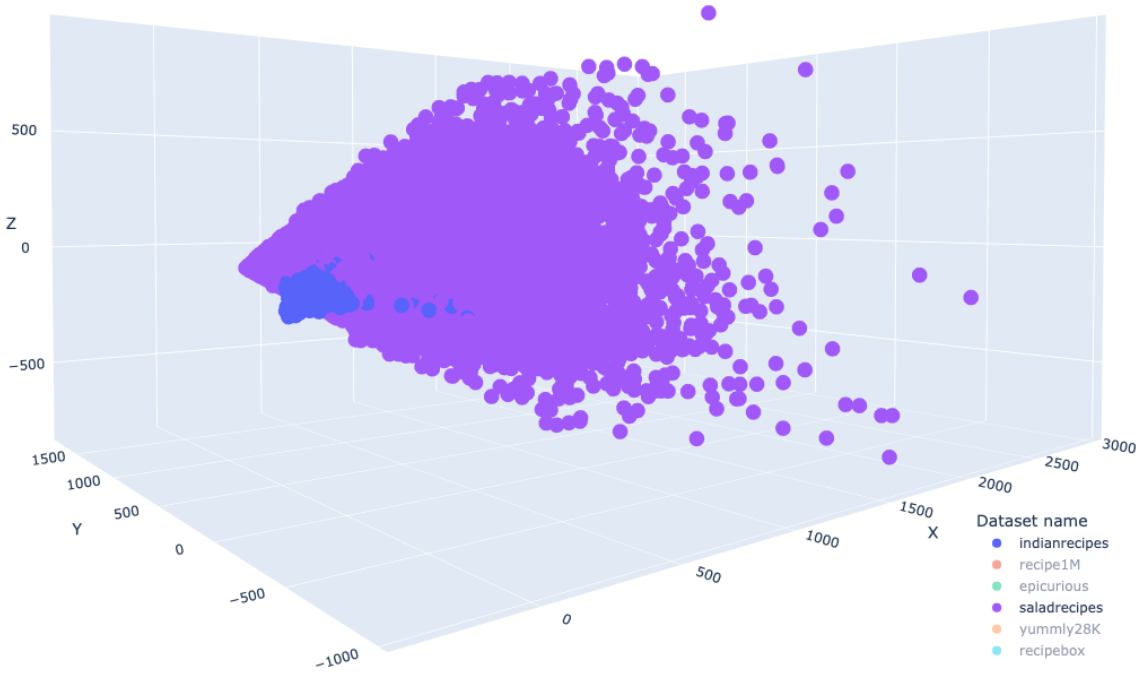

Figure 7.9: Distributions in the feature space of the Indian recipes dataset and the Salad recipes dataset.

of the Yummly28K recipe dataset are distributed more widely in the feature space, and the distribution of the instances of the Recipe1M dataset is more dense in the feature space. From the results presented in Table 7.3 we can observe that the models trained on the Recipe1M dataset, and tested on these three datasets (Epicurious, Yummly28K and Recipe box) yield much higher accuracies in contrast to the testing on the other two datasets (Indian recipes and Salad recipes).

- For the Epicurious recipe dataset we can observe that the highest generalizability index is obtained for the Recipe1M, and the second highest for the Recipe box dataset, which is expected from the distributions in the feature space presented in Figure 7.11. The Yummly28K recipe dataset, again, has yielded a lower generalizability index (Figure 7.12), which we can see also reflects the results presented in Table 7.3.
- For the Salad recipes dataset we can see that the results from calculating the generalizability indexes are quite different. Very low values are obtained for the Recipe1M and the Epicurious datasets, and for the Yummly28K and the Recipe box datasets slightly higher generalizability indexes are obtained. The highest generalizability index is obtained for the Indian recipes dataset, and from Figure 7.9, as observed previously, we can see that this happens because out of all of the datasets, the Salad recipes dataset has the most similar distribution with the Indian recipes dataset. This is also evident from the results presented in Table 7.3.
  Another point that can be made here is, if we observe the heat map presented in Figure 7.7, we can see that in Cluster 6, there are only instances from the Salad recipes dataset, which therefore cannot be captured from any model trained on the other five datasets (presented in Figure 7.13).
  An interesting observation from the heat map presented in Figure 7.7 is that

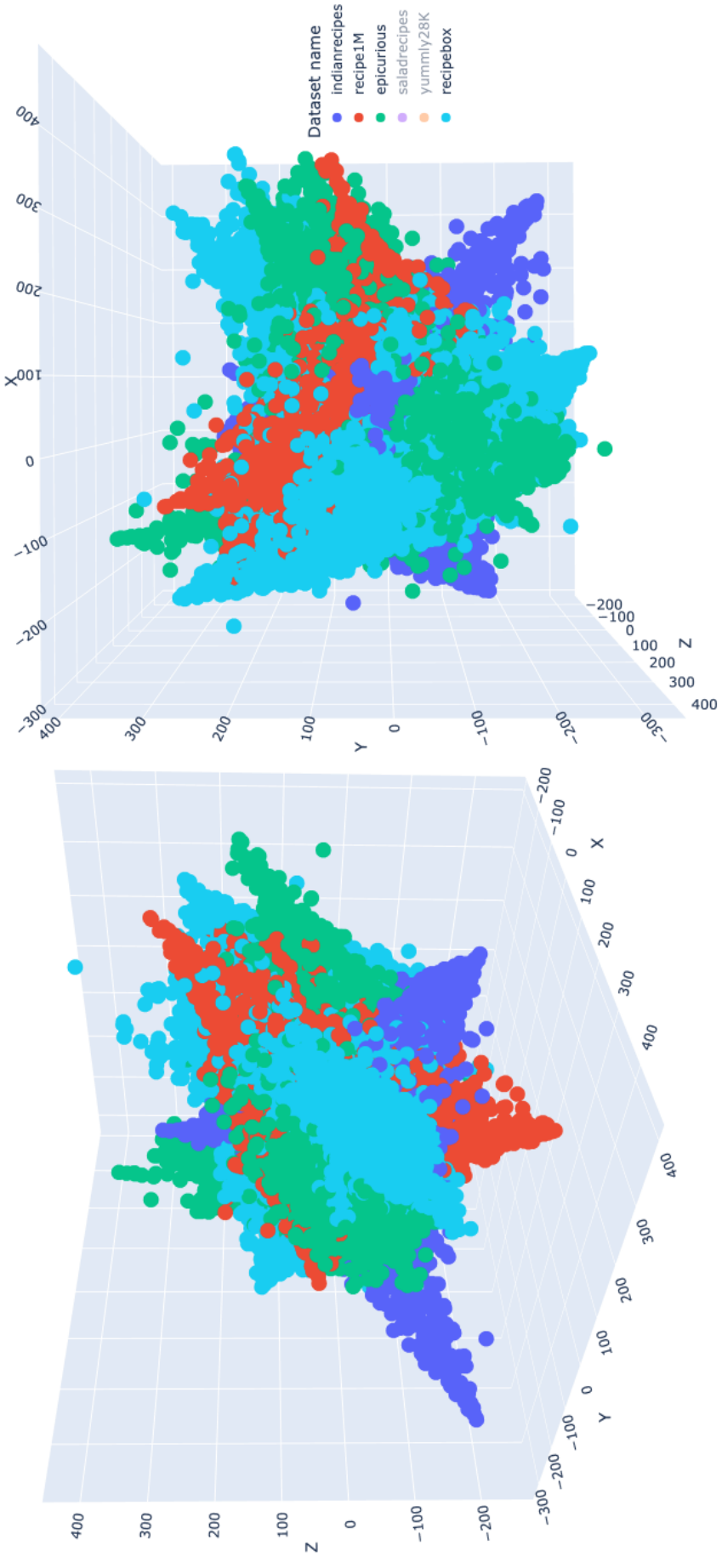

Figure 7.10: Distributions in the feature space of the Indian recipes, Recipe1M, Epicurious and Recipe box datasets.

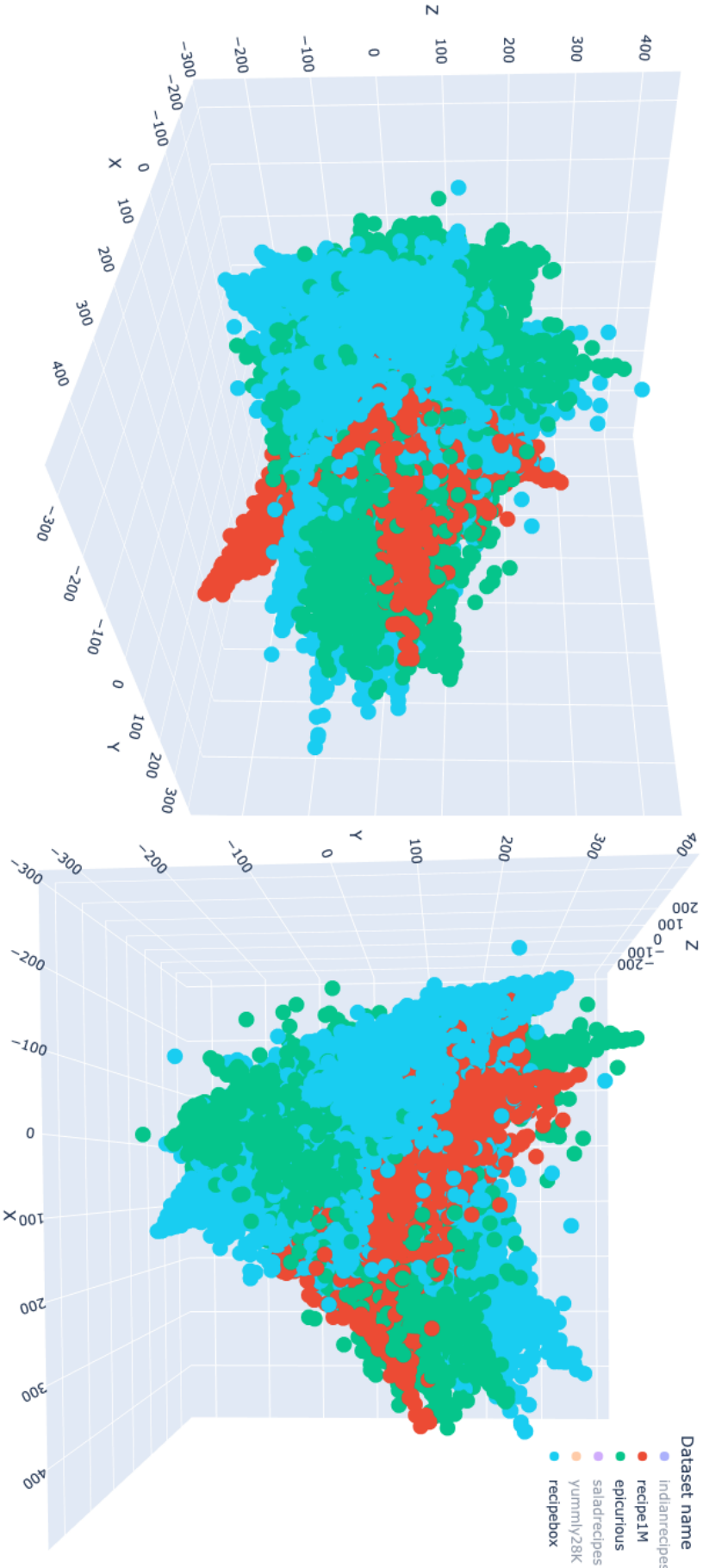

Figure 7.11: Distributions in the feature space of the Recipe1M, Epicurious and Recipe box datasets.

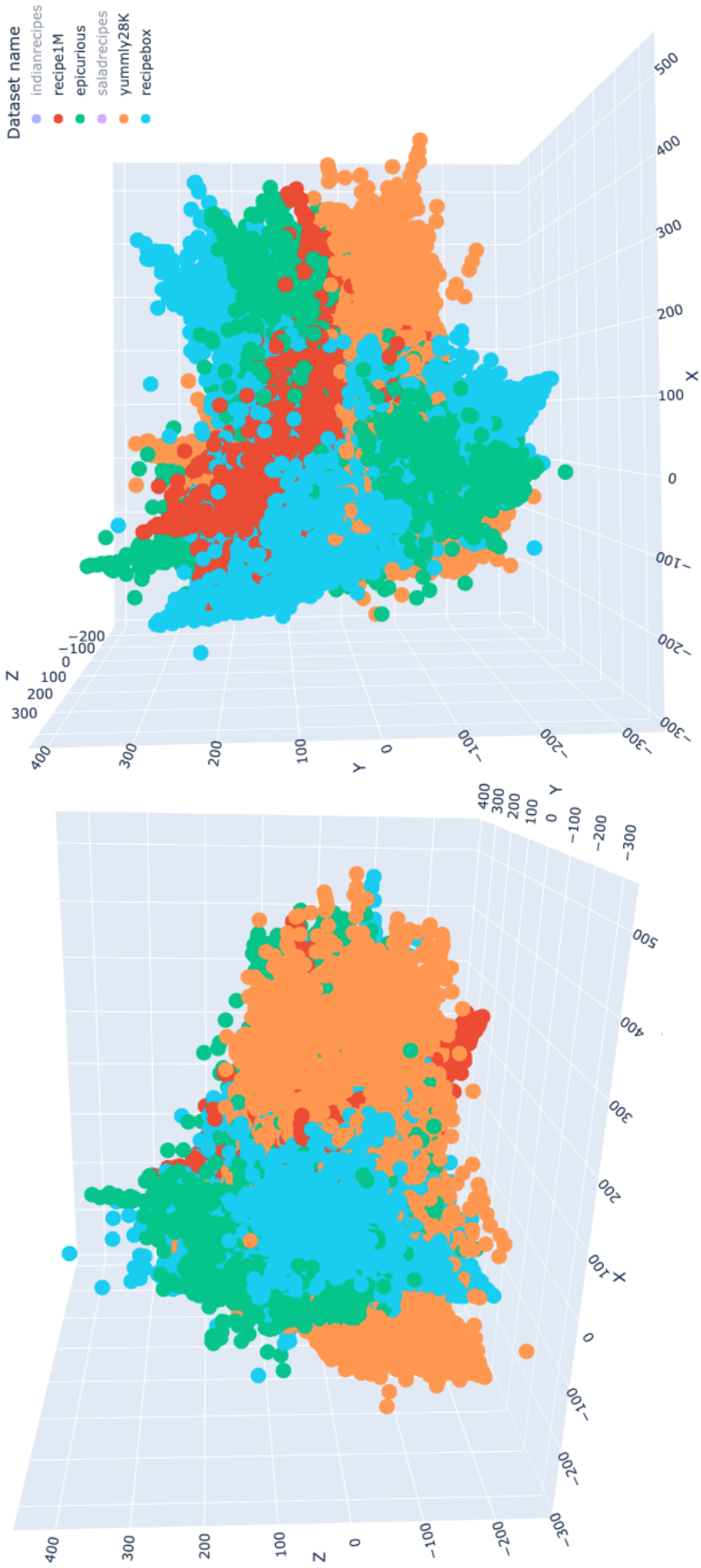

Figure 7.12: Distributions in the feature space of the Recipe1M, Epicurious, Recipe box and Yummly28K datasets.

Figure 7.13: Distributions in the feature space of the instances belonging to cluster number 6 and the instances from Indian recipes, Recipe1M, Epicurious, Yummly28K and Recipe box datasets.

Cluster 7 is densely populated by instances from the Recipe1M, Epicurious, Yummly28K and Recipe box datasets (above 30% from each dataset), while from the other two datasets – Indian recipes and Salad recipes, Cluster 7 has only 15.9% and 18.2%, respectively. These two observations go to show why the generalizability indexes of the Salad recipes dataset are so low for these four datasets, and why they are the highest for the Indian recipes dataset.

- For the Yummly28K recipe dataset, the highest generalizability index is obtained for the Recipe box dataset, followed by the Recipe1M dataset, the Epicurious and then the Indian recipe datasets. If we compare the distributions of instances in the feature space of the Yummly28K recipe dataset and those of the Recipe box, Recipe1M, Epicurious and Indian recipe datasets, we can see that all of these four datasets have overlapping with the Yummly28K recipe dataset, and the most interesting part here is that the generalizability indexes are sorted the same as the number of instances per dataset. Because the instances from the Yummly28K recipe dataset are distributed more far apart (i.e. widely) in the feature space, the generalizability index is the highest for the Recipe box dataset, because it has around 82,000 instances, therefore it is more densely distributed and the Yummly28K dataset covers more "area" from the feature space and therefore more instances can be represented.
  Then, the Recipe1M generalizability index is next, because the Recipe1M dataset has around 55,000 instances, again much more densely distributed compared to the Yummly28K distribution in the feature space. Then follow the generalizability indexes of 0.66 and 0.61 for the Epicurious dataset (with around 21,000 instances) and the Indian recipes dataset (with around 6,700 instances). From the presented accuracies in Table 7.3 we can observe how the above-made statements, i.e., how the obtained generalizability indexes correspond with the performance of the predictive models – a higher generalizability index implies a higher accuracy.
- For the Recipe box dataset, the highest generalizability index is achieved for the Recipe1M dataset and it is very comparable with the generalizability index for the Epicurious recipe dataset, and right after is the generalizability index for the Yummly28K recipe dataset. The results obtained from the predictive models trained on the Recipe box dataset and tested on the other five recipe datasets are presented in Table 7.3. From the presented accuracies we can observe the reliability of the generalizability indexes – the predictive models trained on the Recipe box dataset performed best when tested on the Recipe1M dataset, and second best on the Epicurious recipe dataset.

The results for all six datasets obtained with the Word2Vec embeddings are presented in Table 7.3. The rest of the results are given in Appendix B.

7. Create a generalized dataset for training – we build a generalized dataset for training a predictive model by selecting representative instances from each cluster.
   The last step is to create a generalized training dataset. The main idea is to select representative instances from each cluster, and the main question is how do we go about doing this. We choose to analyze three different sampling techniques for sampling the representative instances from each cluster:

   (a) First, we decided to select 10%, 20% and 30% random instances from each cluster (the number of instances from each cluster for each percentage is presented

Table 7.3: Results from the predictive models trained on each one of the recipe datasets separately and tested on the rest obtained with the Word2Vec embeddings merged with the domain heuristic. *Max* is the maximum accuracy obtained, and *Average* is the average accuracy obtained.

Train dataset	**Test dataset**	**Target**									
		Fat		**Protein**		**Saturated fat**		**Sugars**		**Sodium**	
		Max	Average	Max	Average	Max	Average	Max	Average	Max	Average
Indian recipes	Recipe1M	32.36	26.22	25.39	19.8	70.3	61.58	28.62	18.35	35.98	32.62
	Epicurious	31.53	26.22	26.06	20.76	72.72	65.31	31.85	19.63	37.69	30.77
	Salad recipes	3.32	3.02	22.72	13.53	45.81	29.05	12.54	4.89	24.25	21.57
	Yummly28K	47.01	41.16	43.76	38.43	89.95	79.01	45.52	33.64	26.28	22.27
	Recipe box	36.34	31.06	28.93	23.75	72.64	65.76	27.51	16.96	34.06	26.64
Recipe1M	Indian recipes	45.72	25.36	56.52	53.99	72.49	55.97	37.95	21.67	42.91	36.23
	Epicurious	64.44	44.44	81.28	77.49	93.83	78.47	60.38	42.63	65.93	53.41
	Salad recipes	17.32	14.21	53.83	47.41	46.26	23.38	12.44	12.10	27.90	22.49
	Yummly28K	36.62	17.60	47.55	42.04	67.51	47.20	32.06	15.33	15.61	4.78
	Recipe box	45.44	27.69	65.21	60.83	72.19	57.29	36.54	19.74	44.21	29.82
Epicurious	Indian recipes	59.83	56.07	57.81	55.29	86.52	84.92	61.38	51.69	95.83	83.99
	Recipe1M	78.29	74.53	82.67	80.48	83.91	82.63	78.54	70.68	95.76	86.87
	Salad recipes	29.24	20.47	58.83	52.69	61.54	57.23	41.68	27.79	88.98	79.53
	Yummly28K	54.67	48.08	48.36	44.49	85.16	82.67	57.59	45.00	93.81	76.05
	Recipe	57.56	54.41	67.39	65.28	83.86	82.47	52.55	44.94	96.28	89.06
Salad recipes	Indian recipes	74.82	72.87	83.51	81.42	90.11	88.37	69.83	67.85	99.87	99.87
	Recipe1M	34.0	31.59	46.14	35.84	51.98	50.96	33.51	30.13	67.8	67.68
	Epicurious	32.05	28.3	41.99	33.02	54.56	53.69	34.42	29.65	67.93	67.85
	Yummly28K	32.02	26.79	25.81	18.74	56.23	55.32	33.13	24.78	68.03	67.29
	Recipe box	30.14	25.88	30.14	22.58	55.72	54.88	28.39	18.47	68.07	67.19
Yummly28K	Indian recipes	60.86	56.62	65.70	64.74	88.59	86.48	68.33	65.37	99.12	92.37
	Recipe1M	58.25	54.11	68.25	67.26	84.34	82.77	64.67	62.56	97.91	93.42
	Epicurious	59.56	55.23	67.50	66.66	86.02	84.90	66.36	63.93	98.05	94.62
	Salad recipes	30.42	21.30	42.36	40.21	68.40	64.10	50.35	43.95	98.38	84.88
	Recipe box	78.67	74.99	86.97	86.19	86.6	85.24	83.4	81.24	98.08	95.26
Recipe box	Indian recipes	40.28	35.87	58.55	54.78	89.20	86.25	68.99	66.42	98.99	89.71
	Recipe1M	63.39	56.87	83.81	80.53	86.33	83.86	86.62	84.38	98.12	91.19
	Epicurious	41.45	35.20	67.13	64.25	87.69	85.15	66.34	64.41	98.2	92.59
	Salad recipes	3.95	2.37	61.88	53.82	63.82	58.43	47.72	42.82	98.55	94.83
	Yummly28K	25.10	21.64	47.92	42.69	88.76	84.32	71.51	67.20	98.55	85.31

Table 7.4: Number of instances from each cluster in the generalized dataset.

Cluster	Number of instances		
	10% of the cluster	**20/% of the cluster**	**30/% of the cluster**
0	4,457	8,915	13,372
1	963	1,873	2,836
2	680	1,359	2,039
3	2,475	4,950	7,425
4	738	1,477	2,215
5	3,227	6,455	9,732
6	196	393	589
7	6,547	13,095	19,642

in Table 7.4). These percentages are selected as more than 30% will lead to potential over-fitting of the predictive models. We repeat these selections several times, each time resulting in different samples. Then, predictive models are trained, with each of these datasets as the training dataset.
When testing the models on the rest of the instances, not included in the training process, we did not observe an increase, more so a slight decline in the accuracy from the percentages presented in Table 7.3. For example, the average accuracy presented in this table of predictive models for fat is between 2.37% and 74.99%, and when training on these generalized datasets, the average accuracies were up no more than 50%.

(b) Second, in order to investigate the sensitivity of sampling the representative instances from each cluster, we identified the cluster centroid (the imaginary or real location representing the center of the cluster) for each cluster, then set a percentage to choose instances closest to the centroid that are going to be used as the representatives. We repeat the same steps as before – select 10%, 20%, and 30% of the instances in each cluster, this time closest to the centroid of that cluster. The distance between each instance and the cluster is calculated with calculating the cosine distance between the reduced embeddings of the instance and the centroid. This cosine distance is calculated as presented in Equation (7.3).

$$cos_{distance} = 1 - cos_{similarity} \tag{7.3}$$

Where cosine similarity is calculated with:

$$cos_{similarity} = \cos(\theta) = \frac{Point \cdot Centroid}{\|Point\|_2 \, \|Centroid\|_2} \tag{7.4}$$

With θ being the angle between the vectors *Point* and *Centroid*, where *Point* is the vector from $(0, 0, 0)$ to the coordinates of our point of interest $P(x_p, y_p, z_p)$ and *Centroid* is the vector from $(0, 0, 0)$ to the coordinates of the centroid in question $C(x_c, y_c, z_c)$. The selection is done in a way that, for example, when we are choosing let us say 30% of the cluster around the centroid in some clusters where the cluster is larger or the instances are populated densely around the cluster, the 30% will be selected in a much smaller area of the cluster. As we are trying to select only the instances closest to the cluster centroid, this is a one time procedure. After the selection, predictive models are trained on a training set consisting of the selected representative instances from each cluster. The

Table 7.5: Results from the evaluation on the models trained when using the generalized training dataset obtained when using the instances closest to the centroids from each cluster.

Generalized training dataset	Target	Average Accuracy
10% closest to centroid of each cluster	Fat	71.23
	Protein	79.39
	Saturated fat	67.58
	Sugars	64.23
	Sodium	70.12
20% closest to centroid of each cluster	Fat	75.63
	Protein	83.82
	Saturated fat	73.54
	Sugars	79.84
	Sodium	76.99
30% closest to centroid of each cluster	Fat	78.76
	Protein	87.45
	Saturated fat	77.12
	Sugars	83.36
	Sodium	80.93

obtained accuracies are presented in Table 7.5, where the average accuracy is calculated over the obtained accuracies from all different regressions mentioned earlier. We can observe that there is an increase in the average accuracies, if we compare them to the accuracies presented in Table 7.3.

(c) At last, we decided to select the representative instances in one more way, very similar to the previous, with the difference that we are only focusing on a determined ϵ neighbourhood from the centroid of a cluster. Meaning the instances that will represent a cluster are selected within a ϵ neighborhood of the centroid. First, the ϵ neighbourhood is determined as follows:

- Calculate the cosine distance (defined in Equations (7.3), and (7.4)) from all points in the cluster to the centroid point.
- Determine the ϵ value for the ϵ neighbourhood – we calculate the maximum and minimum $cosine_{distance}$ from each cluster's centroid to an instance belonging to the set cluster (presented in Table 7.6).

Table 7.6: Maximum $cosine_{distance}$ from each cluster's centroid to an instance belonging to the same cluster.

Cluster number	0	1	2	3	4	5	6	7
Maximum $cos_{distance}$	0.422	0.186	0.507	0.184	0.244	0.0.367	0.180	0.536

The lowest maximum $cos_{distance}$ is obtained for Cluster 6 (0.180), meaning the instance that is furthest from the centroid of the cluster is within a $cos_{distance} = 0.180$. This cluster is also the cluster with the lowest number of instances – 1,966. The goal is to select the same number of instances from each cluster within the same ϵ neighbourhood, and if we select the $\epsilon = 0.180$, we would have to select 1,966 instances from each cluster, and we would end up with a training dataset of 15,728 instances, which is only 7.16% from

the original corpus. This percentage is quite low for a training dataset. If we try this with $\epsilon = 0.184$, which is the next lowest $cos_{distance}$ (obtained for Cluster 4), the number of instances to select from each cluster would be 7,387, meaning the training dataset would consist of 53,675 instances, which is 24% from the recipe embeddings corpus. At the end, we decided to select the next highest $cos_{distance}$ from Table 7.6, 0.186, meaning for the clusters with maximum $cos_{distance} \leq 0.186$ all the instances from the cluster will be selected. Those clusters are: Cluster 1, Cluster 4 and Cluster 6. The number of instances from each cluster with $cos_{distance} \leq 0.186$ is given in Table 7.7. The final training dataset selected in this way has 65,531 instances, which is 29.82% of the whole recipe embeddings corpus consisting of the six recipe datasets.

Table 7.7: Number of instances from each cluster for $\epsilon \leq 0.186$.

Cluster number	0	1	2	3	4	5	6	7
Number of instances	38,534	9,361	29,475	24,751	7,346	28,961	1,966	47,197

- Select all the instances from the cluster that have a $cosine_{distance} \leq \epsilon$ with the centroid.– After determining the ϵ neighbourhood for the selections of the representative instances from each cluster, for the clusters containing more than 9,363 instances, we do the selection randomly (repeated 3 times), and for the Cluster 1, Cluster 4 and Cluster 6 where this is not the case (as they have 9,363 instances or less), we select all the instances from the designated ϵ neighbourhood. When the representative instances are selected from each cluster, the predictive modeling is then repeated for all possible combinations of the selections. The predictive models are then tested for all the remaining instances from the clusters. For comparison we decided to conduct the same experiments with the three lowest maximum cosine distances: 0.180, 0.184, and 0.186. The results are presented in Table 7.8. We can see that there is an improvement in accuracies for all target variables for $\epsilon = 0.186$ in comparison to the previous two ways of choosing the training dataset. We can also see how the accuracies improved when choosing $\epsilon = 0.186$ rather than 0.184 or 0.180, which is due to the fact of the very low samples included in the training dataset if 0.814 or 0.180 are chosen as the ϵ value.

7.4.1 Discussion

By generating domain-specific recipe embeddings for six different heterogeneous recipe datasets (previously harmonized and put in the same format with data normalization and data mapping) with the same dataset of ingredient embeddings, we brought them to the same feature space. Reducing the high-dimensional vectors of the embeddings to three-dimensions allowed us to see their distributions in the feature space and observe which of these datasets have similar distributions. Then, with the process of unsupervised ML – clustering, we separated the feature space into eight different clusters. Observing the distributions of the clusters, we could see how different datasets "behaved" across clusters, and how some were similar in this "behaviour". Having this, we proceeded with defining a generalizability index, and a generalizability matrix for all six datasets that indicated the generalizability of a predictive model i.e. how successful will be the transferring a predictive model learned on one dataset to the other. Although the Salad recipes dataset is

Table 7.8: Results from the evaluation on the models trained when using the generalized training dataset obtained when using the instances in a defined ϵ neighbourhood of the centroids from each cluster.

Generalized training dataset	Target	Average Accuracy
$\epsilon = 0.180$	Fat	44.23
	Protein	55.73
	Saturated fat	53.82
	Sugars	57.31
	Sodium	60.13
$\epsilon = 0.184$	Fat	85.17
	Protein	81.34
	Saturated fat	77.22
	Sugars	82.91
	Sodium	84.34
$\epsilon = 0.186$	Fat	88.24
	Protein	96.53
	Saturated fat	83.41
	Sugars	91.32
	Sodium	93.45

the largest and has the highest number of instances, it scored low generalizability indexes due to its distribution in the feature space and in the clusters. While the other datasets, specifically the Epicurious, Recipe1M, Yummly28K, and Recipe box, scored high amongst each other due to their similar distributions in the feature space. These two statements can be directly seen thought the results from the predictive modeling and evaluation of the ML pipeline, presented in Table 7.3.

We introduce three distinctive techniques for choosing a representative training dataset. Initially, instances from each cluster were randomly chosen (the process was repeated three times for three different percentages from each cluster), and then each cluster's centroid is identified, and the same three percentages of instances from each cluster were again chosen, but this time only around the centroid (closest to the centroid according to the cosine distance). The third choice is also in a predetermined area – ϵ neighbourhood around each centroid.

After the selection of the representative training dataset we train predictive models on the chosen training dataset and test them on the rest of the instances not included in this dataset. The results from these evaluations help to clarify the importance of the choice of the training dataset in predictive modeling. This methodology is not of closed nature, new data can be introduced in the process. The steps for incorporating a new recipe dataset are:

1. Obtain the recipe embeddings for the instances using the unified ingredient embeddings corpus.

2. Calculate the cosine distances for each instance to each of the centroids.

3. Assign each instance to the closest cluster based on the cosine distances to the centroids.

4. Calculate the distribution of instances across clusters for the dataset.
5. Calculate the generalizability indexes of the dataset for all other datasets.
6. Select the representative instances for the training dataset according to the set ϵ neighbourhood.

Overall, from the results from this study we can conclude that representative datasets are needed in the training process of predictive modeling. The selection of the training dataset is a sensitive procedure that should be looked at from different aspects.

Chapter 8

Conclusions

As the last chapter, this chapter includes the scientific contributions of this thesis, if the research hypotheses are confirmed and directions for future work.

8.1 Research Hypotheses and Their Confirmation

This thesis had three main hypotheses. Below, each hypothesis is presented together with its scientific contribution:

H1. Including the domain knowledge into different stages of data-driven modeling improves the performance in a prediction task.

- This is the main hypothesis and it was proven as a cumulative outcome from results of the methodologies presented in Chapters 3, 4, and 5. We included domain knowledge in all stages of the initially presented ML/DM pipeline outlined in Chapter 3:
  - In the stage of prepossessing – domain knowledge was embedded in the form of domain dictionaries, databases, and rules in order to obtain the required information of raw text and structure the datasets. This was shown in Chapter 6. The publication related to this work is [44].
  - In the first stage, i.e., the RL part of the pipeline – domain knowledge was embedded when generating the final vectors which were then used as input features for the next parts. The domain knowledge in question here is a heuristic for fusing the multi-word vectors. This was shown in Chapter 5. The publication related to this work is [41].
  - In the second stage, i.e., unsupervised ML – domain knowledge was embedded in the unsupervised ML part of the pipeline in two different ways, using two different external semantic resources. This was shown in Chapter 3 and Chapter 4. The publications related to this work are [39] and [40].
  - In the third stage, i.e., supervised ML – domain knowledge was embedded in the evaluation step. We included tolerance levels suggested and defined by The European Commission Health and Consumers Directorate General in 2012 and published in [46], when defining the evaluation of the results from the ML/DM pipeline with and without the incorporation of the domain knowledge.

H2. Handling the domain knowledge bias can improve the results of a predictive task. To prove this hypothesis we adhered to two methodologies:

H2.a. Incorporating different semantic information about the domain knowledge in the modeling process has a significant impact on the prediction task and has the potential to enhance prediction outcomes.

- This hypothesis was proven in Chapter 4 where domain knowledge was embedded with two different external semantic resources, and this showed improvement in the prediction results. This work is published in [40].

H2.b. Fusion of multi-word representations with a domain knowledge-based heuristic provides representations (i.e., embeddings) that improve the results from a prediction task.

- This hypothesis was proven in Chapter 5. We presented an extension of the ML/DM pipeline – a newly proposed RL pipeline which incorporates domain knowledge as a heuristic for fusing the multi-word vectors, which resulted in improving the prediction results. This work is published in [41].

H3. Integrating domain knowledge into data-driven modeling generalizes the predictive models and allows generalization of the knowledge over food and nutrition data that come from different sources.

- This hypothesis was proven in Chapter 7. We presented a pipeline for generalizing predictive models. We defined an index called generalizability index, which is an indicator of how well will a predictive model learned on one dataset perform on another. At the end, we constructed a generalized training dataset with the goal of it consisting of the most representative samples of the data. The publications related to this work are [48] and [45].

8.2 Final Conclusions and Future Work

Being both a poison and a cure for many lifestyle and non-communicable diseases, food is inscribing itself into the prime focus of precise medicine. The monitoring of few groups of nutrients is crucial for some patients, and methods for easing their calculations are emerging. With new goods and formulas being consistently launched into the market while other items are being discontinued, the food supply is continually changing and evolving. Ingredients and agricultural raw materials are likewise subject to frequent change. Furthermore, the current understanding of the relationship between nutrition, health, and wellness is expanding beyond the focus on specific nutrients – take into account specific foods, food groupings, and dietary patterns.

Guided from the huge interest in the Food and Nutrition domain in the scientific community over the last decade, and especially the last few years reaching its peak with the COVID 19 pandemic, with the idea that domain knowledge and expertise has a huge impact on understanding the data and getting the most out of it, we decided to follow the

idea of researching the question on how domain knowledge from the Food and Nutrition domain impacts a prediction task.

We constructed a ML/DM pipeline (described in Chapter 3), and incorporated domain knowledge in different ways in every stage of the pipeline. Domain knowledge in the form of external semantic resources was incorporated in two ways. First, for clustering the datasets in order to obtain similar groups of recipes – with the FoodEx2 classification system and with the FSA traffic light system (shown in Chapter 3 and Chapter 4). Second, for data normalization of six heterogeneous multilingual recipe datasets and data mapping to the USDA FCDB in order to obtain two predefined corpora of domain-specific embeddings – one with ingredient and one with recipe embeddings (presented in Chapter 6). The data normalization process involved dictionary-based NER with four newly composed domain dictionaries (a dictionary for units of measurement, for converting units of measurement, for names of branded foods, and a dictionary for redundant words specifically for recipes), constructed with and approved by a domain expert. While the data mapping process as an ensemble of multiple approaches involved the USDA FCDB as an external resource as well as the four domain-specific dictionaries. By generating the recipe embeddings of the six heterogeneous recipe datasets with the same ingredient dataset, the recipes are brought to the same feature space. This contribution is important due to the fact that these datasets are the largest predefined corpora of their kind and the first to incorporate domain knowledge of this kind.

The other way of incorporating domain knowledge is through defining a heuristic for fusing multi-word representations (presented in Chapter 5) so that two instances – recipes that have the same ingredient list but different nutritional values, can be mapped to two different points in the feature space, and therefore make it logical for the ML algorithm to map them to two different points in the performance space.

Generalizability of predictive models is one of the main open questions in ML today. With the corpus of predefined recipe embeddings and the ML/DM pipeline we researched the question of how generalizable are the ML models for predicting nutrient values (presented in Chapter 7). Observing the different distributions of the reduced high-dimensional vectors of the recipe embeddings from each of the six heterogeneous recipe datasets, and their distribution across the clusters obtained with a clustering approach, we were able to define a generalizability index. This index indicates how will the transferring of a predictive model learnt on one of the recipe datasets perform to another. From the evaluation of the ML/DM pipeline on the six recipe datasets, by training predictive models on one of the recipe datasets, and testing the models on the rest of the recipe datasets separately, we prove that the defined generalizability index is consistent with the results. Consistency in this sense means – the higher the generalizability index, the higher the predictive accuracies.

At the end, we elucidate the sensitivity of the selection of the training dataset in predictive modeling by introducing three different ways of selecting a representative training dataset. First, by random selection of instances from each cluster (repeating the procedure three times for three different percentages from each cluster – 10%, 20% and 30%). Second, by determining the centroids of each cluster and selecting the same three percentages (10%, 20% and 30%) of instances from each cluster but this time selecting the set percentage of instances closest to the centroid. The closeness being indicated by the cosine distance of each instance to the centroid, meaning when selecting 10% of the instances we select 10% of the number of instances in that cluster that have the smallest cosine distance to the

centroid. And the third selection is in a determined ϵ neighbourhood around each centroid. The ϵ neighbourhood consists of instances that have $cosine_{distance} \leq \epsilon$ to the centroid of the cluster.

The application of the presented ML/DM pipeline opens up many possibilities for facilitating and easing the process of calculating nutrient content, which is crucial for many professionals, such as: dietary assessment, dietary recommendations, dietary guidelines, nutrient tracking, and other such tasks which are key tools for doctors, health professionals, dietitians, nutritional experts, policy makers, professional sport coaches, athletes, fitness professionals, etc. The results from all the studies included in this thesis revealed the significant difference that any domain insight provides in the prediction results. When dealing with data of any specific field, the fusion of domain and data-driven knowledge is crucial for making performant vector representations. Having a better prior understanding of the problem at hand, the domain is a key factor when dealing with a prediction task, and domain knowledge is the single, most important step in predictive modeling. This knowledge is very useful for European projects that are on-going like: FNS-Cloud [196], Metrofoods [197], FishEUTrust [198], COMFOCUS [199], and new ones yet to come.

This is in line with the current focus on data-centric AI. Many domains have already built and trained foundation models that are very large models, trained on extensive amount of data, and can be later tuned for specific applications (in the NLP community such model is Generative Pre-trained Transformer 3 i.e. GPT - 3 [200], which is an autoregressive language model that uses deep learning to produce human-like text). Foundation models as a new approach of developing ML applications have a lot of potential, but also exhibit some flaws and challenges, and present issues when it comes to ensuring that they are limiting the influence of the bias and making them reasonably fairly distributed.

One of the most prevalent problems in many domains is the lack of data, and especially data that can be used for ML. The biggest issue in this challenge is that big data does not exist in some domains and industries. Shifting the focus from obtaining as much data as possible to obtaining good quality data can be revolutionary and change the course of ML applications. The old saying: "The bigger the better" and the motto of "scaling up" seems to be losing its hype and influence when it comes to data for predictive modeling. Researchers are paving the way towards data-centric AI slowly but surely.

Inconsistency in ML data is a very persisting issue – even in data provided and/or annotated by domain experts, where it can happen that even trained human annotators do not agree on the label that should be assigned. This not only makes the data inconsistent, but prevails issues in the training process that can be reflected when introducing new data for prediction. Specifically for data that has a lot of inconsistencies and is noisy, if we go by the rule "more is better" and collect a lot of data so the algorithm in the training process can average over the noisy data, we are in a way damaging the good quality data present in the dataset. This problem can be solved with a more elegant and efficient solution, which is improving the data consistency.

As we have shown data has a major impact on model performance, perhaps even more so than model performance itself. Data-centric AI is focused on methodically iterating over data to improve its quality and performance. Even when models are put to production, the process of gathering, annotating, and preparing training data continues. One thing left to point out is that data-centric AI is one piece of the solution, but not the entire solution

to the problem of the bias, over-fitted predictive modeling.

Appendix A

Results from Chapter 6

In this Appendix, we provide the results obtained from the predictive modeling with the recipe embeddings obtained with and without using the domain heuristic on all six recipe datasets described in Chapter 6. The results obtained with the domain heuristic are provided in the following tables:

1. For the Recipe1M dataset in Table A.1.
2. For the Indian recipes dataset in Table A.2.
3. For the Epicuripus dataset in Table A.3.
4. For the Salad recipes dataset in Table A.4.
5. For the Yummly28K dataset in Table A.5.
6. For the Recipe box dataset in Table A.6.

The results obtained with the domain heuristic are provided in the following tables:

1. For the Recipe1M dataset are given in Table A.7.
2. For the Indian recipes dataset in Table A.8.
3. For the Epicurious dataset in Table A.9.
4. For the Salad recipes dataset in Table A.10.
5. For the Yummly28K dataset in Table A.11.
6. For the Recipe box dataset in Table A.12.

Table A.1: Results for the **Recipe1M** dataset obtained with the embeddings merged with the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	87.75	67.86	7.86	7.48
	Protein	91.70	86.22	27.14	22.89
	Saturated fat	92.17	85.35	7.27	8.78
	Sugars	88.29	60.76	6.96	13.47
	Sodium	92.73	91.00	25.42	43.58
GloVe	Fat	79.69	62.96	7.84	7.53
	Protein	90.73	83.95	27.06	22.89
	Saturated fat	91.34	81.69	7.27	8.78
	Sugars	81.98	61.50	6.94	13.47
	Sodium	92.69	90.50	25.39	43.57
Doc2Vec	Fat	82.19	62.53	7.84	7.53
	Protein	90.94	82.60	27.06	22.89
	Saturated fat	91.68	81.90	7.27	8.78
	Sugars	81.28	55.62	6.94	13.47
	Sodium	91.86	89.60	25.39	43.57
BERT	Fat	92.99	77.64	7.84	7.53
	Protein	93.00	89.32	27.06	22.89
	Saturated fat	93.00	90.95	7.27	8.78
	Sugars	92.99	85.64	6.94	13.47
	Sodium	93.00	92.06	25.39	43.57

Table A.2: Results for the **Indian recipes** dataset obtained with the embeddings merged with the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	81.17	67.77	22.52	21.14
	Protein	87.89	81.40	43.07	29.82
	Saturated fat	89.67	85.85	15.92	27.36
	Sugars	76.91	59.88	8.69	21.53
	Sodium	92.96	92.88	95.63	47.01
GloVe	Fat	79.99	67.14	22.52	21.14
	Protein	86.42	81.07	43.07	29.82
	Saturated fat	89.36	85.56	15.92	27.36
	Sugars	75.49	59.44	8.69	21.53
	Sodium	92.91	92.88	95.63	47.01
Doc2Vec	Fat	79.53	66.08	22.52	21.14
	Protein	89.08	83.16	43.07	29.82
	Saturated fat	90.38	85.02	15.92	27.36
	Sugars	77.43	60.98	8.69	21.53
	Sodium	92.88	92.72	95.63	47.01
BERT	Fat	86.28	74.42	22.52	21.14
	Protein	89.56	86.61	43.07	29.82
	Saturated fat	90.96	87.69	15.92	27.36
	Sugars	86.37	75.86	8.69	21.53
	Sodium	92.88	92.78	95.63	47.01

Table A.3: Results for the **Epicurious** recipe dataset obtained with the embeddings merged with the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	65.54	54.04	19.22	12.89
	Protein	84.96	78.84	24.73	16.48
	Saturated fat	88.80	86.05	18.92	16.03
	Sugars	72.35	61.96	7.96	17.52
	Sodium	91.75	90.52	43.66	44.99
GloVe	Fat	64.48	52.77	19.10	12.89
	Protein	83.58	75.88	24.71	16.48
	Saturated fat	88.87	86.03	18.90	16.04
	Sugars	68.7	55.75	7.95	17.57
	Sodium	91.7	91.12	43.61	45.0
Doc2Vec	Fat	60.87	47.12	19.10	12.89
	Protein	82.56	66.66	24.71	16.48
	Saturated fat	88.12	84.34	18.90	16.04
	Sugars	64.84	45.00	7.95	17.57
	Sodium	91.70	90.56	43.61	45.0
BERT	Fat	67.54	59.32	19.10	12.89
	Protein	82.87	79.82	24.71	16.48
	Saturated fat	88.49	86.93	18.9	16.04
	Sugars	67.28	60.85	7.95	17.57
	Sodium	91.62	89.98	43.61	45.00

Table A.4: Results for the **Salad recipes** dataset obtained with the embeddings merged with the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	63.30	44.98	17.25	12.33
	Protein	83.12	69.79	21.91	16.00
	Saturated fat	87.73	83.81	16.48	13.22
	Sugars	70.35	50.38	8.75	17.55
	Sodium	91.29	88.06	22.39	44.85
GloVe	Fat	62.16	44.02	17.25	12.33
	Protein	82.41	66.96	21.91	16.0
	Saturated fat	87.66	83.46	16.48	13.22
	Sugars	66.13	45.67	8.75	17.55
	Sodium	91.14	89.95	22.39	44.85
Doc2Vec	Fat	56.80	40.56	17.25	12.33
	Protein	77.74	59.23	21.91	16.0
	Saturated fat	86.24	82.44	16.48	13.22
	Sugars	56.74	38.43	8.75	17.55
	Sodium	91.14	90.76	22.39	44.85
BERT	Fat	62.51	46.60	17.25	12.33
	Protein	82.98	70.66	21.91	16.0
	Saturated fat	87.69	84.56	16.48	13.22
	Sugars	66.39	51.86	8.75	17.55
	Sodium	91.31	89.92	22.39	44.85

Table A.5: Results for the **Yummly28K** recipe dataset obtained with the embeddings merged with the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	63.13	53.47	19.44	13.81
	Protein	83.25	76.51	27.06	15.33
	Saturated fat	88.44	85.69	19.06	14.54
	Sugars	74.21	64.17	9.62	28.34
	Sodium	91.57	89.64	26.18	45.83
GloVe	Fat	61.95	51.60	19.40	13.82
	Protein	81.95	73.69	27.09	15.33
	Saturated fat	88.34	85.26	19.06	14.55
	Sugars	70.07	58.54	9.64	28.35
	Sodium	91.42	88.25	27.03	45.84
Doc2Vec	Fat	52.76	44.79	19.40	13.82
	Protein	73.42	60.52	27.09	15.33
	Saturated fat	85.92	83.54	19.06	14.55
	Sugars	59.06	44.60	9.64	28.35
	Sodium	91.40	87.26	27.03	45.84
BERT	Fat	65.02	57.36	19.40	13.82
	Protein	82.33	77.26	27.09	15.33
	Saturated fat	87.89	86.58	19.06	14.55
	Sugars	70.25	65.31	9.64	28.35
	Sodium	91.42	82.92	27.03	45.84

Table A.6: Results for the **Recipe box** dataset obtained with the embeddings merged with the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	71.27	59.27	18.61	11.08
	Protein	86.14	80.68	26.53	18.82
	Saturated fat	89.36	83.79	16.00	10.04
	Sugars	73.71	59.02	9.61	12.82
	Sodium	91.42	89.88	38.39	45.77
GloVe	Fat	67.90	56.51	18.61	11.08
	Protein	84.94	78.30	26.53	18.82
	Saturated fat	88.66	83.09	16.00	10.04
	Sugars	69.42	53.33	9.61	12.82
	Sodium	91.43	90.84	38.39	45.77
Doc2Vec	Fat	66.06	51.17	18.61	11.08
	Protein	81.63	70.94	26.53	18.82
	Saturated fat	85.72	80.41	16.00	10.04
	Sugars	59.36	42.91	9.61	12.82
	Sodium	91.37	90.13	38.39	45.77
BERT	Fat	78.16	67.43	18.61	11.08
	Protein	86.35	82.78	26.53	18.82
	Saturated fat	89.62	86.64	16.00	10.04
	Sugars	74.50	64.96	9.61	12.82
	Sodium	91.35	89.67	38.39	45.77

Table A.7: Results for the **Recipe1M** dataset obtained without the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	39.08	21.88	7.84	7.53
	Protein	69.50	62.17	27.06	22.89
	Saturated fat	63.10	49.69	7.27	8.78
	Sugars	41.79	25.73	6.94	13.47
	Sodium	81.49	81.14	25.39	43.57
GloVe	Fat	36.51	21.46	7.84	7.53
	Protein	68.59	60.91	27.06	22.89
	Saturated fat	61.20	49.54	7.27	8.78
	Sugars	42.90	26.13	6.94	13.47
	Sodium	81.43	81.06	25.39	43.57
Doc2Vec	Fat	37.08	20.57	7.84	7.53
	Protein	67.69	59.76	27.06	22.89
	Saturated fat	62.52	48.05	7.27	8.78
	Sugars	43.25	22.47	6.94	13.47
	Sodium	81.25	80.93	25.39	43.57
BERT	Fat	37.51	26.29	7.84	7.53
	Protein	70.35	63.57	27.06	22.89
	Saturated fat	62.63	55.08	7.27	8.78
	Sugars	50.28	33.44	6.94	13.47
	Sodium	81.48	81.19	25.39	43.57

Table A.8: Results for the **Indian recipes** dataset obtained without the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	56.82	47.82	22.52	21.14
	Protein	73.11	70.29	43.07	29.82
	Saturated fat	77.69	75.71	15.92	27.36
	Sugars	52.28	36.67	8.69	21.53
	Sodium	84.90	84.88	95.63	47.01
GloVe	Fat	55.58	47.04	22.52	21.14
	Protein	72.68	69.99	43.07	29.82
	Saturated fat	77.51	75.54	15.92	27.36
	Sugars	52.17	36.09	8.69	21.53
	Sodium	84.88	84.87	95.63	47.01
Doc2Vec	Fat	57.01	46.6	22.52	21.14
	Protein	73.00	69.87	43.07	29.82
	Saturated fat	77.64	75.24	15.92	27.36
	Sugars	52.59	32.71	8.69	21.53
	Sodium	84.88	84.88	95.63	47.01
BERT	Fat	58.07	48.00	22.52	21.14
	Protein	74.40	71.00	43.07	29.82
	Saturated fat	78.04	75.44	15.92	27.36
	Sugars	45.93	35.91	8.69	21.53
	Sodium	84.88	84.87	95.63	47.01

Table A.9: Results for the **Epicurious** dataset obtained without the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	40.75	33.06	19.22	12.89
	Protein	62.64	54.0	24.73	16.48
	Saturated fat	77.21	75.96	18.92	16.03
	Sugars	56.50	40.48	7.96	17.52
	Sodium	83.72	82.34	43.66	44.99
GloVe	Fat	40.64	33.25	19.10	12.89
	Protein	60.29	51.06	24.71	16.48
	Saturated fat	77.64	76.04	18.90	16.04
	Sugars	54.31	35.76	7.95	17.57
	Sodium	83.69	82.95	43.61	45.00
Doc2Vec	Fat	39.45	31.50	19.10	12.89
	Protein	58.53	48.36	24.71	16.48
	Saturated fat	77.20	75.66	18.90	16.04
	Sugars	52.72	30.50	7.95	17.57
	Sodium	83.69	82.84	43.61	45.00
BERT	Fat	38.41	31.71	19.10	12.89
	Protein	61.12	51.47	24.71	16.48
	Saturated fat	77.78	76.09	18.90	16.04
	Sugars	50.81	37.56	7.95	17.57
	Sodium	83.57	82.14	43.61	45.00

Table A.10: Results for the **Salad recipes** dataset obtained without the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	43.06	29.11	17.25	12.33
	Protein	60.81	49.84	21.91	16.00
	Saturated fat	75.39	74.12	16.48	13.22
	Sugars	52.79	34.99	8.75	17.55
	Sodium	83.16	76.01	22.39	44.85
GloVe	Fat	34.87	28.44	17.25	12.33
	Protein	57.28	46.39	21.91	16.00
	Saturated fat	75.52	74.30	16.48	13.22
	Sugars	50.11	32.49	8.75	17.55
	Sodium	83.17	82.09	22.39	44.85
Doc2Vec	Fat	33.56	26.21	17.25	12.33
	Protein	48.84	41.66	21.91	16.00
	Saturated fat	74.65	73.50	16.48	13.22
	Sugars	44.20	26.66	8.75	17.55
	Sodium	83.10	82.46	22.39	44.85
BERT	Fat	35.40	29.25	17.25	12.33
	Protein	56.71	48.90	21.91	16.00
	Saturated fat	75.81	74.69	16.48	13.22
	Sugars	48.81	33.20	8.75	17.55
	Sodium	83.18	80.54	22.39	44.85

Table A.11: Results for the **Yummly28K** recipes dataset obtained without the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	40.34	33.83	19.44	13.81
	Protein	60.10	51.85	27.06	15.33
	Saturated fat	77.25	75.8	19.06	14.54
	Sugars	61.39	42.43	9.62	28.34
	Sodium	83.41	77.73	26.18	45.83
GloVe	Fat	39.78	33.37	19.40	13.82
	Protein	57.65	49.38	27.09	15.33
	Saturated fat	77.09	75.79	19.06	14.55
	Sugars	59.53	40.03	9.64	28.35
	Sodium	83.40	79.46	27.03	45.84
Doc2Vec	Fat	39.53	31.71	19.40	13.82
	Protein	54.67	45.84	27.09	15.33
	Saturated fat	76.26	75.15	19.06	14.55
	Sugars	56.38	34.12	9.64	28.35
	Sodium	83.40	79.91	27.03	45.84
BERT	Fat	38.21	33.20	19.40	13.82
	Protein	57.97	48.26	27.09	15.33
	Saturated fat	77.16	75.08	19.06	14.55
	Sugars	53.86	38.82	9.64	28.35
	Sodium	83.40	75.98	27.03	45.84

Table A.12: Results for the **Recipe box** dataset obtained without the domain heuristic.

Embedding Algorithm	Target	Max Accuracy	Average Accuracy	Baseline Mean Average	Baseline Median Average
Word2Vec	Fat	45.40	35.11	18.61	11.08
	Protein	67.19	58.32	26.53	18.82
	Saturated fat	75.32	72.30	16.00	10.04
	Sugars	53.3	35.48	9.61	12.82
	Sodium	83.54	81.02	38.39	45.77
GloVe	Fat	44.17	34.50	18.61	11.08
	Protein	64.73	55.64	26.53	18.82
	Saturated fat	74.42	72.04	16.00	10.04
	Sugars	49.51	32.84	9.61	12.82
	Sodium	83.55	82.89	38.39	45.77
Doc2Vec	Fat	43.63	32.51	18.61	11.08
	Protein	59.46	51.94	26.53	18.82
	Saturated fat	73.15	70.75	16.00	10.04
	Sugars	40.17	27.24	9.61	12.82
	Sodium	83.40	82.22	38.39	45.77
BERT	Fat	37.51	26.29	7.84	7.53
	Protein	70.35	63.57	27.06	22.89
	Saturated fat	62.63	55.08	7.27	8.78
	Sugars	50.28	33.44	6.94	13.47
	Sodium	81.48	81.19	25.39	43.57

Appendix B

Results from Chapter 7

Table B.1: Results from training on one recipe dataset and testing on other five recipe datasets with embeddings merged with the domain heuristic as features.

Train	Test	Embedding	Target	Max	Avg	Mean	Median
recipe1M	indianrecipes	Word2Vec	Fat	45.72	25.36	6.42	12.17
recipe1M	indianrecipes	Word2Vec	Protein	56.52	53.99	35.31	20.88
recipe1M	indianrecipes	Word2Vec	Saturated fat	72.49	55.97	6.16	10.12
recipe1M	indianrecipes	Word2Vec	Sugars	37.95	21.67	6.41	8.51
recipe1M	indianrecipes	Word2Vec	Sodium	42.91	36.23	30.42	17.13
recipe1M	indianrecipes	GloVe	Fat	64.64	37.13	8.86	16.45
recipe1M	indianrecipes	GloVe	Protein	80.85	69.57	49.37	23.72
recipe1M	indianrecipes	GloVe	Saturated fat	85.77	57.44	7.08	11.39
recipe1M	indianrecipes	GloVe	Sugars	48.98	23.83	6.85	13.28
recipe1M	indianrecipes	GloVe	Sodium	99.91	69.01	54.35	31.11
recipe1M	indianrecipes	Doc2Vec	Fat	36.23	31.07	6.42	12.17
recipe1M	indianrecipes	Doc2Vec	Protein	58.63	53.43	35.31	20.88
recipe1M	indianrecipes	Doc2Vec	Saturated fat	66.39	54.41	6.16	10.12
recipe1M	indianrecipes	Doc2Vec	Sugars	26.68	12.35	6.41	8.51
recipe1M	indianrecipes	Doc2Vec	Sodium	36.47	34.29	30.42	17.13
recipe1M	indianrecipes	BERT	Fat	64.64	37.13	8.86	16.45
recipe1M	indianrecipes	BERT	Protein	80.85	69.57	49.37	23.72
recipe1M	indianrecipes	BERT	Saturated fat	85.77	57.44	7.08	11.39
recipe1M	indianrecipes	BERT	Sugars	48.98	23.83	6.85	13.28
recipe1M	indianrecipes	BERT	Sodium	99.91	69.01	54.35	31.11
recipe1M	epicurious	Word2Vec	Fat	64.44	44.44	6.24	11.51
recipe1M	epicurious	Word2Vec	Protein	81.28	77.49	33.7	20.77
recipe1M	epicurious	Word2Vec	Saturated fat	93.83	78.47	6.59	10.63
recipe1M	epicurious	Word2Vec	Sugars	60.38	42.63	7.18	7.75
recipe1M	epicurious	Word2Vec	Sodium	65.93	53.41	23.98	13.29
recipe1M	epicurious	GloVe	Fat	70.44	52.26	7.5	14.63
recipe1M	epicurious	GloVe	Protein	94.51	91.16	44.46	22.85
recipe1M	epicurious	GloVe	Saturated fat	98.92	77.44	7.35	12.0
recipe1M	epicurious	GloVe	Sugars	59.33	44.74	8.41	11.57
recipe1M	epicurious	GloVe	Sodium	96.16	78.57	41.72	23.6
recipe1M	epicurious	Doc2Vec	Fat	55.25	49.78	6.24	11.51
recipe1M	epicurious	Doc2Vec	Protein	79.17	75.56	33.7	20.77

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
recipe1M	epicurious	Doc2Vec	Saturated fat	87.33	76.45	6.59	10.63
recipe1M	epicurious	Doc2Vec	Sugars	45.65	32.43	7.18	7.75
recipe1M	epicurious	Doc2Vec	Sodium	60.94	49.09	23.98	13.29
recipe1M	epicurious	BERT	Fat	70.44	52.26	7.5	14.63
recipe1M	epicurious	BERT	Protein	94.51	91.16	44.46	22.85
recipe1M	epicurious	BERT	Saturated fat	98.92	77.44	7.35	12.0
recipe1M	epicurious	BERT	Sugars	59.33	44.74	8.41	11.57
recipe1M	epicurious	BERT	Sodium	96.16	78.57	41.72	23.6
recipe1M	saladrecipes	Word2Vec	Fat	37.32	15.79	4.6	7.41
recipe1M	saladrecipes	Word2Vec	Protein	73.83	67.41	34.21	25.37
recipe1M	saladrecipes	Word2Vec	Saturated fat	66.26	43.38	5.42	8.42
recipe1M	saladrecipes	Word2Vec	Sugars	32.44	17.9	8.24	4.89
recipe1M	saladrecipes	Word2Vec	Sodium	27.9	22.49	7.41	3.58
recipe1M	saladrecipes	GloVe	Fat	40.95	24.29	6.24	11.51
recipe1M	saladrecipes	GloVe	Protein	60.43	57.49	33.7	20.77
recipe1M	saladrecipes	GloVe	Saturated fat	71.76	45.51	6.59	10.63
recipe1M	saladrecipes	GloVe	Sugars	29.73	18.27	7.18	7.75
recipe1M	saladrecipes	GloVe	Sodium	60.53	51.0	23.98	13.29
recipe1M	saladrecipes	Doc2Vec	Fat	27.11	19.31	4.6	7.41
recipe1M	saladrecipes	Doc2Vec	Protein	68.94	60.36	34.21	25.37
recipe1M	saladrecipes	Doc2Vec	Saturated fat	63.17	39.83	5.42	8.42
recipe1M	saladrecipes	Doc2Vec	Sugars	28.14	9.91	8.24	4.89
recipe1M	saladrecipes	Doc2Vec	Sodium	23.53	21.34	7.41	3.58
recipe1M	saladrecipes	BERT	Fat	40.95	24.29	6.24	11.51
recipe1M	saladrecipes	BERT	Protein	60.43	57.49	33.7	20.77
recipe1M	saladrecipes	BERT	Saturated fat	71.76	45.51	6.59	10.63
recipe1M	saladrecipes	BERT	Sugars	29.73	18.27	7.18	7.75
recipe1M	saladrecipes	BERT	Sodium	60.53	51.0	23.98	13.29
recipe1M	yummly28K	Word2Vec	Fat	36.62	17.6	4.99	8.38
recipe1M	yummly28K	Word2Vec	Protein	47.55	42.04	22.93	18.68
recipe1M	yummly28K	Word2Vec	Saturated fat	67.51	47.2	5.83	9.26
recipe1M	yummly28K	Word2Vec	Sugars	32.06	15.33	5.95	3.94
recipe1M	yummly28K	Word2Vec	Sodium	15.61	4.78	6.23	2.97
recipe1M	yummly28K	GloVe	Fat	45.54	27.44	6.79	12.87
recipe1M	yummly28K	GloVe	Protein	59.29	55.18	33.52	19.24
recipe1M	yummly28K	GloVe	Saturated fat	74.67	50.19	6.98	11.37
recipe1M	yummly28K	GloVe	Sugars	32.41	19.7	6.82	8.71
recipe1M	yummly28K	GloVe	Sodium	53.34	40.67	29.5	16.52
recipe1M	yummly28K	Doc2Vec	Fat	27.82	22.51	4.99	8.38
recipe1M	yummly28K	Doc2Vec	Protein	51.65	41.16	22.93	18.68
recipe1M	yummly28K	Doc2Vec	Saturated fat	63.23	45.15	5.83	9.26
recipe1M	yummly28K	Doc2Vec	Sugars	21.31	7.33	5.95	3.94
recipe1M	yummly28K	Doc2Vec	Sodium	7.4	2.25	6.23	2.97
recipe1M	yummly28K	BERT	Fat	45.54	27.44	6.79	12.87
recipe1M	yummly28K	BERT	Protein	59.29	55.18	33.52	19.24
recipe1M	yummly28K	BERT	Saturated fat	74.67	50.19	6.98	11.37
recipe1M	yummly28K	BERT	Sugars	32.41	19.7	6.82	8.71
recipe1M	yummly28K	BERT	Sodium	53.34	40.67	29.5	16.52

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
recipe1M	recipebox	Word2Vec	Fat	45.44	27.69	8.26	11.83
recipe1M	recipebox	Word2Vec	Protein	65.21	60.83	34.4	21.53
recipe1M	recipebox	Word2Vec	Saturated fat	72.19	57.29	7.58	10.58
recipe1M	recipebox	Word2Vec	Sugars	36.54	19.74	6.19	7.16
recipe1M	recipebox	Word2Vec	Sodium	44.21	29.82	20.13	11.06
recipe1M	recipebox	GloVe	Fat	42.61	27.18	8.26	11.83
recipe1M	recipebox	GloVe	Protein	63.86	60.57	34.4	21.53
recipe1M	recipebox	GloVe	Saturated fat	70.08	44.35	7.58	10.58
recipe1M	recipebox	GloVe	Sugars	26.1	15.82	6.19	7.16
recipe1M	recipebox	GloVe	Sodium	37.35	27.15	20.13	11.06
recipe1M	recipebox	Doc2Vec	Fat	37.37	32.31	8.26	11.83
recipe1M	recipebox	Doc2Vec	Protein	61.66	58.33	34.4	21.53
recipe1M	recipebox	Doc2Vec	Saturated fat	65.95	55.88	7.58	10.58
recipe1M	recipebox	Doc2Vec	Sugars	22.41	10.5	6.19	7.16
recipe1M	recipebox	Doc2Vec	Sodium	38.04	24.79	20.13	11.06
recipe1M	recipebox	BERT	Fat	42.61	27.18	8.26	11.83
recipe1M	recipebox	BERT	Protein	63.86	60.57	34.4	21.53
recipe1M	recipebox	BERT	Saturated fat	70.08	44.35	7.58	10.58
recipe1M	recipebox	BERT	Sugars	26.1	15.82	6.19	7.16
recipe1M	recipebox	BERT	Sodium	37.35	27.15	20.13	11.06
indianrecipes	recipe1M	Word2Vec	Fat	32.36	26.22	10.14	8.82
indianrecipes	recipe1M	Word2Vec	Protein	25.39	19.81	8.39	6.23
indianrecipes	recipe1M	Word2Vec	Saturated fat	70.3	61.58	13.49	15.94
indianrecipes	recipe1M	Word2Vec	Sugars	28.62	18.35	7.94	8.45
indianrecipes	recipe1M	Word2Vec	Sodium	35.98	32.62	21.87	17.51
indianrecipes	recipe1M	GloVe	Fat	30.42	21.9	10.14	8.82
indianrecipes	recipe1M	GloVe	Protein	25.23	17.12	8.39	6.23
indianrecipes	recipe1M	GloVe	Saturated fat	69.31	56.75	13.49	15.94
indianrecipes	recipe1M	GloVe	Sugars	29.23	17.39	7.94	8.45
indianrecipes	recipe1M	GloVe	Sodium	36.54	33.69	21.87	17.51
indianrecipes	recipe1M	Doc2Vec	Fat	31.34	21.11	10.14	8.82
indianrecipes	recipe1M	Doc2Vec	Protein	25.2	18.22	8.39	6.23
indianrecipes	recipe1M	Doc2Vec	Saturated fat	71.19	56.85	13.49	15.94
indianrecipes	recipe1M	Doc2Vec	Sugars	28.93	18.13	7.94	8.45
indianrecipes	recipe1M	Doc2Vec	Sodium	35.26	32.98	21.87	17.51
indianrecipes	recipe1M	BERT	Fat	30.42	21.9	10.14	8.82
indianrecipes	recipe1M	BERT	Protein	25.23	17.12	8.39	6.23
indianrecipes	recipe1M	BERT	Saturated fat	69.31	56.75	13.49	15.94
indianrecipes	recipe1M	BERT	Sugars	29.23	17.39	7.94	8.45
indianrecipes	recipe1M	BERT	Sodium	36.54	33.69	21.87	17.51
indianrecipes	epicurious	Word2Vec	Fat	31.53	26.22	10.45	9.18
indianrecipes	epicurious	Word2Vec	Protein	26.06	20.76	9.5	6.83
indianrecipes	epicurious	Word2Vec	Saturated fat	72.72	65.31	14.63	16.71
indianrecipes	epicurious	Word2Vec	Sugars	31.85	19.63	7.92	8.17
indianrecipes	epicurious	Word2Vec	Sodium	37.69	30.77	17.48	13.66
indianrecipes	epicurious	GloVe	Fat	30.22	21.97	10.45	9.18
indianrecipes	epicurious	GloVe	Protein	26.5	18.38	9.5	6.83
indianrecipes	epicurious	GloVe	Saturated fat	72.16	60.64	14.63	16.71

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
indianrecipes	epicurious	GloVe	Sugars	30.76	18.86	7.92	8.17
indianrecipes	epicurious	GloVe	Sodium	36.46	31.48	17.48	13.66
indianrecipes	epicurious	Doc2Vec	Fat	31.16	21.54	10.45	9.18
indianrecipes	epicurious	Doc2Vec	Protein	26.77	20.02	9.5	6.83
indianrecipes	epicurious	Doc2Vec	Saturated fat	75.62	61.7	14.63	16.71
indianrecipes	epicurious	Doc2Vec	Sugars	29.28	19.61	7.92	8.17
indianrecipes	epicurious	Doc2Vec	Sodium	39.79	30.11	17.48	13.66
indianrecipes	epicurious	BERT	Fat	30.22	21.97	10.45	9.18
indianrecipes	epicurious	BERT	Protein	26.5	18.38	9.5	6.83
indianrecipes	epicurious	BERT	Saturated fat	72.16	60.64	14.63	16.71
indianrecipes	epicurious	BERT	Sugars	30.76	18.86	7.92	8.17
indianrecipes	epicurious	BERT	Sodium	36.46	31.48	17.48	13.66
indianrecipes	saladrecipes	Word2Vec	Fat	23.32	16.98	9.34	8.16
indianrecipes	saladrecipes	Word2Vec	Protein	42.72	33.53	15.89	10.62
indianrecipes	saladrecipes	Word2Vec	Saturated fat	65.81	49.05	12.54	13.62
indianrecipes	saladrecipes	Word2Vec	Sugars	32.54	15.11	8.51	6.14
indianrecipes	saladrecipes	Word2Vec	Sodium	24.25	21.57	7.26	3.61
indianrecipes	saladrecipes	GloVe	Fat	24.02	14.03	9.34	8.16
indianrecipes	saladrecipes	GloVe	Protein	39.41	30.75	15.89	10.62
indianrecipes	saladrecipes	GloVe	Saturated fat	62.28	45.0	12.54	13.62
indianrecipes	saladrecipes	GloVe	Sugars	26.73	14.99	8.51	6.14
indianrecipes	saladrecipes	GloVe	Sodium	23.94	21.72	7.26	3.61
indianrecipes	saladrecipes	Doc2Vec	Fat	29.95	15.23	9.34	8.16
indianrecipes	saladrecipes	Doc2Vec	Protein	42.61	34.69	15.89	10.62
indianrecipes	saladrecipes	Doc2Vec	Saturated fat	68.04	47.89	12.54	13.62
indianrecipes	saladrecipes	Doc2Vec	Sugars	34.88	16.18	8.51	6.14
indianrecipes	saladrecipes	Doc2Vec	Sodium	23.15	21.18	7.26	3.61
indianrecipes	saladrecipes	BERT	Fat	24.02	14.03	9.34	8.16
indianrecipes	saladrecipes	BERT	Protein	39.41	30.75	15.89	10.62
indianrecipes	saladrecipes	BERT	Saturated fat	62.28	45.0	12.54	13.62
indianrecipes	saladrecipes	BERT	Sugars	26.73	14.99	8.51	6.14
indianrecipes	saladrecipes	BERT	Sodium	23.94	21.72	7.26	3.61
indianrecipes	yummly28K	Word2Vec	Fat	47.01	41.16	10.3	9.15
indianrecipes	yummly28K	Word2Vec	Protein	43.76	38.43	8.14	5.88
indianrecipes	yummly28K	Word2Vec	Saturated fat	89.95	79.01	13.47	15.0
indianrecipes	yummly28K	Word2Vec	Sugars	45.52	33.64	7.03	4.74
indianrecipes	yummly28K	Word2Vec	Sodium	26.28	22.27	6.11	3.05
indianrecipes	yummly28K	GloVe	Fat	46.01	37.07	10.3	9.15
indianrecipes	yummly28K	GloVe	Protein	41.96	36.86	8.14	5.88
indianrecipes	yummly28K	GloVe	Saturated fat	89.38	73.32	13.47	15.0
indianrecipes	yummly28K	GloVe	Sugars	41.38	32.88	7.03	4.74
indianrecipes	yummly28K	GloVe	Sodium	26.71	22.55	6.11	3.05
indianrecipes	yummly28K	Doc2Vec	Fat	49.52	38.65	10.3	9.15
indianrecipes	yummly28K	Doc2Vec	Protein	43.69	39.01	8.14	5.88
indianrecipes	yummly28K	Doc2Vec	Saturated fat	91.45	77.56	13.47	15.0
indianrecipes	yummly28K	Doc2Vec	Sugars	46.15	34.26	7.03	4.74
indianrecipes	yummly28K	Doc2Vec	Sodium	24.85	21.82	6.11	3.05
indianrecipes	yummly28K	BERT	Fat	46.01	37.07	10.3	9.15

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
indianrecipes	yummly28K	BERT	Protein	41.96	36.86	8.14	5.88
indianrecipes	yummly28K	BERT	Saturated fat	89.38	73.32	13.47	15.0
indianrecipes	yummly28K	BERT	Sugars	41.38	32.88	7.03	4.74
indianrecipes	yummly28K	BERT	Sodium	26.71	22.55	6.11	3.05
indianrecipes	recipebox	Word2Vec	Fat	36.34	31.06	12.77	10.02
indianrecipes	recipebox	Word2Vec	Protein	28.93	23.75	10.61	7.61
indianrecipes	recipebox	Word2Vec	Saturated fat	72.64	65.76	15.09	15.93
indianrecipes	recipebox	Word2Vec	Sugars	27.51	16.96	7.26	7.44
indianrecipes	recipebox	Word2Vec	Sodium	34.06	26.64	14.9	11.37
indianrecipes	recipebox	GloVe	Fat	34.48	27.22	12.77	10.02
indianrecipes	recipebox	GloVe	Protein	29.29	21.47	10.61	7.61
indianrecipes	recipebox	GloVe	Saturated fat	72.66	60.96	15.09	15.93
indianrecipes	recipebox	GloVe	Sugars	27.02	16.35	7.26	7.44
indianrecipes	recipebox	GloVe	Sodium	32.96	27.04	14.9	11.37
indianrecipes	recipebox	Doc2Vec	Fat	34.11	26.56	12.77	10.02
indianrecipes	recipebox	Doc2Vec	Protein	30.58	23.18	10.61	7.61
indianrecipes	recipebox	Doc2Vec	Saturated fat	76.19	62.77	15.09	15.93
indianrecipes	recipebox	Doc2Vec	Sugars	25.52	17.1	7.26	7.44
indianrecipes	recipebox	Doc2Vec	Sodium	32.3	25.32	14.9	11.37
indianrecipes	recipebox	BERT	Fat	34.48	27.22	12.77	10.02
indianrecipes	recipebox	BERT	Protein	29.29	21.47	10.61	7.61
indianrecipes	recipebox	BERT	Saturated fat	72.66	60.96	15.09	15.93
indianrecipes	recipebox	BERT	Sugars	27.02	16.35	7.26	7.44
indianrecipes	recipebox	BERT	Sodium	32.96	27.04	14.9	11.37
epicurious	recipe1M	Word2Vec	Fat	78.29	74.53	22.27	14.78
epicurious	recipe1M	Word2Vec	Protein	82.67	80.48	30.15	18.05
epicurious	recipe1M	Word2Vec	Saturated fat	83.91	82.63	14.68	12.83
epicurious	recipe1M	Word2Vec	Sugars	78.54	70.68	13.54	18.83
epicurious	recipe1M	Word2Vec	Sodium	95.76	86.87	44.73	27.19
epicurious	recipe1M	GloVe	Fat	81.35	76.75	24.68	16.12
epicurious	recipe1M	GloVe	Protein	77.18	74.05	30.3	17.64
epicurious	recipe1M	GloVe	Saturated fat	86.84	83.96	15.21	13.48
epicurious	recipe1M	GloVe	Sugars	76.32	68.85	15.37	20.29
epicurious	recipe1M	GloVe	Sodium	98.02	96.02	57.43	35.36
epicurious	recipe1M	Doc2Vec	Fat	71.31	67.41	22.27	14.78
epicurious	recipe1M	Doc2Vec	Protein	76.98	73.72	30.15	18.05
epicurious	recipe1M	Doc2Vec	Saturated fat	86.6	80.25	14.68	12.83
epicurious	recipe1M	Doc2Vec	Sugars	70.33	62.73	13.54	18.83
epicurious	recipe1M	Doc2Vec	Sodium	95.76	89.72	44.73	27.19
epicurious	recipe1M	BERT	Fat	81.35	76.75	24.68	16.12
epicurious	recipe1M	BERT	Protein	77.18	74.05	30.3	17.64
epicurious	recipe1M	BERT	Saturated fat	86.84	83.96	15.21	13.48
epicurious	recipe1M	BERT	Sugars	76.32	68.85	15.37	20.29
epicurious	recipe1M	BERT	Sodium	98.02	96.02	57.43	35.36
epicurious	indianrecipes	Word2Vec	Fat	59.83	56.07	24.62	16.3
epicurious	indianrecipes	Word2Vec	Protein	57.81	55.29	33.09	18.45
epicurious	indianrecipes	Word2Vec	Saturated fat	86.52	84.92	14.92	13.41
epicurious	indianrecipes	Word2Vec	Sugars	61.38	51.69	11.54	16.64

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
epicurious	indianrecipes	Word2Vec	Sodium	95.83	83.99	32.58	23.12
epicurious	indianrecipes	GloVe	Fat	74.4	63.95	32.93	20.29
epicurious	indianrecipes	GloVe	Protein	78.01	71.15	48.57	24.15
epicurious	indianrecipes	GloVe	Saturated fat	90.54	86.43	16.11	14.8
epicurious	indianrecipes	GloVe	Sugars	67.89	53.52	17.5	23.11
epicurious	indianrecipes	GloVe	Sodium	99.87	95.8	67.46	37.97
epicurious	indianrecipes	Doc2Vec	Fat	55.62	51.6	24.62	16.3
epicurious	indianrecipes	Doc2Vec	Protein	55.99	52.46	33.09	18.45
epicurious	indianrecipes	Doc2Vec	Saturated fat	90.37	83.94	14.92	13.41
epicurious	indianrecipes	Doc2Vec	Sugars	52.23	43.14	11.54	16.64
epicurious	indianrecipes	Doc2Vec	Sodium	95.83	87.77	32.58	23.12
epicurious	indianrecipes	BERT	Fat	74.4	63.95	32.93	20.29
epicurious	indianrecipes	BERT	Protein	78.01	71.15	48.57	24.15
epicurious	indianrecipes	BERT	Saturated fat	90.54	86.43	16.11	14.8
epicurious	indianrecipes	BERT	Sugars	67.89	53.52	17.5	23.11
epicurious	indianrecipes	BERT	Sodium	99.87	95.8	67.46	37.97
epicurious	saladrecipes	Word2Vec	Fat	49.24	40.47	15.02	10.76
epicurious	saladrecipes	Word2Vec	Protein	78.83	72.69	29.68	19.29
epicurious	saladrecipes	Word2Vec	Saturated fat	81.54	77.23	13.1	10.86
epicurious	saladrecipes	Word2Vec	Sugars	61.68	47.79	8.06	14.46
epicurious	saladrecipes	Word2Vec	Sodium	88.98	79.53	6.65	2.66
epicurious	saladrecipes	GloVe	Fat	55.7	52.38	22.27	14.78
epicurious	saladrecipes	GloVe	Protein	61.58	58.12	30.15	18.05
epicurious	saladrecipes	GloVe	Saturated fat	84.47	82.75	14.68	12.83
epicurious	saladrecipes	GloVe	Sugars	55.87	47.31	13.54	18.83
epicurious	saladrecipes	GloVe	Sodium	95.76	87.94	44.73	27.19
epicurious	saladrecipes	Doc2Vec	Fat	44.77	34.67	15.02	10.76
epicurious	saladrecipes	Doc2Vec	Protein	70.71	65.21	29.68	19.29
epicurious	saladrecipes	Doc2Vec	Saturated fat	89.1	76.32	13.1	10.86
epicurious	saladrecipes	Doc2Vec	Sugars	49.43	35.23	8.07	14.46
epicurious	saladrecipes	Doc2Vec	Sodium	88.98	89.23	6.65	2.66
epicurious	saladrecipes	BERT	Fat	55.7	52.38	22.27	14.78
epicurious	saladrecipes	BERT	Protein	61.58	58.12	30.15	18.05
epicurious	saladrecipes	BERT	Saturated fat	84.47	82.75	14.68	12.83
epicurious	saladrecipes	BERT	Sugars	55.87	47.31	13.54	18.83
epicurious	saladrecipes	BERT	Sodium	95.76	87.94	44.73	27.19
epicurious	yummly28K	Word2Vec	Fat	54.67	48.08	16.95	12.25
epicurious	yummly28K	Word2Vec	Protein	48.36	44.49	18.9	13.66
epicurious	yummly28K	Word2Vec	Saturated fat	85.16	82.67	14.01	11.92
epicurious	yummly28K	Word2Vec	Sugars	57.59	45.0	6.9	10.95
epicurious	yummly28K	Word2Vec	Sodium	93.81	76.05	5.22	13.25
epicurious	yummly28K	GloVe	Fat	67.78	62.5	29.41	19.06
epicurious	yummly28K	GloVe	Protein	50.83	46.49	34.79	18.04
epicurious	yummly28K	GloVe	Saturated fat	91.3	88.43	15.83	14.69
epicurious	yummly28K	GloVe	Sugars	59.21	48.67	13.28	17.73
epicurious	yummly28K	GloVe	Sodium	99.26	96.24	45.54	33.35
epicurious	yummly28K	Doc2Vec	Fat	49.63	41.64	16.94	12.24
epicurious	yummly28K	Doc2Vec	Protein	44.63	39.44	18.9	13.66

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
epicurious	yummly28K	Doc2Vec	Saturated fat	90.59	81.52	14.01	11.92
epicurious	yummly28K	Doc2Vec	Sugars	46.94	32.84	6.9	10.95
epicurious	yummly28K	Doc2Vec	Sodium	93.81	81.72	5.22	13.25
epicurious	yummly28K	BERT	Fat	67.78	62.5	29.41	19.06
epicurious	yummly28K	BERT	Protein	50.83	46.49	34.79	18.04
epicurious	yummly28K	BERT	Saturated fat	91.3	88.43	15.83	14.69
epicurious	yummly28K	BERT	Sugars	59.21	48.67	13.28	17.73
epicurious	yummly28K	BERT	Sodium	99.26	96.24	45.54	33.35
epicurious	recipebox	Word2Vec	Fat	57.56	54.41	23.08	14.54
epicurious	recipebox	Word2Vec	Protein	67.39	65.28	30.53	18.69
epicurious	recipebox	Word2Vec	Saturated fat	83.86	82.47	15.23	12.56
epicurious	recipebox	Word2Vec	Sugars	52.55	44.94	12.86	16.24
epicurious	recipebox	Word2Vec	Sodium	96.28	89.06	39.11	30.97
epicurious	recipebox	GloVe	Fat	55.63	52.66	23.08	14.54
epicurious	recipebox	GloVe	Protein	66.08	63.01	30.53	18.69
epicurious	recipebox	GloVe	Saturated fat	84.97	82.55	15.23	12.56
epicurious	recipebox	GloVe	Sugars	51.14	41.78	12.86	16.24
epicurious	recipebox	GloVe	Sodium	96.28	89.83	39.11	30.97
epicurious	recipebox	Doc2Vec	Fat	51.25	48.26	23.08	14.54
epicurious	recipebox	Doc2Vec	Protein	60.7	58.01	30.53	18.69
epicurious	recipebox	Doc2Vec	Saturated fat	86.31	80.69	15.23	12.56
epicurious	recipebox	Doc2Vec	Sugars	46.15	37.83	12.86	16.24
epicurious	recipebox	Doc2Vec	Sodium	96.28	91.26	39.11	30.97
epicurious	recipebox	BERT	Fat	55.63	52.66	23.08	14.54
epicurious	recipebox	BERT	Protein	66.08	63.01	30.53	18.69
epicurious	recipebox	BERT	Saturated fat	84.97	82.55	15.23	12.56
epicurious	recipebox	BERT	Sugars	51.14	41.78	12.86	16.24
epicurious	recipebox	BERT	Sodium	96.28	89.83	39.11	30.97
saladrecipes	recipe1M	Word2Vec	Fat	64.0	61.59	30.69	19.32
saladrecipes	recipe1M	Word2Vec	Protein	76.14	65.84	37.77	21.61
saladrecipes	recipe1M	Word2Vec	Saturated fat	81.98	80.96	20.21	18.3
saladrecipes	recipe1M	Word2Vec	Sugars	63.51	60.13	26.04	25.83
saladrecipes	recipe1M	Word2Vec	Sodium	97.8	97.68	81.58	40.44
saladrecipes	recipe1M	GloVe	Fat	62.55	59.84	27.73	18.04
saladrecipes	recipe1M	GloVe	Protein	50.19	45.93	26.05	16.04
saladrecipes	recipe1M	GloVe	Saturated fat	85.64	84.36	20.12	17.5
saladrecipes	recipe1M	GloVe	Sugars	57.98	51.81	19.23	20.87
saladrecipes	recipe1M	GloVe	Sodium	97.91	97.83	84.36	40.18
saladrecipes	recipe1M	Doc2Vec	Fat	64.0	61.59	30.69	19.32
saladrecipes	recipe1M	Doc2Vec	Protein	76.14	66.06	37.77	21.61
saladrecipes	recipe1M	Doc2Vec	Saturated fat	81.98	80.96	20.21	18.3
saladrecipes	recipe1M	Doc2Vec	Sugars	63.51	60.1	26.04	25.83
saladrecipes	recipe1M	Doc2Vec	Sodium	97.8	97.68	81.58	40.44
saladrecipes	recipe1M	BERT	Fat	62.55	59.84	27.73	18.04
saladrecipes	recipe1M	BERT	Protein	50.19	45.93	26.05	16.04
saladrecipes	recipe1M	BERT	Saturated fat	85.64	84.36	20.12	17.5
saladrecipes	recipe1M	BERT	Sugars	57.98	51.81	19.23	20.87
saladrecipes	recipe1M	BERT	Sodium	97.91	97.83	84.36	40.18

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
saladrecipes	indianrecipes	Word2Vec	Fat	94.82	92.87	44.17	26.22
saladrecipes	indianrecipes	Word2Vec	Protein	83.51	81.42	61.06	31.04
saladrecipes	indianrecipes	Word2Vec	Saturated fat	90.11	88.37	23.04	21.51
saladrecipes	indianrecipes	Word2Vec	Sugars	89.83	87.85	28.56	27.07
saladrecipes	indianrecipes	Word2Vec	Sodium	99.87	99.87	85.14	42.28
saladrecipes	indianrecipes	GloVe	Fat	94.49	86.94	38.22	22.93
saladrecipes	indianrecipes	GloVe	Protein	81.59	90.77	45.48	24.65
saladrecipes	indianrecipes	GloVe	Saturated fat	88.51	86.89	22.17	19.53
saladrecipes	indianrecipes	GloVe	Sugars	87.94	77.31	23.08	23.32
saladrecipes	indianrecipes	GloVe	Sodium	99.87	99.09	85.17	40.99
saladrecipes	indianrecipes	Doc2Vec	Fat	94.82	92.92	44.17	26.22
saladrecipes	indianrecipes	Doc2Vec	Protein	83.51	81.44	61.06	31.04
saladrecipes	indianrecipes	Doc2Vec	Saturated fat	90.11	88.37	23.04	21.51
saladrecipes	indianrecipes	Doc2Vec	Sugars	89.83	87.79	28.56	27.07
saladrecipes	indianrecipes	Doc2Vec	Sodium	99.87	99.87	85.14	42.28
saladrecipes	indianrecipes	BERT	Fat	94.49	86.94	38.22	22.93
saladrecipes	indianrecipes	BERT	Protein	81.59	90.77	45.48	24.65
saladrecipes	indianrecipes	BERT	Saturated fat	88.51	86.89	22.17	19.53
saladrecipes	indianrecipes	BERT	Sugars	87.94	77.31	23.08	23.32
saladrecipes	indianrecipes	BERT	Sodium	99.87	99.09	85.17	40.99
saladrecipes	epicurious	Word2Vec	Fat	62.05	58.3	27.62	17.61
saladrecipes	epicurious	Word2Vec	Protein	71.99	63.02	31.12	18.81
saladrecipes	epicurious	Word2Vec	Saturated fat	84.56	83.69	21.42	18.4
saladrecipes	epicurious	Word2Vec	Sugars	64.42	59.65	20.44	24.16
saladrecipes	epicurious	Word2Vec	Sodium	97.93	97.85	82.76	39.72
saladrecipes	epicurious	GloVe	Fat	63.84	59.27	26.16	17.08
saladrecipes	epicurious	GloVe	Protein	55.43	51.31	24.0	15.34
saladrecipes	epicurious	GloVe	Saturated fat	87.27	85.76	21.05	17.78
saladrecipes	epicurious	GloVe	Sugars	60.29	52.61	16.73	20.86
saladrecipes	epicurious	GloVe	Sodium	98.03	97.97	84.55	39.71
saladrecipes	epicurious	Doc2Vec	Fat	62.0	58.23	27.62	17.61
saladrecipes	epicurious	Doc2Vec	Protein	71.99	63.1	31.12	18.81
saladrecipes	epicurious	Doc2Vec	Saturated fat	84.33	83.68	21.42	18.4
saladrecipes	epicurious	Doc2Vec	Sugars	64.42	59.69	20.44	24.16
saladrecipes	epicurious	Doc2Vec	Sodium	97.93	97.86	82.76	39.72
saladrecipes	epicurious	BERT	Fat	63.84	59.27	26.16	17.08
saladrecipes	epicurious	BERT	Protein	55.43	51.31	24.0	15.34
saladrecipes	epicurious	BERT	Saturated fat	87.27	85.76	21.05	17.78
saladrecipes	epicurious	BERT	Sugars	60.29	52.61	16.73	20.86
saladrecipes	epicurious	BERT	Sodium	98.03	97.97	84.55	39.71
saladrecipes	yummly28K	Word2Vec	Fat	62.02	56.79	26.16	17.08
saladrecipes	yummly28K	Word2Vec	Protein	55.81	48.74	24.0	15.34
saladrecipes	yummly28K	Word2Vec	Saturated fat	86.23	85.32	21.05	17.78
saladrecipes	yummly28K	Word2Vec	Sugars	63.13	54.78	16.73	20.86
saladrecipes	yummly28K	Word2Vec	Sodium	98.03	97.29	84.55	39.71
saladrecipes	yummly28K	GloVe	Fat	69.05	64.82	32.99	20.85
saladrecipes	yummly28K	GloVe	Protein	47.42	43.44	31.84	17.98
saladrecipes	yummly28K	GloVe	Saturated fat	91.17	89.72	21.49	18.71

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
saladrecipes	yummly28K	GloVe	Sugars	60.36	51.13	17.09	19.01
saladrecipes	yummly28K	GloVe	Sodium	99.11	99.01	87.53	40.97
saladrecipes	yummly28K	Doc2Vec	Fat	62.02	56.74	26.16	17.08
saladrecipes	yummly28K	Doc2Vec	Protein	55.81	48.8	24.0	15.34
saladrecipes	yummly28K	Doc2Vec	Saturated fat	85.95	85.31	21.05	17.78
saladrecipes	yummly28K	Doc2Vec	Sugars	62.89	54.76	16.73	20.86
saladrecipes	yummly28K	Doc2Vec	Sodium	98.03	97.24	84.55	39.71
saladrecipes	yummly28K	BERT	Fat	69.05	64.82	32.99	20.85
saladrecipes	yummly28K	BERT	Protein	47.42	43.44	31.84	17.98
saladrecipes	yummly28K	BERT	Saturated fat	91.17	89.72	21.49	18.71
saladrecipes	yummly28K	BERT	Sugars	60.36	51.13	17.09	19.01
saladrecipes	yummly28K	BERT	Sodium	99.11	99.01	87.53	40.97
saladrecipes	recipebox	Word2Vec	Fat	60.14	55.88	26.32	16.34
saladrecipes	recipebox	Word2Vec	Protein	60.14	52.58	23.66	15.71
saladrecipes	recipebox	Word2Vec	Saturated fat	85.72	84.88	20.94	16.75
saladrecipes	recipebox	Word2Vec	Sugars	58.39	48.47	15.41	18.07
saladrecipes	recipebox	Word2Vec	Sodium	98.07	97.19	85.2	39.18
saladrecipes	recipebox	GloVe	Fat	61.36	57.61	26.32	16.34
saladrecipes	recipebox	GloVe	Protein	60.96	57.01	23.66	15.71
saladrecipes	recipebox	GloVe	Saturated fat	87.08	85.36	20.94	16.75
saladrecipes	recipebox	GloVe	Sugars	54.52	46.82	15.41	18.07
saladrecipes	recipebox	GloVe	Sodium	98.06	98.01	85.2	39.18
saladrecipes	recipebox	Doc2Vec	Fat	60.14	55.83	26.32	16.34
saladrecipes	recipebox	Doc2Vec	Protein	60.14	52.81	23.66	15.71
saladrecipes	recipebox	Doc2Vec	Saturated fat	85.72	84.88	20.94	16.75
saladrecipes	recipebox	Doc2Vec	Sugars	55.95	48.39	15.41	18.07
saladrecipes	recipebox	Doc2Vec	Sodium	98.07	97.17	85.2	39.18
saladrecipes	recipebox	BERT	Fat	61.36	57.61	26.32	16.34
saladrecipes	recipebox	BERT	Protein	60.96	57.01	23.66	15.71
saladrecipes	recipebox	BERT	Saturated fat	87.08	85.36	20.94	16.75
saladrecipes	recipebox	BERT	Sugars	54.52	46.82	15.41	18.07
saladrecipes	recipebox	BERT	Sodium	98.06	98.01	85.2	39.18
yummly28K	recipe1M	Word2Vec	Fat	58.25	54.11	24.11	15.23
yummly28K	recipe1M	Word2Vec	Protein	68.25	67.26	50.2	25.57
yummly28K	recipe1M	Word2Vec	Saturated fat	84.34	82.77	18.2	14.69
yummly28K	recipe1M	Word2Vec	Sugars	64.67	62.56	37.79	24.17
yummly28K	recipe1M	Word2Vec	Sodium	97.91	93.42	56.88	33.63
yummly28K	recipe1M	GloVe	Fat	63.07	58.69	25.46	15.93
yummly28K	recipe1M	GloVe	Protein	70.32	69.44	48.46	26.19
yummly28K	recipe1M	GloVe	Saturated fat	86.28	84.89	20.57	16.42
yummly28K	recipe1M	GloVe	Sugars	64.41	62.28	39.65	23.83
yummly28K	recipe1M	GloVe	Sodium	97.99	97.9	85.84	40.9
yummly28K	recipe1M	Doc2Vec	Fat	51.6	48.72	24.11	15.23
yummly28K	recipe1M	Doc2Vec	Protein	69.97	68.57	50.2	25.57
yummly28K	recipe1M	Doc2Vec	Saturated fat	84.47	81.02	18.2	14.69
yummly28K	recipe1M	Doc2Vec	Sugars	63.23	59.88	37.79	24.17
yummly28K	recipe1M	Doc2Vec	Sodium	97.91	91.59	56.88	33.63
yummly28K	recipe1M	BERT	Fat	63.07	58.69	25.46	15.93

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
yummly28K	recipe1M	BERT	Protein	70.32	69.44	48.46	26.19
yummly28K	recipe1M	BERT	Saturated fat	86.28	84.89	20.57	16.42
yummly28K	recipe1M	BERT	Sugars	64.41	62.28	39.65	23.83
yummly28K	recipe1M	BERT	Sodium	97.99	97.9	85.84	40.9
yummly28K	indianrecipes	Word2Vec	Fat	60.86	56.62	28.35	17.69
yummly28K	indianrecipes	Word2Vec	Protein	65.7	64.74	48.56	22.88
yummly28K	indianrecipes	Word2Vec	Saturated fat	88.59	86.48	18.88	15.69
yummly28K	indianrecipes	Word2Vec	Sugars	68.33	65.37	36.52	25.24
yummly28K	indianrecipes	Word2Vec	Sodium	99.12	92.37	45.75	31.02
yummly28K	indianrecipes	GloVe	Fat	74.39	65.28	34.63	20.79
yummly28K	indianrecipes	GloVe	Protein	70.64	68.33	48.75	23.77
yummly28K	indianrecipes	GloVe	Saturated fat	89.62	87.28	20.92	17.3
yummly28K	indianrecipes	GloVe	Sugars	66.99	62.45	43.0	23.4
yummly28K	indianrecipes	GloVe	Sodium	99.87	98.04	78.93	40.15
yummly28K	indianrecipes	Doc2Vec	Fat	57.73	54.08	28.35	17.69
yummly28K	indianrecipes	Doc2Vec	Protein	66.38	65.76	48.56	22.88
yummly28K	indianrecipes	Doc2Vec	Saturated fat	89.49	85.1	18.88	15.69
yummly28K	indianrecipes	Doc2Vec	Sugars	63.44	60.44	36.52	25.24
yummly28K	indianrecipes	Doc2Vec	Sodium	99.12	89.69	45.75	31.02
yummly28K	indianrecipes	BERT	Fat	74.39	65.28	34.63	20.79
yummly28K	indianrecipes	BERT	Protein	70.64	68.33	48.75	23.77
yummly28K	indianrecipes	BERT	Saturated fat	89.62	87.28	20.92	17.3
yummly28K	indianrecipes	BERT	Sugars	66.99	62.45	43.0	23.4
yummly28K	indianrecipes	BERT	Sodium	99.87	98.04	78.93	40.15
yummly28K	epicurious	Word2Vec	Fat	59.56	55.23	23.0	14.66
yummly28K	epicurious	Word2Vec	Protein	67.5	66.66	48.46	25.36
yummly28K	epicurious	Word2Vec	Saturated fat	86.02	84.9	19.39	15.37
yummly28K	epicurious	Word2Vec	Sugars	66.36	63.93	34.89	24.58
yummly28K	epicurious	Word2Vec	Sodium	98.05	94.62	65.75	35.56
yummly28K	epicurious	GloVe	Fat	70.1	64.2	30.38	18.74
yummly28K	epicurious	GloVe	Protein	67.98	67.17	45.95	23.81
yummly28K	epicurious	GloVe	Saturated fat	91.75	89.96	22.44	18.29
yummly28K	epicurious	GloVe	Sugars	67.41	64.62	39.3	24.74
yummly28K	epicurious	GloVe	Sodium	99.2	99.08	89.18	41.92
yummly28K	epicurious	Doc2Vec	Fat	52.97	49.39	23.0	14.66
yummly28K	epicurious	Doc2Vec	Protein	68.74	67.69	48.46	25.36
yummly28K	epicurious	Doc2Vec	Saturated fat	85.75	83.26	19.39	15.37
yummly28K	epicurious	Doc2Vec	Sugars	62.99	60.04	34.89	24.58
yummly28K	epicurious	Doc2Vec	Sodium	98.04	93.22	65.75	35.56
yummly28K	epicurious	BERT	Fat	70.1	64.2	30.38	18.74
yummly28K	epicurious	BERT	Protein	67.98	67.17	45.95	23.81
yummly28K	epicurious	BERT	Saturated fat	91.75	89.96	22.44	18.29
yummly28K	epicurious	BERT	Sugars	67.41	64.62	39.3	24.74
yummly28K	epicurious	BERT	Sodium	99.2	99.08	89.18	41.92
yummly28K	saladrecipes	Word2Vec	Fat	50.42	41.3	15.63	10.85
yummly28K	saladrecipes	Word2Vec	Protein	62.36	60.21	48.46	22.87
yummly28K	saladrecipes	Word2Vec	Saturated fat	88.4	84.1	15.84	12.23
yummly28K	saladrecipes	Word2Vec	Sugars	70.35	63.95	20.64	26.82

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
yummly28K	saladrecipes	Word2Vec	Sodium	98.38	84.88	5.49	19.55
yummly28K	saladrecipes	GloVe	Fat	57.13	54.1	23.0	14.66
yummly28K	saladrecipes	GloVe	Protein	67.98	67.17	48.46	25.36
yummly28K	saladrecipes	GloVe	Saturated fat	86.38	85.15	19.39	15.37
yummly28K	saladrecipes	GloVe	Sugars	64.62	61.47	34.89	24.58
yummly28K	saladrecipes	GloVe	Sodium	98.05	94.85	65.75	35.56
yummly28K	saladrecipes	Doc2Vec	Fat	45.18	35.73	15.63	10.85
yummly28K	saladrecipes	Doc2Vec	Protein	62.05	60.86	48.46	22.87
yummly28K	saladrecipes	Doc2Vec	Saturated fat	90.21	81.41	15.84	12.23
yummly28K	saladrecipes	Doc2Vec	Sugars	61.16	54.94	20.64	26.82
yummly28K	saladrecipes	Doc2Vec	Sodium	98.37	99.52	5.49	19.55
yummly28K	saladrecipes	BERT	Fat	57.13	54.1	23.0	14.66
yummly28K	saladrecipes	BERT	Protein	67.98	67.17	48.46	25.36
yummly28K	saladrecipes	BERT	Saturated fat	86.38	85.15	19.39	15.37
yummly28K	saladrecipes	BERT	Sugars	64.62	61.47	34.89	24.58
yummly28K	saladrecipes	BERT	Sodium	98.05	94.85	65.75	35.56
yummly28K	recipebox	Word2Vec	Fat	78.67	74.99	23.7	14.45
yummly28K	recipebox	Word2Vec	Protein	86.97	86.19	48.86	25.0
yummly28K	recipebox	Word2Vec	Saturated fat	86.6	85.24	19.53	14.72
yummly28K	recipebox	Word2Vec	Sugars	83.4	81.24	29.83	23.03
yummly28K	recipebox	Word2Vec	Sodium	98.08	95.26	69.04	36.88
yummly28K	recipebox	GloVe	Fat	76.71	74.16	23.7	14.45
yummly28K	recipebox	GloVe	Protein	87.3	86.62	48.86	25.0
yummly28K	recipebox	GloVe	Saturated fat	86.93	85.39	19.53	14.72
yummly28K	recipebox	GloVe	Sugars	82.76	78.31	29.83	23.03
yummly28K	recipebox	GloVe	Sodium	98.08	95.47	69.04	36.88
yummly28K	recipebox	Doc2Vec	Fat	73.04	69.91	23.7	14.45
yummly28K	recipebox	Doc2Vec	Protein	87.84	87.06	48.86	25.0
yummly28K	recipebox	Doc2Vec	Saturated fat	85.28	83.46	19.53	14.72
yummly28K	recipebox	Doc2Vec	Sugars	78.27	75.43	29.83	23.03
yummly28K	recipebox	Doc2Vec	Sodium	98.08	94.15	69.04	36.88
yummly28K	recipebox	BERT	Fat	76.71	74.16	23.7	14.45
yummly28K	recipebox	BERT	Protein	87.3	86.62	48.86	25.0
yummly28K	recipebox	BERT	Saturated fat	86.93	85.39	19.53	14.72
yummly28K	recipebox	BERT	Sugars	82.76	78.31	29.83	23.03
yummly28K	recipebox	BERT	Sodium	98.08	95.47	69.04	36.88
recipebox	recipe1M	Word2Vec	Fat	63.39	56.87	8.7	6.75
recipebox	recipe1M	Word2Vec	Protein	83.81	80.53	27.16	16.72
recipebox	recipe1M	Word2Vec	Saturated fat	86.33	83.86	12.94	9.11
recipebox	recipe1M	Word2Vec	Sugars	86.62	84.38	30.93	24.9
recipebox	recipe1M	Word2Vec	Sodium	98.12	91.19	44.73	30.56
recipebox	recipe1M	GloVe	Fat	58.13	53.41	8.47	6.57
recipebox	recipe1M	GloVe	Protein	82.32	79.3	25.92	16.29
recipebox	recipe1M	GloVe	Saturated fat	87.16	84.67	13.65	9.56
recipebox	recipe1M	GloVe	Sugars	85.76	81.49	30.03	25.29
recipebox	recipe1M	GloVe	Sodium	98.2	93.72	49.55	33.25
recipebox	recipe1M	Doc2Vec	Fat	57.18	51.14	8.7	6.75
recipebox	recipe1M	Doc2Vec	Protein	73.69	69.99	27.16	16.72

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
recipebox	recipe1M	Doc2Vec	Saturated fat	83.88	99.53	12.94	9.11
recipebox	recipe1M	Doc2Vec	Sugars	81.62	79.45	30.93	24.9
recipebox	recipe1M	Doc2Vec	Sodium	98.12	91.74	44.73	30.56
recipebox	recipe1M	BERT	Fat	58.13	53.41	8.47	6.57
recipebox	recipe1M	BERT	Protein	82.32	79.3	25.92	16.29
recipebox	recipe1M	BERT	Saturated fat	87.16	84.67	13.65	9.56
recipebox	recipe1M	BERT	Sugars	85.76	81.49	30.03	25.29
recipebox	recipe1M	BERT	Sodium	98.2	93.72	49.55	33.25
recipebox	indianrecipes	Word2Vec	Fat	40.28	35.87	8.7	7.34
recipebox	indianrecipes	Word2Vec	Protein	58.55	54.78	30.45	17.7
recipebox	indianrecipes	Word2Vec	Saturated fat	89.2	86.25	13.07	9.35
recipebox	indianrecipes	Word2Vec	Sugars	68.99	66.42	27.7	25.62
recipebox	indianrecipes	Word2Vec	Sodium	98.99	89.71	32.86	27.68
recipebox	indianrecipes	GloVe	Fat	71.31	53.44	11.21	9.76
recipebox	indianrecipes	GloVe	Protein	80.55	70.97	46.82	24.22
recipebox	indianrecipes	GloVe	Saturated fat	90.65	87.58	14.2	10.26
recipebox	indianrecipes	GloVe	Sugars	65.73	63.17	43.13	24.92
recipebox	indianrecipes	GloVe	Sodium	99.87	97.31	71.2	38.79
recipebox	indianrecipes	Doc2Vec	Fat	37.45	32.22	8.7	7.34
recipebox	indianrecipes	Doc2Vec	Protein	54.37	50.69	30.45	17.7
recipebox	indianrecipes	Doc2Vec	Saturated fat	86.82	82.54	13.07	9.35
recipebox	indianrecipes	Doc2Vec	Sugars	62.26	59.61	27.7	25.62
recipebox	indianrecipes	Doc2Vec	Sodium	98.99	90.46	32.86	27.68
recipebox	indianrecipes	BERT	Fat	71.31	53.44	11.21	9.76
recipebox	indianrecipes	BERT	Protein	80.55	70.97	46.82	24.22
recipebox	indianrecipes	BERT	Saturated fat	90.65	87.58	14.2	10.26
recipebox	indianrecipes	BERT	Sugars	65.73	63.17	43.13	24.92
recipebox	indianrecipes	BERT	Sodium	99.87	97.31	71.2	38.79
recipebox	epicurious	Word2Vec	Fat	41.45	35.2	8.47	6.57
recipebox	epicurious	Word2Vec	Protein	67.13	64.25	25.92	16.29
recipebox	epicurious	Word2Vec	Saturated fat	87.69	85.15	13.65	9.56
recipebox	epicurious	Word2Vec	Sugars	66.34	64.41	30.03	25.29
recipebox	epicurious	Word2Vec	Sodium	98.2	92.59	49.55	33.25
recipebox	epicurious	GloVe	Fat	49.91	46.04	10.14	8.68
recipebox	epicurious	GloVe	Protein	80.94	76.55	41.37	22.24
recipebox	epicurious	GloVe	Saturated fat	91.92	89.71	15.54	11.08
recipebox	epicurious	GloVe	Sugars	64.58	62.99	39.47	25.76
recipebox	epicurious	GloVe	Sodium	99.26	99.05	77.75	43.37
recipebox	epicurious	Doc2Vec	Fat	35.4	29.68	8.47	6.57
recipebox	epicurious	Doc2Vec	Protein	57.11	52.55	25.92	16.29
recipebox	epicurious	Doc2Vec	Saturated fat	85.32	81.63	13.65	9.56
recipebox	epicurious	Doc2Vec	Sugars	61.43	59.41	30.03	25.29
recipebox	epicurious	Doc2Vec	Sodium	98.2	92.95	49.55	33.25
recipebox	epicurious	BERT	Fat	49.91	46.04	10.14	8.68
recipebox	epicurious	BERT	Protein	80.94	76.55	41.37	22.24
recipebox	epicurious	BERT	Saturated fat	91.92	89.71	15.54	11.08
recipebox	epicurious	BERT	Sugars	64.58	62.99	39.47	25.76
recipebox	epicurious	BERT	Sodium	99.26	99.05	77.75	43.37

Table B.1 continued from previous page

Train	Test	Embedding	Target	Max	Avg	Mean	Median
recipebox	saladrecipes	Word2Vec	Fat	20.95	17.63	6.01	4.77
recipebox	saladrecipes	Word2Vec	Protein	81.88	73.82	24.99	17.25
recipebox	saladrecipes	Word2Vec	Saturated fat	83.82	78.43	11.59	7.84
recipebox	saladrecipes	Word2Vec	Sugars	67.72	62.82	20.83	27.83
recipebox	saladrecipes	Word2Vec	Sodium	98.55	94.83	6.31	4.65
recipebox	saladrecipes	GloVe	Fat	36.79	33.21	8.41	6.96
recipebox	saladrecipes	GloVe	Protein	61.62	58.04	28.09	16.93
recipebox	saladrecipes	GloVe	Saturated fat	89.46	86.87	13.92	9.86
recipebox	saladrecipes	GloVe	Sugars	67.61	62.39	27.39	25.93
recipebox	saladrecipes	GloVe	Sodium	98.87	93.26	41.85	31.77
recipebox	saladrecipes	Doc2Vec	Fat	20.08	15.35	6.01	4.77
recipebox	saladrecipes	Doc2Vec	Protein	70.11	64.11	24.99	17.25
recipebox	saladrecipes	Doc2Vec	Saturated fat	80.74	70.95	11.59	7.84
recipebox	saladrecipes	Doc2Vec	Sugars	62.55	57.87	20.83	27.83
recipebox	saladrecipes	Doc2Vec	Sodium	98.55	96.82	6.31	4.65
recipebox	saladrecipes	BERT	Fat	36.79	33.21	8.41	6.96
recipebox	saladrecipes	BERT	Protein	61.62	58.04	28.09	16.93
recipebox	saladrecipes	BERT	Saturated fat	89.46	86.87	13.92	9.86
recipebox	saladrecipes	BERT	Sugars	67.61	62.39	27.39	25.93
recipebox	saladrecipes	BERT	Sodium	98.87	93.26	41.85	31.77
recipebox	yummly28K	Word2Vec	Fat	25.1	21.64	6.69	5.25
recipebox	yummly28K	Word2Vec	Protein	47.92	42.69	14.8	11.61
recipebox	yummly28K	Word2Vec	Saturated fat	88.76	84.32	12.3	8.65
recipebox	yummly28K	Word2Vec	Sugars	71.51	67.2	15.3	26.11
recipebox	yummly28K	Word2Vec	Sodium	98.55	85.31	5.96	20.16
recipebox	yummly28K	GloVe	Fat	42.46	38.86	9.22	7.7
recipebox	yummly28K	GloVe	Protein	56.53	54.62	29.12	16.82
recipebox	yummly28K	GloVe	Saturated fat	92.67	89.42	14.7	10.54
recipebox	yummly28K	GloVe	Sugars	68.79	63.17	29.57	25.3
recipebox	yummly28K	GloVe	Sodium	98.98	98.18	53.7	40.8
recipebox	yummly28K	Doc2Vec	Fat	23.8	18.94	6.69	5.25
recipebox	yummly28K	Doc2Vec	Protein	41.68	36.06	14.8	11.61
recipebox	yummly28K	Doc2Vec	Saturated fat	85.09	79.06	12.3	8.65
recipebox	yummly28K	Doc2Vec	Sugars	62.1	57.21	15.3	26.11
recipebox	yummly28K	Doc2Vec	Sodium	98.55	85.76	5.96	20.16
recipebox	yummly28K	BERT	Fat	42.46	38.86	9.22	7.7
recipebox	yummly28K	BERT	Protein	56.53	54.62	29.12	16.82
recipebox	yummly28K	BERT	Saturated fat	92.67	89.42	14.7	10.54
recipebox	yummly28K	BERT	Sugars	68.79	63.17	29.57	25.3
recipebox	yummly28K	BERT	Sodium	98.98	98.18	53.7	40.8

References

[1] M. F. Ijaz, M. Attique, and Y. Son, "Data-Driven Cervical Cancer Prediction Model with Outlier Detection and Over-Sampling Methods," *Sensors*, vol. 20, no. 10, p. 2809, 2020, Publisher: Multidisciplinary Digital Publishing Institute.

[2] World Health Organization, *Diet, nutrition, and the prevention of chronic diseases: report of a joint WHO/FAO expert consultation*. World Health Organization: Geneva, Switzerland, 2003, vol. 916.

[3] F. Branca, A. Demaio, E. Udomkesmalee, *et al.*, "A new nutrition manifesto for a new nutrition reality," *The Lancet*, vol. 395, no. 10217, pp. 8–10, 2020, Publisher: Elsevier.

[4] B. Keeley, C. Little, and E. Zuehlke, "The State of the World's Children 2019: Children, Food and Nutrition–Growing Well in a Changing World.," *UNICEF: New York, NY, USA*, 2019, Publisher: ERIC.

[5] H.-O. P. Mbow, A. Reisinger, J. Canadell, and P. O'Brien, "Special Report on climate change, desertification, land degradation, sustainable land management, food security, and greenhouse gas fluxes in terrestrial ecosystems (SR2)," *Ginevra, IPCC*, 2017.

[6] W. Willett, J. Rockström, B. Loken, *et al.*, "Food in the Anthropocene: The EAT–Lancet Commission on healthy diets from sustainable food systems," *The Lancet*, vol. 393, no. 10170, pp. 447–492, 2019, Publisher: Elsevier.

[7] M. Alexander and J. Anderson, "The Hansard corpus, 1803-2003," 2012, Publisher: University of Glasgow.

[8] https://www.hansard-corpus.org. "Hansard corpus." (2022), [Online]. Available: `https://www.hansard-corpus.org`.

[9] K. Donnelly *et al.*, "SNOMED-CT: The advanced terminology and coding system for eHealth," *Studies in health technology and informatics*, vol. 121, p. 279, 2006, Publisher: IOS Press; 1999.

[10] "Unified Medical Language System." (2022), [Online]. Available: `http://%20umlsks.nlm.nih.gov`.

[11] *UFAL medical corpus v. 1.0*. [Online]. Available: `https://ufal.mff.cuni.cz/ufal_medical_corpus`.

[12] P. Rajpurkar, E. Chen, O. Banerjee, and E. J. Topol, "AI in health and medicine," *Nature Medicine*, pp. 1–8, 2022, Publisher: Nature Publishing Group.

[13] C. Krupitzer and A. Stein, "Food Informatics—Review of the Current State-of-the-Art, Revised Definition, and Classification into the Research Landscape," *Foods*, vol. 10, no. 11, p. 2889, 2021, Publisher: Multidisciplinary Digital Publishing Institute.

[14] M. Mazidi, E. R. Leeming, J. Merino, *et al.*, "Diet and lifestyle behaviour disruption related to the pandemic was varied and bidirectional among US and UK adults participating in the ZOE COVID Study," *Nature Food*, vol. 2, no. 12, pp. 957–969, 2021, Publisher: Nature Publishing Group.

[15] T. Eftimov, G. Popovski, M. Petković, B. Koroušić Seljak, and D. Kocev, "COVID-19 pandemic changes the food consumption patterns," *Trends in food science & technology*, vol. 104, pp. 268–272, 2020, Publisher: Elsevier.

[16] J. Wiens, S. Saria, M. Sendak, *et al.*, "Do no harm: A roadmap for responsible machine learning for health care," *Nature medicine*, vol. 25, no. 9, pp. 1337–1340, 2019, Publisher: Nature Publishing Group.

[17] Food and Administration, Drug and others, "Proposed regulatory framework for modifications to artificial intelligence/machine learning (AI/ML)-based software as a medical device (SaMD)," 2019, Publisher: Department of Health and Human Services (United States).

[18] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013, Publisher: IEEE.

[19] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[20] 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*, Lake Tahoe, Nevada, USA, Dec. 2013, pp. 3111–3119.

[21] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[22] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, Beijing, China, Jun. 2014, pp. 1188–1196.

[23] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[25] S. Moen and T. S. S. Ananiadou, "Distributional semantics resources for biomedical text processing," *Proceedings of LBM*, pp. 39–44, 2013.

[26] O. Papakyriakopoulos, S. Hegelich, J. C. M. Serrano, and F. Marco, "Bias in word embeddings," in *Proceedings of the 2020 conference on fairness, accountability, and transparency*, 2020, pp. 446–457.

[27] A. L. Beam, B. Kompa, I. Fried, *et al.*, "Clinical concept embeddings learned from massive sources of medical data," 2018.

[28] K. Agarwal, T. Eftimov, R. Addanki, S. Choudhury, S. Tamang, and R. Rallo, "Snomed2Vec: Random Walk and Poincare Embeddings of a Clinical Knowledge Base for Healthcare Analytics," *arXiv preprint arXiv:1907.08650*, 2019.

[29] D. Li and M. J. Zaki, "Reciptor: An effective pretrained model for recipe representation learning," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1719–1727.

[30] Y. Tian, C. Zhang, R. Metoyer, and N. V. Chawla, "Recipe representation learning with networks," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 1824–1833.

[31] M. Carvalho, R. Cadène, D. Picard, L. Soulier, N. Thome, and M. Cord, "Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 35–44.

[32] A. Salvador, N. Hynes, Y. Aytar, *et al.*, "Learning cross-modal embeddings for cooking recipes and food images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3020–3028.

[33] H. Wang, D. Sahoo, C. Liu, E.-p. Lim, and S. C. Hoi, "Learning cross-modal embeddings with adversarial networks for cooking recipes and food images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 572–11 581.

[34] A. Salvador, E. Gundogdu, L. Bazzani, and M. Donoser, "Revamping cross-modal recipe retrieval with hierarchical transformers and self-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 475–15 484.

[35] M. Gharibi, A. Zachariah, and P. Rao, "Foodkg: A tool to enrich knowledge graphs using machine learning techniques," *Frontiers in big Data*, vol. 3, p. 12, 2020.

[36] W. Min, C. Liu, L. Xu, and S. Jiang, "Applications of knowledge graphs for food science and industry," *Patterns*, vol. 3, no. 5, p. 100 484, 2022.

[37] Z. Obermeyer and E. J. Topol, "Artificial intelligence, bias, and patients' perspectives," *The Lancet*, vol. 397, no. 10289, p. 2038, 2021, Publisher: Elsevier.

[38] H. Blockeel, S. Dzeroski, J. Struyf, and B. Zenko, *Predictive Clustering*, 2019.

[39] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "P-NUT: Predicting NUTrient Content from Short Text Descriptions," *Mathematics*, vol. 8, no. 10, p. 1811, 2020, Publisher: Multidisciplinary Digital Publishing Institute.

[40] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Exploring Knowledge Domain Bias on a Prediction Task for Food and Nutrition Data," in *2020 IEEE International Conference on Big Data (Big Data)*, IEEE, 2020, pp. 3563–3572.

[41] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Domain Heuristic Fusion of Multi-Word Embeddings for Nutrient Value Prediction," *Mathematics*, vol. 9, no. 16, p. 1941, 2021, Publisher: Multidisciplinary Digital Publishing Institute.

[42] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Nutrient prediction on recipes – use case on heterogeneous recipe datasets," in *Abstract book for the ISBNPA 2022 Annual Meeting in Phoenix,Arizona. Published by: International Society of Behavioral Nutrition and Physical Activity*, Phoenix, Arizona, USA, May 2022.

[43] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Predefined domain specific embeddings of food concepts and recipes," in *International Society of Behavioral Nutrition and Physical Activity ISBNPA 2022 Annual Meeting in Phoenix,Arizona*, Phoenix, Arizona, USA, May 2022.

[44] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Predefined domain specific embeddings of food concepts and recipes: A case study on heterogeneous recipe datasets," in *2022 IEEE International Conference on Big Data (Big Data)*, IEEE, 2022.

[45] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Defining the trust of generalizing the knowledge learned by predictive modeling on heterogeneous recipe data," *Expert systems with applications*, 2022, Submited; Publisher: Elsevier.

[46] European commission health and consumers directorate-general, *Guidance document for competent authorities for the control of compliance with eu legislation on: Regulation (EU) No 1169/2011 of the European Parliament and of the Council of 25 October 2011 on the provision of food information to consumers, amending Regulations (EC) No 1924/2006 and (EC) No 1925/2006 of the European Parliament and of the Council, and repealing Commission Directive 87/250/EEC, Council Directive 90/496/EEC, Commission Directive 1999/10/EC, Directive 2000/13/EC of the European Parliament and of the Council, Commission Directives 2002/67/EC and 2008/5/EC and Commission Regulation (EC) No 608/2004Devlin*, Dec. 2012. [Online]. Available: `https://ec.europa.eu/food/sites/food/files/safety/docs/labelling_nutrition-supplements-guidance_tolerances_1212_en.pdf`.

[47] European Commission, "Regulation (EU) No 1169/2011 of the European Parliament and of the Council of 25 October 2011 on the provision of food information to consumers, amending Regulations (EC) No 1924/2006 and (EC) No 1925/2006 of the European Parliament and of the Council, and repealing Commission Directive 87/250/EEC, Council Directive 90/496/EEC, Commission Directive 1999/10/EC, Directive 2000/13/EC of the European Parliament and of the Council, Commission Directives 2002/67/EC and 2008/5/EC and Commission Regulation (EC) No 608/2004," *Off. J. Eur. Union L*, vol. 304, pp. 18–63, 2011.

[48] T. Eftimov, G. Petelin, G. Cenikj, *et al.*, "Less is more: Selecting the right benchmarking set of data for time series classification," *Expert Systems with Applications*, vol. 198, p. 116 871, 2022.

[49] European Food Safety Authority, *The food classification and description system FoodEx2 (revision 2)*. [Online]. Available: `https://efsa.onlinelibrary.wiley.com/doi/epdf/10.2903/sp.efsa.2015.EN-804`.

[50] European Food Safety Authority, "Standard sample description ver. 2.0," *EFSA Journal*, vol. 11, no. 10, p. 3424, 2013.

[51] T. Eftimov, P. Korošec, and B. Koroušić Seljak, "StandFood: Standardization of foods using a semi-automatic system for classifying and describing foods according to FoodEx2," *Nutrients*, vol. 9, no. 6, p. 542, 2017, Publisher: Multidisciplinary Digital Publishing Institute.

[52] D. E. Rumelhart, G. E. Hinton, R. J. Williams, *et al.*, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.

[53] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[54] T. Mikolov, "Statistical language models based on neural networks," *Presentation at Google, Mountain View, 2nd April*, vol. 80, 2012.

[55] C. Caracciolo, A. Stellato, S. Rajbahndari, *et al.*, "Thesaurus maintenance, alignment and publication as linked data: The AGROVOC use case," *International Journal of Metadata, Semantics and Ontologies*, vol. 7, no. 1, pp. 65–75, 2012.

[56] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," in *Twenty-Second International Joint Conference on Artificial Intelligence*, Catalonia, Spain, Jul. 2011.

[57] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, Bellevue, WA, USA, Jul. 2011, pp. 129–136.

[58] X. Glorot, A. Bordes, and Y. Bengio, "Domain adaptation for large-scale sentiment classification: A deep learning approach," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 513–520.

[59] P. D. Turney, "Distributional semantics beyond words: Supervised learning of analogy and paraphrase," *Transactions of the Association for Computational Linguistics*, vol. 1, pp. 353–366, 2013.

[60] P. D. Turney and P. Pantel, "From frequency to meaning: Vector space models of semantics," *Journal of artificial intelligence research*, vol. 37, pp. 141–188, 2010.

[61] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Atlanta, GA, USA, Jun. 2013, pp. 746–751.

[62] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[63] M. Habibi, L. Weber, M. Neves, D. L. Wiegandt, and U. Leser, "Deep learning with word embeddings improves biomedical named entity recognition," *Bioinformatics*, vol. 33, no. 14, pp. i37–i48, 2017, Publisher: Oxford University Press.

[64] A. Drozd, A. Gladkova, and S. Matsuoka, "Word embeddings, analogies, and machine learning: Beyond king-man+ woman= queen," in *Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers*, Osaka, Japan, Dec. 2016, pp. 3519–3530.

[65] V. Tshitoyan, J. Dagdelen, L. Weston, *et al.*, "Unsupervised word embeddings capture latent knowledge from materials science literature," *Nature*, vol. 571, no. 7763, pp. 95–98, 2019, Publisher: Nature Publishing Group.

[66] G. A. Blog, *Open sourcing bert: State-of-the-art pre-training for natural language processing*. [Online]. Available: `https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html`.

[67] A. Rogers, O. Kovaleva, and A. Rumshisky, "A primer in bertology: What we know about how bert works," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 2020.

[68] Y. Zhu, R. Kiros, R. Zemel, *et al.*, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 19–27.

[69] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[70] R. Horev, "Bert explained: State of the art language model for nlp," *Towards Data Science*, vol. 10, 2018.

[71] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Advances in neural information processing systems*, 2017, pp. 6338–6347.

[72] Z. Yang, W. Cohen, and R. Salakhudinov, "Revisiting Semi-Supervised Learning with Graph Embeddings," M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, Jun. 2016, pp. 40–48. [Online]. Available: `http://proceedings.mlr.press/v48/yanga16.html`.

[73] P. Ristoski and H. Paulheim, "Rdf2vec: Rdf graph embeddings for data mining," in *International Semantic Web Conference*, Hyogo, Japan: Springer, Oct. 2016, pp. 498–514.

[74] T. Eftimov, G. Popovski, E. Valenčič, and B. Koroušić Seljak, "FoodEx2vec: New foods' representation for advanced food data analysis," *Food and Chemical Toxicology*, vol. 138, p. 111 169, 2020, Publisher: Elsevier.

[75] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, 4. Springer, 2006, vol. 4.

[76] P. N. Stuart J. Russell, *Artificial Intelligence: A Modern Approach, Third Edition*. Prentice Hall, 2010.

[77] D. A. Freedman, *Statistical models: theory and practice*. cambridge university press, 2009.

[78] A. C. Rencher and W. F. Christensen, "Chapter 10, multivariate regression–section 10.1, introduction," *Methods of multivariate analysis, Wiley Series in Probability and Statistics*, vol. 709, p. 19, 2012.

[79] H. L. Seal, "Studies in the history of probability and statistics. xv the historical development of the gauss linear model," *Biometrika*, vol. 54, no. 1-2, pp. 1–24, 1967.

[80] F. Santosa and W. W. Symes, "Linear inversion of band-limited reflection seismograms," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 4, pp. 1307–1330, 1986.

[81] R. Tibshirani, "Regression shrinkage and selection via the lasso," 2011.

[82] D. E. Hilt and D. W. Seegrist, *Ridge, a computer program for calculating ridge regression estimates*. Department of Agriculture, Forest Service, Northeastern Forest Experiment, 1977.

[83] Y. Haitovsky, "Multicollinearity in regression analysis: Comment," *The Review of economics and statistics*, pp. 486–489, 1969.

[84] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[85] A. E. Hoerl and R. W. Kennard, "Ridge regression: Applications to nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 69–82, 1970.

[86] J. V. Beck and K. J. Arnold, *Parameter estimation in engineering and science*. James Beck, 1977.

[87] M. Merriman, *A List of Writings Relating to the Method of Least Squares: With Historical and Critical Notes*. Academy, 1877, vol. 4.

[88] M. H. Gruber, *Improving efficiency by shrinkage: the James-Stein and ridge regression estimators*. Routledge, 2017.

[89] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20 150 202, 2016.

[90] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the royal statistical society: series B (statistical methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[91] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[92] X. Wu, V. Kumar, J. Ross Quinlan, *et al.*, "Top 10 algorithms in data mining," *Knowledge and information systems*, vol. 14, no. 1, pp. 1–37, 2008.

[93] L. Rokach and O. Maimon, "Data mining with decision tree; series in machine perception and artificial intelligence," *World Scientific*, vol. 81, pp. 61–62, 2014.

[94] T. K. Ho, "Random decision forests," in *Proceedings of 3rd international conference on document analysis and recognition*, IEEE, vol. 1, 1995, pp. 278–282.

[95] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 8, pp. 832–844, 1998.

[96] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[97] L. Hardesty, "Explained: Neural networks," *MIT News*, vol. 14, 2017.

[98] A. Brahme, *Comprehensive biomedical physics*. Newnes, 2014.

[99] A. Gulli and S. Pal, *Deep learning with Keras*. Packt Publishing Ltd, 2017.

[100] G. Hinton and T. J. Sejnowski, *Unsupervised learning: foundations of neural computation*. MIT press, 1999.

[101] J. MacQueen, "Classification and analysis of multivariate observations," in *5th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281–297.

[102] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[103] P. Bradley, O. Mangasarian, and W. Street, "Clustering via concave minimization," *Advances in neural information processing systems*, vol. 9, 1996.

[104] L. Kaufman and P. J. Rousseeuw, "Partitioning around medoids (program pam)," *Finding groups in data: an introduction to cluster analysis*, vol. 344, pp. 68–125, 1990.

[105] L. Kaufman, "Rousseeuw pj," *Partitioning around medoids (program pam), Finding groups in data: an introduction to cluster analysis*, pp. 68–125, 1990.

[106] M. Van der Laan, K. Pollard, and J. Bryan, "A new partitioning around medoids algorithm," *Journal of Statistical Computation and Simulation*, vol. 73, no. 8, pp. 575–584, 2003, Publisher: Taylor & Francis.

[107] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[108] D. H. Wolpert, W. G. Macready, *et al.*, "No free lunch theorems for search," Technical Report SFI-TR-95-02-010, Santa Fe Institute, Tech. Rep., 1995.

[109] D. H. Wolpert, "The lack of a priori distinctions between learning algorithms," *Neural computation*, vol. 8, no. 7, pp. 1341–1390, 1996.

[110] W. M. Rand, J. A. Pennington, S. P. Murphy, J. C. Klensin, *et al.*, *Compiling data for food composition data bases*. United Nations University Press Tokyo, Japan: 1991.

[111] H. Greenfield and D. A. Southgate, *Food composition data: production, management, and use*. Food and Agriculture Org.: Rome, Italy, 2003, ISBN: 978-92-5-104949-5.

[112] S. F. Schakel, I. M. Buzzard, and S. E. Gebhardt, "Procedures for estimating nutrient values for food composition databases," *Journal of food composition and analysis*, vol. 10, no. 2, pp. 102–114, 1997, Publisher: Elsevier.

[113] I. Kononenko, "Machine learning for medical diagnosis: History, state of the art and perspective," *Artificial Intelligence in medicine*, vol. 23, no. 1, pp. 89–109, 2001.

[114] A. Sharma and R. Rani, "A systematic review of applications of machine learning in cancer prediction and diagnosis," *Archives of Computational Methods in Engineering*, vol. 28, no. 7, pp. 4875–4896, 2021.

[115] M. A. Naji, S. El Filali, K. Aarika, E. H. Benlahmar, R. A. Abdelouhahid, and O. Debauche, "Machine learning algorithms for breast cancer prediction and diagnosis," *Procedia Computer Science*, vol. 191, pp. 487–492, 2021.

[116] Y. Kumar, S. Gupta, R. Singla, and Y.-C. Hu, "A systematic review of artificial intelligence techniques in cancer prediction and diagnosis," *Archives of Computational Methods in Engineering*, pp. 1–28, 2021.

[117] E. Erdem and F. Bozkurt, "A comparison of various supervised machine learning techniques for prostate cancer prediction," *Avrupa Bilim ve Teknoloji Dergisi*, no. 21, pp. 610–620, 2021.

[118] S. Ray *et al.*, "A survey on application of machine learning algorithms in cancer prediction and prognosis," in *Data Management, Analytics and Innovation*, Springer, 2021, pp. 349–361.

[119] A. Alhazmi, Y. Alhazmi, A. Makrami, *et al.*, "Application of artificial intelligence and machine learning for prediction of oral cancer risk," *Journal of Oral Pathology & Medicine*, vol. 50, no. 5, pp. 444–450, 2021.

[120] T. A. Assegie, R. L. Tulasi, and N. K. Kumar, "Breast cancer prediction model with decision tree and adaptive boosting," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 1, p. 184, 2021.

[121] M. A. Myszczynska, P. N. Ojamies, A. Lacoste, *et al.*, "Applications of machine learning to diagnosis and treatment of neurodegenerative diseases," *Nature Reviews Neurology*, vol. 16, no. 8, pp. 440–456, 2020.

[122] Z. K. Senturk, "Early diagnosis of parkinson's disease using machine learning algorithms," *Medical hypotheses*, vol. 138, p. 109 603, 2020.

[123] H. Mohammad-Rahimi, M. Nadimi, A. Ghalyanchi-Langeroudi, M. Taheri, and S. Ghafouri-Fard, "Application of machine learning in diagnosis of covid-19 through x-ray and ct images: A scoping review," *Frontiers in cardiovascular medicine*, vol. 8, p. 638 011, 2021.

[124] R. Yunus, O. Arif, H. Afzal, *et al.*, "A framework to estimate the nutritional value of food in real time using deep learning techniques," *IEEE Access*, vol. 7, pp. 2643–2652, 2018, Publisher: IEEE.

[125] L. Jiang, B. Qiu, X. Liu, C. Huang, and K. Lin, "DeepFood: Food Image Analysis and Dietary Assessment via Deep Model," *IEEE Access*, vol. 8, pp. 47 477–47 489, 2020, Publisher: IEEE.

[126] P. Pouladzadeh, S. Shirmohammadi, and R. Al-Maghrabi, "Measuring calorie and nutrition from food image," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 8, pp. 1947–1956, 2014, Publisher: IEEE.

[127] T. Ege and K. Yanai, "Image-based food calorie estimation using recipe information," *IEICE TRANSACTIONS on Information and Systems*, vol. 101, no. 5, pp. 1333–1341, 2018, Publisher: The Institute of Electronics, Information and Communication Engineers.

[128] F. T. S. Gunawan, M. Kartiwi, N. Abd Malik, and N. Ismail, "Food intake calorie prediction using generalized regression neural network," in *2018 IEEE 5th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, IEEE, 2018, pp. 1–4.

[129] R. Ruede, V. Heusser, L. Frank, A. Roitberg, M. Haurilet, and R. Stiefelhagen, "Multi-task learning for calorie prediction on a novel large-scale recipe dataset enriched with nutritional information," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 4001–4008. DOI: 10.1109/ICPR48806.2021.9412839.

[130] N. Darapaneni, V. Singh, Y. S. Tarkar, *et al.*, "Food image recognition and calorie prediction," in *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, 2021, pp. 1–6. DOI: 10.1109/IEMTRONICS52119.2021.9422510.

[131] "FoodAI." (2022), [Online]. Available: https://foodai.org/.

[132] "Clarifai Food Item Recognition." (2022), [Online]. Available: https://clarifai.com/clarifai/main/models/food-item-recognition.

[133] "Snap it Lose it." (2022), [Online]. Available: https://www.loseit.com/snapit/.

[134] "Calorie Mama." (2022), [Online]. Available: https://www.caloriemama.ai/.

[135] "FoodVisor." (2022), [Online]. Available: https://www.foodvisor.io/en/.

[136] "BiteSnap." (2022), [Online]. Available: https://getbitesnap.com/.

[137] "LogMeal." (2022), [Online]. Available: https://logmeal.es.

[138] "Samsung Health (S-Health)." (2022), [Online]. Available: https://health.apps.samsung.com/terms.

[139] "MyFitnessPal." (2022), [Online]. Available: https://www.myfitnesspal.com/.

[140] G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Comparing Semantic and Nutrient Value Similarities of Recipes," in *2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA: IEEE, Dec. 2019, pp. 5131–5139.

[141] P. Arora, S. Varshney, *et al.*, "Analysis of k-means and k-medoids algorithm for big data," *Procedia Computer Science*, vol. 78, pp. 507–512, 2016.

[142] M. Gabrijelčič Blenkuš, M. Gregorič, B. Tivadar, *et al.*, "Prehrambene navade odraslih prebivalcev slovenije z vidika varovanja zdravja," *Ljubljana: Univerza v Ljubljani, Pedagoška fakulteta*, 2009.

[143] European Food Safety Authority, *Use of the efsa comprehensive european food consumption database in exposure assessment*, 2011.

[144] T. Korenius, J. Laurikkala, K. Järvelin, and M. Juhola, "Stemming and lemmatization in the clustering of finnish text documents," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, Washington, DC, USA, 2004, pp. 625–633.

[145] R. Rehurek, P. Sojka, *et al.*, "Gensim—statistical semantics in python," *NLP Centre, Faculty of Informatics, Masaryk University*, 2011.

[146] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011, Publisher: JMLR. org.

[147] Y. Sun, S. Wang, Y. Li, *et al.*, "Ernie: Enhanced representation through knowledge integration," *arXiv preprint arXiv:1904.09223*, 2019.

[148] European Food Safety Authority. (2019), [Online]. Available: `https://www.efsa.europa.eu/en/data/food-consumption-data` (visited on 05/11/2020).

[149] UK Food Standards Agency, "Guide to creating a front of pack (FoP) nutrition label for pre-packed products sold through retail outlets," *Food Standards Agency*, 2013.

[150] J. Marin, A. Biswas, F. Ofli, *et al.*, "Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images," *IEEE transactions on pattern analysis and machine intelligence*, 2019, Publisher: IEEE.

[151] "GloVe: Global Vectors for Word Representation." (2022), [Online]. Available: `https://nlp.stanford.edu/projects/glove/`.

[152] H. Greenfield and D. A. T. Southgate, *Food Composition Data: Production, Management, and Use*, en. Food & Agriculture Org., 2003, Google-Books-ID: KQRzKDr9bgcC, ISBN: 978-92-5-104949-5.

[153] M. Machackova, A. Giertlova, J. Porubska, M. Roe, C. Ramos, and P. Finglas, "EuroFIR Guideline on calculation of nutrient content of foods for food business operators," *Food chemistry*, vol. 238, pp. 35–41, 2018, Publisher: Elsevier.

[154] M. Jiang, T. Sanger, and X. Liu, "Combining contextualized embeddings and prior knowledge for clinical named entity recognition: Evaluation study," *JMIR medical informatics*, vol. 7, no. 4, e14850, 2019, Publisher: JMIR Publications Inc., Toronto, Canada.

[155] Y. Li, X. Wang, L. Hui, *et al.*, "Chinese Clinical Named Entity Recognition in Electronic Medical Records: Development of a Lattice Long Short-Term Memory Model With Contextualized Character Representations," *JMIR Medical Informatics*, vol. 8, no. 9, e19848, 2020, Publisher: JMIR Publications Inc., Toronto, Canada.

[156] L. Rasmy, Y. Xiang, Z. Xie, C. Tao, and D. Zhi, "Med-BERT: Pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction," *npj Digital Medicine*, vol. 4, no. 1, pp. 1–13, 2021, Publisher: Nature Publishing Group.

[157] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987, Publisher: Elsevier.

[158] G. Popovski, S. Kochev, B. Korousic-Seljak, and T. Eftimov, "FoodIE: A Rule-based Named-entity Recognition Method for Food Information Extraction.," in *ICPRAM*, 2019, pp. 915–922.

[159] G. Cenikj, G. Popovski, R. Stojanov, B. Koroušić Seljak, and T. Eftimov, "BuTTER: BidirecTional LSTM for Food Named-Entity Recognition," in *2020 IEEE International Conference on Big Data (Big Data)*, IEEE, 2020, pp. 3550–3556.

[160] L. Wright. "Cooking measurement conversion tables." (2022), [Online]. Available: `https://www.saga.co.uk/magazine/food/cooking-tips/cooking-measurement-conversion-tables`.

[161] G. Ispirova, T. Eftimov, B. Korousic-Seljak, and P. Korosec, "Mapping Food Composition Data from Various Data Sources to a Domain-Specific Ontology.," in *KEOD*, 2017, pp. 203–210.

[162] G. Popovski, G. Ispirova, N. Hadzi-Kotarova, E. Valencic, T. Eftimov, and B. Korousic-Seljak, "Food Data Integration by using Heuristics based on Lexical and Semantic Similarities.," in *HEALTHINF*, 2020, pp. 208–216.

[163] G. Ispirova, G. Popovski, N. Hadzi-Kotarova, E. Valenčič, T. Eftimov, and B. Koroušić Seljak, "Food Data Normalization Using Lexical and Semantic Similarities Heuristics," in *13th International Joint Conference, BIOSTEC 2020*, ser. Communications in Computer and Information Science, Springer, 2021.

[164] Meredith Food Group. "All recipes website." (2022), [Online]. Available: `https://www.allrecipes.com/`.

[165] "Yummly website." (2022), [Online]. Available: `https://www.yummly.com/`.

[166] "Epicurious website." (2022), [Online]. Available: `https://www.epicurious.com/`.

[167] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[168] V. Gulshan, L. Peng, M. Coram, *et al.*, "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *Jama*, vol. 316, no. 22, pp. 2402–2410, 2016.

[169] A. L. Beam and I. S. Kohane, "Translating artificial intelligence into clinical care," *Jama*, vol. 316, no. 22, pp. 2368–2369, 2016.

[170] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[171] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[172] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[173] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[174] A. L. Beam, B. Kompa, A. Schmaltz, *et al.*, "Clinical concept embeddings learned from massive sources of multimodal medical data," in *PACIFIC SYMPOSIUM ON BIOCOMPUTING 2020*, World Scientific, 2019, pp. 295–306.

[175] E. Choi, M. T. Bahadori, E. Searles, *et al.*, "Multi-layer representation learning for medical concepts," in *proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1495–1504.

[176] Y. Tian, C. Zhang, Z. Guo, Y. Ma, R. Metoyer, and N. V. Chawla, "Recipe2vec: Multi-modal recipe representation learning with graph neural networks," *arXiv preprint arXiv:2205.12396*, 2022.

[177] Department of Agriculture, US, *USDA, Food Composition Database*. 2021. [Online]. Available: `https://fdc.nal.usda.gov`.

[178] "National health and nutrition examination survey (NHANES)." (2022), [Online]. Available: `https://www.cdc.gov/nchs/nhanes/index.htm`.

[179] E. Gőbel, I. Mills, and A. Wallard, "The international system of units (si)," 2006.

[180] D. B. Newell, E. Tiesinga, *et al.*, "The international system of units (si)," *NIST Special Publication*, vol. 330, pp. 1–138, 2019.

[181] J. Turner, "Interpretation of the international system of units (the metric system of measurement) for the united states," *Federal Register*, vol. 73, no. 96, pp. 28 432–3,

[182] J. F. Krüger, *Completeïnternational handbook of the coins, masses and weights of all countries of the world....* G. basse, 1830.

[183] W. Henry, *The elements of experimental chemistry*. Robert Desilver, 1831, vol. 2.

[184] J. S. T. Gehler, *Johann Samuel Traugott Gehler's Physikalisches wörterbunch*, 3. EB Schurckert, 1844, vol. 10.

[185] S. Han, "Googletrans 3.0. 0," *PyPI Library*, 2020.

[186] Python Software Foundation, *Python 3.10.6 documentation*, (accessed on 11 August 2022), 2022. [Online]. Available: `https://docs.python.org/3/library/string.html` (visited on 08/11/2022).

[187] A. V. Aho, "Algorithms for finding patterns in strings, handbook of theoretical computer science vol a," *A, ed. J. van Leeuwen, ElsevierSciencePublishersB*, vol. 1990, pp. 257–297, 1990.

[188] R. Mitkov, *The Oxford handbook of computational linguistics*. Oxford University Press, 2022.

[189] G. Ispirova, G. Cenikj, M. Ogrinc, *et al.*, "Cafeteriafcd corpus: Food consumption data annotated with regard to different food semantic resources," *Foods*, vol. 11, no. 17, p. 2684, 2022.

[190] V. I. Levenshtein *et al.*, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, Soviet Union, vol. 10, 1966, pp. 707–710.

[191] Y. Chung, P. J. Haas, E. Upfal, and T. Kraska, "Unknown examples & machine learning model generalization," *arXiv preprint arXiv:1808.08294*, 2018.

[192] J. P. Miller, R. Taori, A. Raghunathan, *et al.*, "Accuracy on the line: On the strong correlation between out-of-distribution and in-distribution generalization," in *International Conference on Machine Learning*, PMLR, 2021, pp. 7721–7735.

[193] G. Cenikj, R. D. Lang, A. P. Engelbrecht, C. Doerr, P. Korošec, and T. Eftimov, "Selector: Selecting a representative benchmark suite for reproducible statistical comparison," *arXiv preprint arXiv:2204.11527*, 2022.

[194] L. Kaufman and P. J. Rousseeuw, *An introduction to cluster analysis*. John Wiley and Sons, Incorporated, 1990.

[195] *Selecting the number of clusters with silhouette analysis on KMeans clustering*, 2022. [Online]. Available: `https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html` (visited on 06/03/2022).

[196] "Food Nutrition Security Cloud (FNS-Cloud)." (2022), [Online]. Available: `https://www.fns-cloud.eu/` (visited on 06/03/2022).

[197] "METROFOOD-RI Metrology in Food and Nutrition." (2022), [Online]. Available: `https://www.metrofood.eu/` (visited on 06/03/2022).

[198] "FishEUTrust Europen project." (2022), [Online]. Available: `https://fisheutrust.org/` (visited on 06/03/2022).

[199] "COMFOCUS – European project." (2022), [Online]. Available: `https://comfocus.eu/` (visited on 06/03/2022).

[200] R. Sagar, *OpenAI Releases GPT-3, The Largest Model so Far*, 2022. [Online]. Available: `https://analyticsindiamag.com/open-ai-gpt-3-language-model/` (visited on 06/03/2022).

Bibliography

Publications Related to the Thesis

Journal Articles

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "P-NUT: Predicting NUTrient Content from Short Text Descriptions," *Mathematics*, vol. 8, no. 10, p. 1811, 2020, Publisher: Multidisciplinary Digital Publishing Institute.

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Domain Heuristic Fusion of Multi-Word Embeddings for Nutrient Value Prediction," *Mathematics*, vol. 9, no. 16, p. 1941, 2021, Publisher: Multidisciplinary Digital Publishing Institute.

T. Eftimov, G. Petelin, G. Cenikj, *et al.*, "Less is more: Selecting the right benchmarking set of data for time series classification," *Expert Systems with Applications*, vol. 198, p. 116 871, 2022.

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Defining the trust of generalizing the knowledge learned by predictive modeling on heterogeneous recipe data," *Expert systems with applications*, 2022, Submited; Publisher: Elsevier.

Conference Paper

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Comparing Semantic and Nutrient Value Similarities of Recipes," in *2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA: IEEE, Dec. 2019, pp. 5131–5139.

G. Ispirova, T. Eftimov, B. Korousic-Seljak, and P. Korosec, "Mapping Food Composition Data from Various Data Sources to a Domain-Specific Ontology.," in *KEOD*, 2017, pp. 203–210.

G. Popovski, G. Ispirova, N. Hadzi-Kotarova, E. Valencic, T. Eftimov, and B. Korousic-Seljak, "Food Data Integration by using Heuristics based on Lexical and Semantic Similarities.," in *HEALTHINF*, 2020, pp. 208–216.

G. Ispirova, G. Popovski, N. Hadzi-Kotarova, E. Valenčič, T. Eftimov, and B. Koroušić Seljak, "Food Data Normalization Using Lexical and Semantic Similarities Heuristics," in *13th International Joint Conference, BIOSTEC 2020*, ser. Communications in Computer and Information Science, Springer, 2021.

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Exploring Knowledge Domain Bias on a Prediction Task for Food and Nutrition Data," in *2020 IEEE International Conference on Big Data (Big Data)*, IEEE, 2020, pp. 3563–3572.

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Nutrient prediction on recipes – use case on heterogeneous recipe datasets," in *Abstract book for the ISBNPA 2022 Annual Meeting in Phoenix,Arizona. Published by: International Society of Behavioral Nutrition and Physical Activity*, Phoenix, Arizona, USA, May 2022.

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Predefined domain specific embeddings of food concepts and recipes," in *International Society of Behavioral Nutrition and Physical Activity ISBNPA 2022 Annual Meeting in Phoenix,Arizona*, Phoenix, Arizona, USA, May 2022.

G. Ispirova, T. Eftimov, and B. Koroušić Seljak, "Predefined domain specific embeddings of food concepts and recipes: A case study on heterogeneous recipe datasets," in *2022 IEEE International Conference on Big Data (Big Data)*, IEEE, 2022.

Biography

Gordana Ispirova was born on the 29th of July, 1994, in Strumica, North Macedonia. In 2012, she started her studies at the Faculty of Electrical Engineering and Information Technologies of Ss. Cyril and Methodius University in Skopje, North Macedonia. She received her Bachelor of Science in Engineering in the field of Computer Technologies and Engineering in September 2016 under the supervision of Prof. Dr. Sanja Velkova. During her undergraduate studies she held a state scholarship for talented students awarded by the Ministry of Education and Science of Macedonia.

In 2016, she started her master's studies in Information and Communication Technologies at the Jožef Stefan International Postgraduate School in Ljubljana, Slovenia. During her master's studies she was awarded the Ad Futura scholarship from the Public Scholarship, Development, Disability and Maintenance Fund of the Republic of Slovenia. She received her Master of Science in Information and Communication Technologies in September 2018, under the supervision of Prof. Dr. Barbara Koroušić Seljak, and co-supervision of Asst. Prof. Dr. Tome Eftimov.

In October 2018, she started her PhD studies in Information and Communication Technologies at the Jožef Stefan International Postgraduate School in Ljubljana, Slovenia, under the supervision of Prof. Dr. Barbara Koroušić Seljak, and co-supervision of Asst. Prof. Dr. Tome Eftimov. Over the course of her doctoral studies, she has been employed as a research assistant at the Computer Systems Department at the Jožef Stefan Institute and has taken part in several EU-funded projects: FNS-Cloud, COMFOCUS, METROFOOD (European Union's Horizon 2020 research and innovation programme), CAFETERIA (EFSA-funded project).

Her areas of research include Natural Language Processing, Machine Learning, Data Mining, Semantic Web, and Deep Learning. Currently, she is concentrating her research work on developing Natural Language Processing methodologies for food- and nutrition-related data, and Data Analysis using Machine Learning on big data. She has presented her work at many international conferences and workshops and has published papers in scientific journals with impact factors.