Food Data Integration by using Heuristics based on Lexical and Semantic Similarities

Gorjan Popovski\textsuperscript{1,2}\textsuperscript{a}, Gordana Ispirova\textsuperscript{1,2}\textsuperscript{b}, Nina Hadzi-Kotarova\textsuperscript{1}, Eva Valenčič\textsuperscript{1,2,4}\textsuperscript{c}, Tomе Eftimov\textsuperscript{2}\textsuperscript{d} and Barbara Koroušić Seljak\textsuperscript{2}\textsuperscript{e}

\textsuperscript{1}Jožef Stefan International Postgraduate School, 1000 Ljubljana, Slovenia
\textsuperscript{2}Computer Systems Department, Jožef Stefan Institute, 1000 Ljubljana, Slovenia
\textsuperscript{3}Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University, 1000 Skopje, North Macedonia
\textsuperscript{4}School of Health Sciences, Faculty of Health and Medicine, Priority Research Centre in Physical Activity and Nutrition, The University of Newcastle, Callaghan, Australia

Keywords: Data Normalization, Food Data Integration, Lexical Similarity, Semantic Similarity, Word Embeddings.

Abstract: With the rapidly growing food supply in the last decade, vast amounts of food-related data have been collected. To make this data inter-operable and equipped for analyses involving studying relations between food, as one of the main environmental and health outcomes, data coming from various data sources needs to be normalized. Food data can have varying sources and formats (food composition, food consumption, recipe data), yet the most familiar type is food product data, often misinterpreted due to marketing strategies of different producers and retailers. Several recent studies have addressed the problem of heterogeneous data by matching food products using lexical similarity between their English names. In this study, we address this problem, while considering a non-English, low researched language in terms of natural language processing, i.e. Slovenian. To match food products, we use our previously developed heuristic based on lexical similarity and propose two new semantic similarity heuristics based on word embeddings. The proposed heuristics are evaluated using a dataset with 438 ground truth pairs of food products, obtained by matching their EAN barcodes. Preliminary results show that the lexical similarity heuristic provides more promising results (75% accuracy), while the best semantic similarity model yields an accuracy of 62%.

1 INTRODUCTION

State-of-the-art data fusion approaches enable integration of various data sources to produce more consistent, usable, and accurate information than those provided by any individual data source. However, before using fused data for predictive modelling, we must find an efficient way of linking unstructured data attributes that are shared across the multiple data sources. To enable this, a data normalization process is required as a pre-processing step before starting with some further analyses (Pramanik and Hussain, 2019). By applying data normalization the same concepts that might be represented with different text descriptions (i.e. names) or standards are linked together.

There are many studies performed in the biomedical domain, where different data sources that include phenotype and genotype information are linked together in order to explore some hidden relations. The biomedical domain is well-researched, as a result of the existence of extensive biomedical vocabularies, standards, and resources that are available (Aronson, 2006). The Unified Medical Language System (UMLS) (Bodenreider, 2004) integrates and distributes key terminology, classification and coding standards, to promote the creation of more effective and inter-operable biomedical information systems and services (Schuyler et al., 1993). It also consists of tools for normalizing English strings, generating lexical variants, and creating indexes. This means that having the text description of a biomedical concept, we can find its matching from the UMLS vocabulary by using the lexical tools.
However, the food domain is still low-resourced regarding the availability of resources that can be used for developing artificial intelligence-based models. The food supply has evolved in recent years, alongside the increasing demand for nutritional and other food-related components. From recent studies involving nutrition huge amounts of data have been collected. In order for this data to be reusable and interoperable and equipped for data analysis, it needs to be harmonized and integrated. Data harmonization is the process of bringing together data of varying different formats, naming conventions, columns, and transforming it into one cohesive data set. One way of performing data harmonization is to match concepts to an existing and widely used domain-specific ontology. In the food and nutrition domain, this translates to matching food concepts to the few food ontologies that exist such as FoodOn (Griffiths et al., 2016), OntoFood and SNOMED CT (Donnelly, 2006). However, a recently published study (Popovski et al., 2019) showed that all of them were developed for some specific problems and their coverage is limited. Regarding other approaches for data harmonization and normalization in the food and nutrition domain, there is a semi-automatic system for classifying and describing foods according to FoodEx2 (EFSA, 2015), known as StandFood (Eftimov et al., 2017), which can be used for data normalization of food concepts. The limitation of StandFood is that currently, it works only with English foods’ names.

Food data can have various sources and formats: food composition data, food consumption data, recipe data, etc. The most commonly used type is food product data. However, this type of data is often misinterpreted as a result of the vast and very competitive marketing system nowadays. Different producers and retailers manipulate product names to achieve better marketing. The misinterpretation can also occur due to the vast variety of diet styles that have emerged recently.

In this paper, we focus on linking food-related concepts provided in a non-English (i.e. Slovenian) language which are extracted from two online grocery stores. By linking food concepts from multiple data sources, which often provide complementary information about food products, we can complete or at least enrich the available information. This can also be helpful in the process of missing value imputation in food composition databases (FCDBs), especially for branded food products.

It is important to note that the Slovenian language is a low-resourced language from the perspective of availability of natural language processing tools such as part-of-speech tagging (POS) (Voutilainen, 2003), chunking, lemmatization, which has represented an additional challenge while working with textual data.

To link food products using their text description, we use lexical and semantic similarity as heuristics. The lexical similarity focuses on the syntactic and morphological similarity of the compared text, while the semantic similarity focuses on their context similarity. In Section 2, we provide a critical overview of the related work. Next, in Section 3, our proposed methodology is explained in detail, followed by an explanation of the data in Section 4. Additionally, the experimental results and discussion are given in Section 5, where we also provide some directions for future work.

2 RELATED WORK

One of the challenges while working on text similarity is that the same concept can be mentioned using phrases with a variety of structures, which is a consequence of how people express themselves. In order to combine the information for the same concept that is represented in different ways, we should apply text normalization methods. Text normalization methods are based on text similarity measures.

Text similarity measures operate on string sequences and give us a metric of similarity (or dissimilarity) between two text strings. Text similarity determines how distant two texts are both in surface (i.e. lexical similarity) and meaning (i.e. semantic similarity).

Normalization methods based on text similarity measures are well presented in (Aronson, 2001; Savova et al., 2010). Several normalization methods that are based on ranking technique are available, with the goal to rank the candidate matches and then to find the most relevant match (Collier et al., 2015). Normalization methods can also utilize machine learning (ML) algorithms to improve results, which was shown in the gene normalization task as part of BioCreative II (Morgan et al., 2008) and BioCreative III (Lu et al., 2011). Regarding the food and nutrition domain, methods for normalization of short text segments (e.g., names or descriptions of nutrients, food composition data, food consumption data) have recently been proposed (Eftimov and Seljak, 2015; Eftimov et al., 2017; Ispirova et al., 2017; Eftimov et al., 2018) by using two approaches: two approaches: (i) standard text similarity measures; and (ii) a modified version of Part of Speech (POS) tagging probability-weighted method, first proposed in (Eftimov and Seljak, 2015).
2.1 Lexical Similarity

Lexical similarity can be calculated either on the character or word level. Most of the lexical similarity measures do not take into account the actual meaning behind words or the entire phrases in context, but focus on how many characters or words overlap.

Let $D_1$ and $D_2$ be two pieces of text. Some of the standard lexical similarity measures are (Metzler et al., 2007):

- The **Levenshtein distance** counts the number of deletions, insertions and substitutions necessary to turn $D_1$ into $D_2$.
- The **Optimal String Alignment distance** is like the Levenshtein distance but also allows transposition of adjacent characters. Each substring may be edited only once.
- The full **Damerau-Levenshtein distance** is like the optimal string alignment distance except that it allows for multiple edits on substrings.
- The **longest common substring** is defined as the longest string that can be obtained by pairing characters from $D_1$ and $D_2$ while keeping the order of characters intact.
- A **q-gram** is a subsequence of $q$ consecutive characters of a string. If $x$ (y) is the vector of counts of q-gram occurrences in $D_1$ ($D_2$), the q-gram distance is given by the sum over the absolute differences $|x_i - y_i|$.
- The **cosine distance** is computed as $1 - \frac{\textbf{x} \cdot \textbf{y}}{||\textbf{x}|| ||\textbf{y}||}$, where $x$ and $y$ were defined above.
- Let $X$ be the set of unique q-grams in $D_1$ and $Y$ the set of unique q-grams in $D_2$. The **Jaccard distance** is defined as $1 - \frac{|X \cap Y|}{|X \cup Y|}$.
- The **Jaro distance** is defined as $1 - \frac{1}{3} (w_1 \frac{|D_1|}{m} + w_2 \frac{|D_2|}{m} + w_3 \frac{|D_1 - D_2|}{m})$, where $|D_i|$ indicates the number of characters in $D_i$, $m$ is the number of character matches and $t$ the number of transpositions of matching characters. The $w_i$ are weights associated with the characters in $D_1$, characters in $D_2$ and with transpositions.
- The **Jaro-Winkler distance** is a correction of the Jaro distance. It uses a prefix scale $p$ which gives more favourable ratings to strings that match from the beginning for a set prefix length $l$.
- The **skip-grams** are generalization of n-grams in which the components (typically words) need not be consecutive in the text, but may leave gaps that are skipped over.

2.2 Semantic Similarity

Semantic similarity is a metric that defines the distance between two pieces of text based on their meaning or semantic content. Calculating semantic similarity is related to representational learning (i.e. learning embeddings), which has become an important research task for learning representation of symbolic data. The idea of representational learning is to represent each piece of text (e.g., word, sentence, paragraph, depending on the problem) as a vector of continuous numbers. In the case of learning word embeddings, the learned vector captures the context of a word in a piece of text, as well as semantic and syntactic similarity, relation with other words, etc. To find the similarity between two words, we should calculate the similarity between their vectors. To do this, we can find the angle between their vectors. The cosine distance between two words represented by their vectors $x$ and $y$ can be calculated using the following equation:

$$\cos(x, y) = \frac{\textbf{x} \cdot \textbf{y}}{||\textbf{x}|| ||\textbf{y}||}.$$ (1)

2.2.1 Word2vec Embeddings

In order to include the semantic information in these representations, Mikolov et al. (Mikolov et al., 2013a; Mikolov et al., 2013b) presented the word2vec model, which learns high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships. These representations are also known as embeddings. Using this model, each token (i.e. words) is represented as a vector of continuous numbers.

2.2.2 GloVe Embeddings

GloVe (Pennington et al., 2014) is an unsupervised learning algorithm for obtaining vector representations for words, which is based on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations are linear substructures of the word vector space.

3 METHODOLOGY

To match the information about the same food products from different data sources, we first preprocess the data. Next, we match the food products by applying lexical similarity measures, followed by matching them with regard to semantic similarity. Finally, we compare the mapping results by evaluating them on a
set of pairs that represent the ground truth, which are pairs matched by their EAN barcodes.

### 3.1 Lexical Similarity

Let $D_1$ and $D_2$ be two pieces of text. First POS tagging, also called grammatical tagging, is applied to each of them to identify the part-of-speech tags such as nouns (NN, NNS, NNP, NNPS), verbs (VB, VBD, VBG, VBN, VBP, VBZ), adjectives (JJ, JJR, JJS), cardinal numbers (CD), etc (Márquez and Rodriguez, 1998). Let us define

$$Y_i = \{\text{tokens from } D_i \text{ that belong to word class } i\},$$

(2)

where $i = 1, 2$. The word classes are: nouns, adjectives, verbs, adverbs, prepositions, determiners, pronouns, conjunctions, modal verbs, particles, and numerals. For example, $Y_1$ can be a set of all tokens from $D_1$ that are tagged as nouns. In such case, the set consists of all tokens that are tagged as NN, NNS, NNP, and NNPS.

The next step is to define which of the extracted word classes (morphological POS tags) are significant to describe the domain to which the text belongs. The set of nouns is crucial because nouns carry most of the information in the text, while all other word classes (adjectives, verbs, numbers, etc.) only give an additional explanation. After extracting the set of nouns and the sets of other word classes that are significant for the domain, lemmatization (Korenius et al., 2004) is applied to each of them. To find string similarity between both pieces of text, a probability event is defined as a product of independent events

$$X = N \prod_{j=1}^{k} Z_j,$$

(3)

where $N$ is the similarity between the sets of nouns found in both pieces of text, $k$ is the number of additional word classes that are selected and are significant for the domain, and $Z_j$ is the similarity between the sets of word class, $i$, found in both text. The additional word classes can be adjectives, verbs, etc.

Because these events are independent, the probability of the event $X$ can be calculated as

$$P(X) = P(N) \prod_{j=1}^{k} P(Z_j).$$

(4)

To calculate it, the probabilities of the independent events need to be defined. Because the problem looks for the similarity between two sets, it is logical to use the Jaccard index, $J$, which is used in statistics for comparing similarity and diversity of sample sets (Kosub, 2019). For the similarity between the nouns, the Jaccard index is used, while for the similarity between the additional word classes the Jaccard index in combination with Laplace probability estimate (Cestnik et al., 1990) is used. This is because, in some short segments of text, the additional information provided by other word classes can be missed, so there will be no zero probabilities. The probabilities are calculated as

$$P(N) = \frac{|N_1 \cap N_2|}{|N_1 \cup N_2|},$$

$$P(Z_i) = \frac{|Z_{i1} \cap Z_{i2}| + 1}{|Z_{i1} \cup Z_{i2}| + 2}. \quad (5)$$

By substituting Equations 5 into Equation 4, we obtain a weight for the matching pair.

If we focus on the food domain, or specifically on the food matching problem, let $D_1$ and $D_2$ be the (Slovenian) names of two selected food products. As we said before, the nouns carry most of the information, while the additional word classes that describe the food domain are adjectives, which explain the food item in more detail (e.g., frozen, fresh), and the verbs, which are generally related with the method of preparation (e.g., cooked, drained). Let us define

$$N_i = \{\text{nouns extracted from } D_i\},$$

$$A_i = \{\text{adjectives extracted from } D_i\},$$

$$V_i = \{\text{verbs extracted from } D_i\} \quad (6)$$

To find the similarity between the names of food products, an event is defined as a product of two other events

$$X = N \cdot (A + V),$$

(8)

where $N$ is the similarity between the noun sets found in $N_1$ and $N_2$, and $A + V$ is the similarity between the two sets of adjectives and verbs handled together as $A_1 + V_1$ and $A_2 + V_2$. The adjectives and verbs are handled together to avoid different forms with the same meaning. Additionally, lemmatization is applied for each extracted noun, verb and adjective, and the similarity event uses their lemmas.

Because these two events are independent, the probability of the event $X$ can be calculated as

$$P(X) = P(N) \cdot P(A + V).$$

(9)

The probabilities are calculated as

$$P(N) = \frac{|N_1 \cap N_2|}{|N_1 \cup N_2|},$$

$$P(A + V) = \frac{|(A_1 \cup V_1) \cap (A_2 \cup V_2)| + 1}{|(A_1 \cup V_1) \cup (A_2 \cup V_2)| + 2}. \quad (10)$$

By substituting Equations 10 into Equation 9, we obtain a weight for each matching pair.
3.2 Semantic Similarity

For mapping the food products from both datasets considering semantic similarity, we decided to apply two different word embedding techniques – word2vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014). For the model training we used the lemmas of the words contained in the names of the food products. The reason for learning vector representations for the lemmas and not the whole words is the fact that one word, grammatically, can have different cases in Slovene. Let us have \( f_p \) is the name of the food product, which is consisted of \( n \) words:

\[
 f_p = \{ word_1, word_2, \ldots, word_n \} \tag{11}
\]

After obtaining the lemmas of each word:

\[
 f_p = \{ lemma_1, lemma_2, \ldots, lemma_n \} \tag{12}
\]

We then apply the two algorithms and obtain vector representations for each lemma (i.e. word) in the food product name:

\[
 E[lemma_a] = [x_{a1}, x_{a2}, \ldots, x_{ad}] \tag{13}
\]

Where \( a \in \{1, \ldots, n\} \), and \( d \) is the dimension of the generated word vectors, manually defined for the both of the algorithms. After obtaining the vector representations, the next step is to apply a heuristic for merging the vectors for all the lemmas of a name, in order to obtain the vector representation for the whole food product name. We chose to work with two heuristics:

1. Average – Calculating the vector representation for the food product name as an average from the vector representations of the lemmas of the words from which it consists of:

\[
 E_{\text{average}}[fp] = \left[ \frac{x_{a1} + \ldots + x_{an}}{n}, \ldots, \frac{x_{ad} + \ldots + x_{nd}}{n} \right] \tag{14}
\]

2. Sum – Calculating the vector representation of the food product name as a sum from the vector representations of the lemmas of the words from which it consists of:

\[
 E_{\text{sum}}[fp] = \left[ x_{a1} + \ldots + x_{a1}, \ldots, x_{ad} + \ldots + x_{nd} \right] \tag{15}
\]

Finally, to perform the matching, we calculate the cosine similarity between the vector representations of the food product.

3.2.1 Word2vec Embeddings

The only numeric parameters that varied between the different word2vec models were the dimension size and the sliding window size. Values for the sliding window were chosen to be \([2, 3, 5]\), while the dimensions were \([100, 200]\). Additionally, the feature extraction algorithms included Bag of Words and Skip-gram. By combining these parameter values, a total of 12 word2vec models were trained.

3.2.2 GloVe Embeddings

Analogous to the word2vec parameter choice, the same values were used for the numeric parameters of GloVe, i.e. \([2, 3, 5]\) for the sliding window and \([100, 200]\) for the number of dimensions. Thus, a total of six models were trained.

In both cases, the sliding windows were chosen according to the average number of words per food product, which rounded equals to nine.

4 DATA

In this section we explain the data collection process, after which we elaborate on the data pre-processing step.

4.1 Data Collection

The data about food products used in this study were scrapped from the web sites of two food retailers (for convenience purposes let us name them: Retailer1 and Retailer2). Each website contains some, but not complete, information about each food product, such as the food product name in Slovenian, the EAN barcode, the food label, the lists of ingredients and allergens, and the name of the producer. For the food products for which we have their food product names and EAN codes, we constructed datasets containing these two pieces of information about each product (the format of the datasets is shown in Table 1). It needs to be pointed out that the food names were similar, but not the same (e.g. bread is named by one retailer as “bel kruh”, i.e. “white bread” in English, and by another retailer as “pšenični kruh, bel”, i.e. “wheat bread, white”).

Where \( fp \) is the food product name, \( bc \) is the corresponding EAN code, and \( n \) is the number of food products in each dataset – for \( \text{Retailer1} \), \( n = 1,836 \) and for \( \text{Retailer2} \), \( n = 6,587 \).
4.2 Data Pre-processing

Having the datasets in the format presented in Table 1, before applying the algorithms for obtaining semantic similarity or calculating lexical similarity with Equation 10, the data needed to be pre-processed. The first step was to perform POS tagging on the food product names. Since we are working with words in Slovenian, the POS tagger that is used is for Slovenian (Grcar et al., 2012). The Slovenian tagger outputs the tokens in three types of data: word form, lemma, and morph-syntactic description or tag. We use the lower case lemmas for each word. The data consists of words spanning across multiple morphological types. However, only the lemmas nouns, adjectives, and verbs convey semantic information. Therefore, these are the only three types that are considered while calculating lexical similarity and training the word embedding models.

5 EVALUATION

In order to produce a dataset consisting of ground truth values, we matched the food products by using their corresponding EAN codes. With this, we obtained 438 food products that are available in both retailers’ catalogues.

Since Retailer 1 has significantly fewer food products to offer, we find the five most similar food products from Retailer 2 and check whether one of them corresponds to the food product matched by the EAN code. If so, we count this as a positive example, otherwise as a negative one.

For computing the similarity between the food products, we fixed the dimensionality to 200, used a sliding window of 5 for both the word2vec and GloVe models. Additionally, word2vec was trained using CBOW. Lastly, the lexical similarity measure was computed according to formulas 9 and 10.

The hyper-parameter choice was made after evaluating the models described in Section 3.2. The model with the best empirical results proved to be the ones with a dimensionality of 200 and a sliding window of 5. Therefore, we use this model in our final evaluation.

To gain some insight regarding the embedding training process, it is useful to look at the values of

The loss function for each training iteration (epoch). On Figure 1 these values are plotted. It is evident that the loss improvement plateaus after a certain point, so it is computationally beneficial to stop the training process after this plateau is reached. This also prevents over-fitting the training data, which is important if new data is added for future evaluation. In this case, the plateau is somewhere around iteration 800, which is where it is favorable to stop the training process.

5.1 Results and Discussion

In Table 2, we present the results of the evaluation on the dataset of 438 food products having similar, but not the same, food names and the same EAN barcodes. It is interesting to note that both summing and averaging the vector embeddings provided identical predictive results. Additionally, in Table 3, the accuracy of each model is presented.

Table 2: Evaluation results for each model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Word2Vec</th>
<th>GloVe</th>
<th>Lexical sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positives</td>
<td>271</td>
<td>238</td>
<td>329</td>
</tr>
<tr>
<td>Negatives</td>
<td>167</td>
<td>200</td>
<td>109</td>
</tr>
</tbody>
</table>

Table 3: Accuracy for each model.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word2Vec</td>
<td>0.61872</td>
</tr>
<tr>
<td>GloVe</td>
<td>0.54338</td>
</tr>
<tr>
<td>Lexical</td>
<td>0.75114</td>
</tr>
</tbody>
</table>
Looking at Table 2 it follows that out of a total of 438 food products, 271 were in the top five predictions when using the word2vec model; 238 were in the top five predictions when using the GloVe model and 329 were in the top five predictions when using the lexical model.

Further insight into the matching evaluation can be obtained by counting how many food products were not found (Negatives) for all models. Specifically, we count how many food products were not positively matched by any model at all. Taking this into consideration, 355 out of a total of 438 products in the evaluation set were positively matched by at least one of the models. These results are additionally presented in Table 4.

<table>
<thead>
<tr>
<th>Total positively matched</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>355</td>
<td>0.81050</td>
</tr>
</tbody>
</table>

For example, in Table 5 the top five matches for a food product (in this case “jogurt mu borovnica 1,3mm 1l”, i.e. blueberry fruit yogurt) from each model is presented. In this example, one of each five matches is a positive match in the ground truth evaluation dataset.

Additionally, even if the matches from the semantic models do not include the ground truth product, they still convey significant semantic information about the food products. In Table 6 we provide one such example, where it is evident that all five matches are related to the food product of interest. In this example, all food products are related to “sir”, i.e. cheese. Therefore, the semantic models are not limited by the lexical information of the food product name and can be used to match food concepts in cases where there is low lexical similarity, but the semantic similarity is high.

One thing to notice is that using lexical similarity as a heuristic will always yield better results when considering the task of matching branded food products, while semantic similarity as a heuristic can provide more insight when considering other tasks, such as matching food data for imputing missing nutrient values from food composition databases.

One weakness of the semantic models is that they are using embeddings on a word level. For our future work, we are planning to explore more advanced textual representational models such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2019), and ALBERT (Lan et al., 2019). There are pre-trained models for English text for all of these embedding methods. However, in order for these methods to be used with Slovenian text, we should acquire more data and train the corresponding models. Additionally, the same methodology described in this paper can be generalized and applied to any language, provided sufficient pre-trained models, or data to train the required models on, exist.

6 CONCLUSIONS

The problem of food data integration becomes especially important with one of the 2030 development goals of the United Nations, which states “End hunger, achieve food security and improved nutrition and promote sustainable agriculture” (Lartey, 2015). With the huge amount of food and nutrition-related data that is collected in the last 10 years, there is a need for data normalization techniques that will link these data sets.

In this paper, we propose two heuristics that can be used for matching food products represented by their non-English descriptions (i.e. Slovenian). To give a matching score of a pair of food products, the first one is based on lexical similarity, and the matching score is a probability event defined as a product of similarity between the set of nouns that appear in their names and the joint set of adjectives and verbs. The second one is based on semantic similarity and uses word embeddings. For it, first vector representations (i.e. embeddings) for the lemmas of nouns, adjectives, and verbs, which appear in food products names, are learned. After that, the vector representation of a food product name can be calculated as an average or sum from the vector representations of the lemmas of the words from which it consists of. The matching score of a pair of food products is the cosine similarity between the vector representations of their names.

We evaluated the proposed heuristics by mapping food products from two online grocery stores. We compared the results for the proposed heuristics using a data set of 438 food product pairs, which present the ground truth. They were obtained by matching the food products from every pair based on their EAN codes. By applying the proposed heuristics, for the first food product from every pair, we returned the 5 most similar food products, and we checked whether one of them corresponds to the second food product from the pair. Experimental results showed that the best semantic models achieve an accuracy of 62%, while the lexical model outperforms this with an accuracy of 75%. Additionally, if all the models are considered together, an accuracy of 81% is obtained.

For our future work, we are planning to explore more advanced textual representational methods (i.e.
embeddings methods), and also use the information from graph-based embeddings to improve the matching process.

ACKNOWLEDGEMENTS

This work was supported by the project from the Slovenian Research Agency (research core funding No. P2-0098), and the European Union’s Horizon 2020 research and innovation programme (grant agreements No. 863059 and No. 769661).

Information and the views set out in this publication are those of the authors and do not necessarily reflect the official opinion of the European Union. Neither the European Union institutions and bodies nor any person acting on their behalf may be held responsible for the use that may be made of the information contained herein.
REFERENCES


