UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Olha Kaminska

Entity linking via topic models in Apache Spark

Master's Thesis (30 ECTS)

Supervisor: Pelle Jakovits, PhD
Supervisor: Peep Küngas, PhD

Tartu 2019

Acknowledgement

I would like to thank my supervisors, Pelle Jakovits and Peep Küngas, for providing the topic, helping with the pipeline, and supporting me throughout the process of writing this work.

I am grateful to Tarkvara Tehnoloogia Arenduskeskus (STACC) for supporting my thesis.

I want to say thanks to Alyssa, who really helped me with my English.

Also, I would like to express special thanks to Viacheslav and Elena for offering understanding and moral support during difficult moments.

Entity linking via topic models in Apache Spark

Abstract:

Entity linking is a field of natural language processing that aims to define the real meaning of a word in a particular text. The same term can have different meanings in different contexts, which demonstrates the importance of the field. Entity linking is actively applied to real-world business problems. One widely known problem is defining companies with similar products to investigate competitors on the market. In this task, products represent entities, and the target of the entity linking is to connect the same or similar products among an assortment of different companies.

In the current work, similar products from different Estonian companies are linked based on their textual descriptions. In the obtained results, every company is linked with at least one other company through similar products.

To define similar products, the textual descriptions are divided into clusters using four different topic modeling techniques. Based on the obtained clusters, linked graphs are built in the Apache Spark environment and manual tests and comparisons using statistical measures are performed. The graphs based on latent Dirichlet allocation topic modeling approaches show the best results.

The performance of the methods illustrates that topic modeling techniques can be used for entity linking and can provide practical results.

Keywords:
Entity Linking, Topic Modeling, Apache Spark, Natural Language Processing, Relationship Graph

CERCS: P170. Computer science, numerical analysis, systems, control.

Üksuste sidumine teemade modelleerimise abil Apache Sparkis

Lühikokkuvõte:

Loomuliku keele töötluse üks harusid tegeleb üksuste sidumisega, eesmärgiga võimaldada selgitada sõna tõelist tähendust kindla teksti kontekstis. Erinevates kontekstides võib samal terminil olla mitu tähendust, millest tuleneb ka valdkonna olulisus. Üksuste sidumist rakendatakse aktiivselt äriprobleemide lahendamisel. Üks levinumaid probleeme on sarnaseid tooteid arendavate ettevõtete tuvastamine, mis võimaldaks uurida konkurentsile rajatud turgu. Antud töös käsitletud tooted tähistavad üksuseid ning eesmärgiks on ühendada erinevate ettevõttete valikust pärinevad tooted, mis on omavahel kas samad või sarnased.

Siinses uurimuses eri ettevõtetest pärinevate sarnaste toodete sidumine toimus nende tekstiliste kirjelduste põhjal. Saadud tulemustes iga ettevõte seoti vähemalt ühe teise ettevõttega sarnaste toodete põhjal.

Võimaldamaks kirjeldada sarnaseid tooteid, kasutati nelja erinevat teemade modelleerimise võtet, et klasterdada vastavate toodete tekstilised kirjeldused. Saadud klastrite põhjal ehitati Apache Sparki keskkonnas seotud graafid ning viidi läbi manuaalne testimine ja statistiliste mõõdikute võrdlemine. Juhendamata masinõppe mudelil (LDA) põhinev graaf näitas parimaid tulemusi.

Saavutatud täpsus näitab, et teemade modelleerimise võtteid saab kasutada üksuste sidumiseks ning need võimaldavad jõuda praktiliste tulemusteni.

Võtmesõnad:

Üksuste sidumine, teemade modelleerimine, Apache Spark, loomuliku keele töötlus, Suhete Graaf

CERCS: P170. Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

List of Figures

List of Tables

1 Introduction

Natural language processing (NLP) is a field of computer science related to the interaction between computer and natural human languages. In particular, this area investigates how to program computers to process and analyze large amounts of textual data in natural language.

The common problems that usually need to be solved with NLP techniques can be separated into three main fields: speech recognition, natural language understanding, and natural language generation. In the current paper, approaches for natural language understanding, such as topic modeling, are employed.

Data, including textual data, can be saved and used as linked data. Linked data is a method of publishing structured data such that it can be interconnected and more useful through semantic queries, but instead of using it only to serve web pages for human readers, it is also used to exchange information in a way can be read automatically on computers. If the linked data is open data, then it is described as linked open data. According to Open Definition[1], open data is information that can be openly used and copied by anybody.

Natural language processing usually works with text as a set of entities. In NLP, an entity is a word or set of words that represents a particular concept or object.

Depending on the context, the same entity in different texts could have various meanings. The main goal of entity linking is to define the correct meaning of the entities mentioned in a particular text. In the current work, the entities represent products of diverse companies, and the purpose of the entity linking here is to detect the same product or products that are highly similar to each other among an assortment of different companies.

This thesis was created based on the needs of the company Register OÜ[2]. This company specializes in the development of information technology (IT) solutions for credit management and sales, and its goal is to optimize business processes in order to maximize sales. The solutions it creates help companies draw important insights and make with business decisions.

To provide the most up-to-date information, Register OÜ collects and stores terabytes of open Estonian web data every week. Data-driven models and applications identify payment behavior patterns and prevent credit loss. The company's approaches are crawling web data, extracting linked data that contains entities, and generating knowledge graphs to extract information about the relationships of Estonian companies.

In 2018, a master's thesis entitled "Large Scale Feature Extraction from Linked Web Data" by Madis-Karli Koppel [Kop17] was composed for Register OÜ. In the thesis, a pipeline with several steps was created, where the input was linked data extracted from

[1] http://opendefinition.org/

[2] https://www.inforegister.ee/

web crawling, and the output was features to be used in a machine-learning model for credit-scoring. The steps were as follows:

- Open linked data were extracted via web crawling.
- Ambiguity was eliminated using named entity recognition (NER)[3].
- A linked graph was created from the data, where the links were based on stock-keeping units (SKUs)[4].
- Features were extracted from the graph to further be used in the machine-learning model for credit-scoring.

For the experiment, Register OÜ provided data for processing and outlined certain result requirements.

The outcome of the experiment was the successful implementation of the whole pipeline, but there was an issue with the entity linking part. Only using SKUs as characteristics was not enough to build a sufficient number of links between entities and to detect all the common products from different companies. This shortcoming was discovered after analyses of the obtained graphs. The number of products that connected two or more companies was not sufficient to extract features for the machine-learning model for credit-scoring. Koppel explored products with the unique SKUs, and discovered that only 13,699 such products were connected to some companies. This number corresponds to 1.9% of all products from the source data. Especially, taking into account that from those 13,699 products amount of products linked to the more than one company is less than 300 products. Thus, the proposed pipeline should be improved. The updated pipeline should generate more links and connect more companies.

Problems with entity linking were highlighted as one of the most important issues with the existing pipeline, and thus, it is necessary to investigate how the linking of Estonian companies and products could be improved. The goal of the current work is to improve entity linking in graph-building to extract more features for a credit-scoring model.

One method that could probably improve the quality of entity-linking results is topic modeling, an approach to building a model that analyzes a collection of text documents and determines which topics each document contains. Topic modeling can be performed in different ways, and in this work, several of the most popular approaches are presented, described in chapter 3 Methodology.

Topic modeling was chosen as a possible method to improve entity linking because it separates products into different groups based on their descriptions. So far as products in

[3]Named entity recognition is a task of text analysis that attempts to find and classify proper names mentioned in the unstructured data

[4]Stock Keeping Unit - separate item type for sale, presented in current data as set of numbers

the same group are related to the one topic, it is possible to further build links between products in the same group. Important is that it uses textual information about the product (such as the name and description), unlike in the previous approach, as is described in subchapter 2.2 Related works.

Thus, the current work should answer the question of whether it is possible to improve entity linking using topic-modeling methods. Topic modeling is presented as a possible method to sufficiently improve the quality of linked graphs. Improvement is important because it enables the construction of more links in the existing graph and provides more features for machine-learning models. The output of the original linking was not sufficient to produce a usable result. In other words, the end goal is to obtain more features to cover more companies and their relationships with each other.

The environment used in the previous approach to build the graph and make the links was Apache Spark[5], a distributed, open-source, universal computing framework that can perform work quickly with a large number of data. Spark is needed for the current project as well because Register OÜ collects terabytes of data (and plans to continue doing so in the future), and distributed computing platforms such as Apache Spark are required to handle such large volumes of data.

The next chapter is entitled State of the Art and provides an overview of the main theoretical concepts of this work and discusses related works that have investigated similar problems.

The Methodology chapter contains an overview of the project source data and their structure, size, content, and meaning. It also presents data statistics and the selection of the textual data for the next steps. Additionally, this chapter presents the several topic-modeling approaches used during the thesis experiments.

The Validation chapter contains a description of all the obtained results from the Methodology chapter and a comparison of the graph structures obtained after different instances of topic modeling, both with each other and with the previous graph structure generated in Koppel thesis [Kop17]. This chapter also contains a discussion on the quality of the results, whether they are significant, and whether topic modeling is a suitable approach for improving entity linking.

In the Conclusion chapter is a summary about the work performed and its results, the findings regarding how useful the proposed improvement is, and opportunities for further work.

[5]https://spark.apache.org/

2 State of the Art

2.1 Background

This chapter provides an overview of the main terms and concepts used during this thesis work.

2.1.1 Entity linking

Linked data is a method of publishing structured data with different vocabularies that can be connected and expounded upon by a computer without human intervention. It enables data from different sources, such as web pages, to be connected and queried. If the linked data is open data, then it is described as linked open data. One of the most popular vocabularies is schema.org, which was used by Register OÜ to obtain source data for the current work.

Presenting data in a linked way can make it more useful in semantic queries. With this method, statements are encoded in triples or N-quads to be spread across different websites. A triple is a set of three components — "subject", "predicate", and "object" — that encodes a statement about the semantic data. An N-Triple is a format for storing and transmitting triples data. It serves as a syntax to represent information in the Resource Description Framework (RDF) format, a group of specifications that was created as a metadata data model, used as a common approach for data representation and structuring in the web.

In Fig. 1, an example of triples in a graph format is illustrated, where the subject "Tartu" has two predicates, "Country" and "Population", with corresponding meanings (objects) for both predicates: "Estonia" and "93,715".

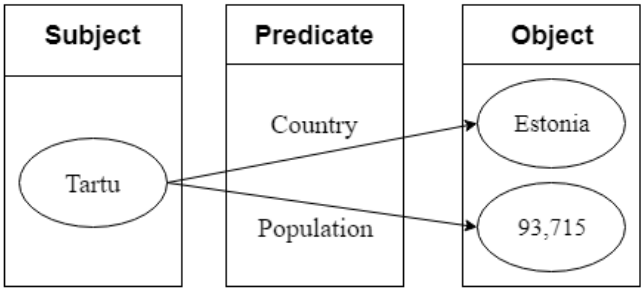


Figure 1. Example of triples in graph form.

An N-quad is a type of N-Triple with an optional "context" component in the fourth position. In the current work, the source data were saved in the N-quad format, where the

subject is a web link to the product, the predicate is the id node of the product, the object is a characteristic of the product, and the context is a value of this characteristic. The initial data are described in more detail in chapter 3 Methodology. In Fig. 2, an example of an N-quad in a graph format is illustrated.

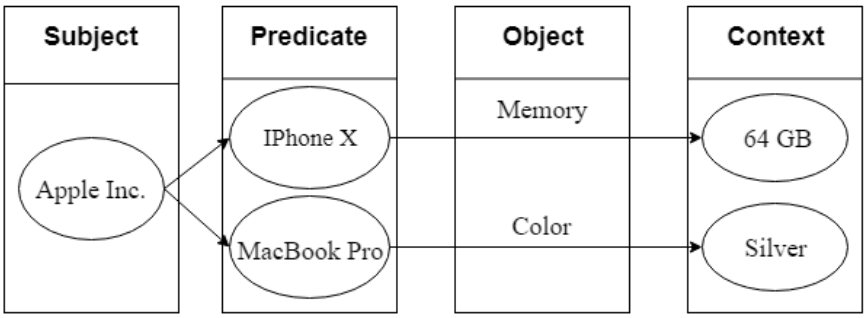


Figure 2. Example of an N-quad into graph form.

In practice, linked data usually is used on web pages to connect concepts in the presented information. A prime example is Wikipedia[6], where on each web page, all the main terms pertaining to the topic are presented with hyperlinks to the page about that term.

From each linked-data term, entity linking can be formed. The main goal of entity linking is to define the correct meaning of the entities mentioned in the particular text. In the current work, the entities are products of diverse companies. Hence, the purpose of the entity linking in this context is to detect the same product or highly similar products among an assortment of different companies. Based on this knowledge, it will be possible to discover links between companies selling common products and to use these links to build a credit-scoring model.

2.1.2 Topic modeling

The method used in this work, which aims to improve the quality of entity-linking results, is topic modeling, a type of statistical model developed to discover abstract topics in a set of documents. In other words, it is a way to build a model that determines which topics are featured in every document in a collection of text documents.

Documents typically contain several topics in different proportions. All topics discovered by a topic-modeling tool are represented by clusters of words close in meaning to each other (further in the work words "cluster" and "topic" will be used as synonyms). This technique was developed to solve text-mining tasks, but currently, topic modeling

[6]https://www.wikipedia.org/

is used in a wide variety of challenges to discover informative structures in genetic information, images, networks, bioinformatics, and more.

Nowadays, the most popular topic-modeling method is latent Dirichlet allocation (LDA), but this approach has several predecessors that are quite popular as well. The rest of the subchapter contains a short description of the main methods inspired by J. Xu article [Xu18].

One of the foundational methods in topic modeling is latent semantic analysis (LSA). The main idea of LSA is to take a matrix of texts and terms and decompose into split text-topic and topic-term matrices. In the initial text-term matrix, every row presents a single text from the corpus, and each column presents a single term among all the terms among entire corpus. The values in each cell represent the term frequency-inverse document frequency score for the text at the current row and the word in the corresponding column. The term frequency-inverse document frequency (TF-IDF) score is described in chapter 2.1.3 Text vectorization.

After obtaining the topic–term matrix, it is clear that the matrix is sparse and large with significant noise. To discover hidden topics and extract the most important term, a dimensionality reduction should be performed using a truncated SVD approach.

Singular value decomposition (SVD) is a method that factorizes a matrix into the product of the three different matrices, as illustrated with in the following formula (1):

$$A = U * S * V, \tag{1}$$

where S presents a diagonal matrix of the singular values of A. The truncated approach of SVD performs a dimensionality reduction using only a particular amount number of the largest singular values, in that way left this amount of the first columns of U and V matrices.

In LSA tasks, matrix U represents the text–topic matrix and matrix V the term–topic matrix. In both matrices, the columns correspond to one of the topics. In matrix U, the rows correspond to the text vectors in terms of topic, and in matrix V, they correspond to the term vectors in terms of topic. With the obtained text and term vectors, measures such as cosine similarity can be applied to evaluate similarities between different texts or words.

The LSA method is fast, but it has some disadvantages. For example, the nature of the topics obtained is unknown, and the method requires a large dataset for high-quality results. For these reasons, some improvements to the original approach have been made.

Probabilistic LSA (pLSA) is an improvement of LSA that uses a probabilistic method instead of the SVD approach. Probabilistic latent semantic analysis appends a probability to the basic postulate of topic modeling, saying that each text from the corpus contains several hidden topics, where each topic is an assemblage of keywords.

The target of pLSA is to find a probabilistic model with the hidden topics that can be generated based on the data from the document–term matrix. In other words, it

is necessary to make a model $P(D, W)$, where for any document d and word w, and probability $P(d, w)$ corresponds to those elements' record in the text–term matrix.

The probability of a particular term occurring in a particular text is presented as the following formula (2):

$$P(D, W) = \sum_{T} P(T)P(D|T)P(W|T), \tag{2}$$

where $P(T)$ is the probability of topic T, $P(D|T)$ is the probability of document D in the set of documents that contain topic T, and $P(W|T)$ is the probability of word W in the set of words that belong to topic T.

In essence, the formula calculates the probability of a certain document containing a particular word based on the topic distribution. There also exists a parallel between this equation and formula (1) for the LSA model: $P(T)$ represents diagonal matrix S, which covers singular topic probabilities, $P(D|T)$ represents text–topic matrix U and $P(W|Z)$ represents term–topic matrix V.

pLSA is a more adaptable model than LSA, but it also has some issues. In general, the main problems are the lack of clarity in assigning probabilities to new texts in the corpus and issues of overfitting when increasing the size of parameters in the texts in the source data. For these reasons, pLSA is rarely used in its original form; it is usually replaced by latent Dirichlet allocation (LDA), an extension of pLSA.

LDA is a method that presents a Bayesian extension of pLSA. The latent Dirichlet allocation uses the Dirichlet priors for both distributions (text–topic and topic–term), allowing it to generalize better.

Put simply, Dirichlet distribution can be described as "the one distribution above the rest," and it answers the question of whether real probability distributions can be observed with this particular type of distribution.

The following example is presented to show how this method works in practice. Consider source data comprised of a text corpus with three mixed topics inside. After a distribution is conducted to build a model, a large weight is set on one topic and tiny weights on the others, leaving them almost neglected. For all three topics, the specific probability distributions are described as 90% for the one topic and 5% each for the other two. After a random probability distribution is drawn over the proposed Dirichlet distribution, the most likely result is a distribution that looks like a mix of the three distributions, each of which presents the same one topic as the major topic. It is highly unlikely to sample a distribution where all three topics are of equal size. In general, the Dirichlet distribution provides a method of sampling probability distributions of a particular type.

LDA is usually used instead of pLSA because it overcomes the main disadvantage of pLSA: it can extrapolate new text easily. In pLSA, if a document is not presented in the source corpus, the method cannot provide a data point for it. However, in the

LDA method, the source data are used as training data for training text–topic Dirichlet distributions. For every new document, it is easy to evaluate the data point from pre-trained Dirichlet distributions and apply them to the new document.

In the current work, among the three listed methods of topic modeling, LDA was chosen as the most efficient and most popular one. Furthermore, to compare results in the form of product clusters, a couple of approaches were developed based on text vectorization and clustering. These methods are described in more detail in chapter 3.2 Methods.

2.1.3 Text vectorization

Text vectorization is a method of presenting words from natural language in a meaningful set of numbers that can be processed by a computer. Since the current thesis works with product descriptions, several approaches of text vectorization are employed.

The basic method of word-embedding is called bag of words (BOW). It represents text as a bag or set of words, ignoring grammar and word order but taking into consideration multiplicity. Usually, Bag of Words is used for feature generation based on the text corpus.

BOW is a simple approach and has one major disadvantage: it ignores word order and semantics during encoding.

The most popular way to encode words based on their importance is term frequency-inverse document frequency, a numerical statistic designed to show the importance of words in a document's corpus. Usually, it is used as a weighting coefficient in the exploration of information retrieval, text analytics, and more. The TF-IDF formula (3) has the following structure:

$$TF - IDF = TF_{i,j} * log(\frac{N}{df_i}), \tag{3}$$

where i is a term from the corpus, j is a document from the set, $TF_{i,j}$ is the number of occurrences of term i in document j, N is the total number of documents, and df_i is the number of documents from the set that contain word i.

As can be observed from the definition, a word has a large weight in the corpus when it occurs frequently across one particular text but rarely across the whole set of texts.

Another popular approach for word vectorization is word to vector. Word to vector (Word2Vec) represents words as vectors in a multidimensional space with one important characteristic: words with common meanings are represented with vectors in close proximity to one another. Word2Vec accepts text corpora as input and as output produces vector space, typically with several hundred dimensions. This approach was developed by researchers from Google and is described in T. Mikolov, K. Chen, G. Corrado and J. Dean article [MCCD15]. Furthermore, many different realizations of this algorithm

have emerged. The most popular one for Python is the Gensim approach[7].

If it is necessary to obtain vectors not from separate words but from whole texts, the document to vector (Doc2Vec) technique can be used. Its main idea is the same as that of Word2Vec, but document to vector processes not each word separately but rather sets of words, so the obtained vector represents an entire text.

2.1.4 Clustering methods

After obtaining vectors from the words, to find the closest ones, a clustering algorithm can be performed on the vector set.

A Density-based spatial clustering of applications with noise is a density-based clustering approach that links together the cluster elements located close to each other (in other words, the ones that have many neighbors). It defines as outliers or noise points isolated elements located somewhere alone.

To train the DBSCAN clustering model, two parameters should be specified:

- eps - the radius of the neighborhood for each element from the input data;
- $minPts$ - the number of neighbors that should be inside the circle created by the radius eps around any element to belong to the same cluster.

An example of these parameters is presented in Fig. 3, inspired by the illustration by J. Sander, X. Xu, M. Ester, and H.-P. Kriegel paper [EKS+96].

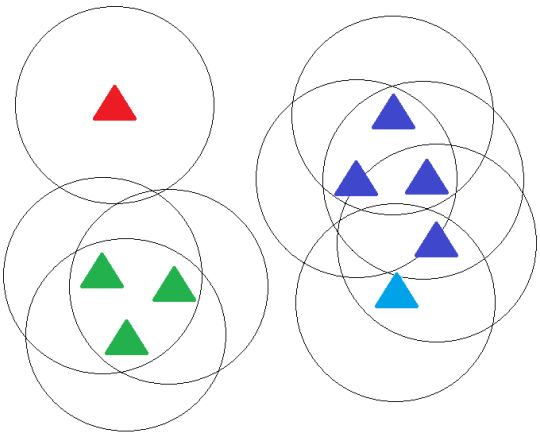


Figure 3. Example of DBSCAN results.

In Fig. 3, the result of the DBSCAN clustering method for a dataset of triangles is illustrated, with an unspecified eps parameter and a $minPts$ equal to 3. The algorithm

[7]https://radimrehurek.com/gensim/models/word2vec.html

in the diagram has discovered two clusters – a green one and a blue one – as well as one red noise point. All the points in the green cluster have three neighbors (including themselves) in their circles with the radius eps, so they belong to one cluster and form its core. Similarly, the dark blue points in the blue cluster have at least three neighbors in their circles with the radius eps and thus are core points. The light blue point also belongs to the blue cluster, but it has only one neighbor. It counts as part of this cluster because it can be reached from any other point of the cluster (points are reachable when there exists between them a path from other points, where the distance from one point to another is smaller than the eps). The red point is noise because it does not have any neighbors in its circle with the radius eps.

Another one of the most popular clustering algorithms in data mining is KMeans. This method is based on the division of elements into a particular number of clusters, where each element lies in the cluster with the closest average value, represented as a cluster prototype. The full process of clustering with KMeans is presented in Fig. 4, which was inspired by an illustration of KMeans clustering in chapter 20 of "An Example Inference Task: Clustering" [MMK03].

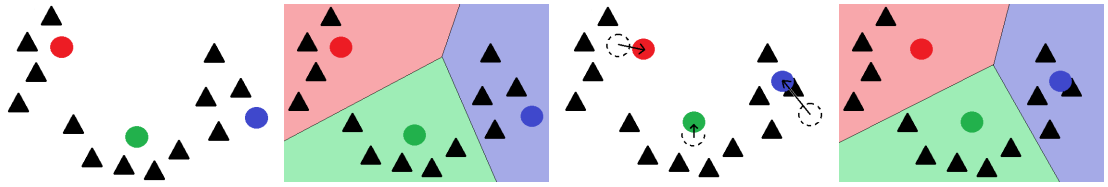
KMeans is carried out through the following steps:

1. A particular number of centers specified in the model setup is randomly generated within the data corpus (in Fig. 4a, the centers are presented as colored circles and the data points as black triangles).

2. To connect every data point with the nearest center, clusters are formed (colored areas in Fig. 4b).

3. The arithmetic mean of all data points in the respective area is taken as the new center (in Fig. 4c, the new centers are presented as colored dots, and the old centers are presented as dotted circles).

4. Steps 2 and 3 are repeated until a convergence is reached (the clusters presented in Fig. 4d are based on the centers from Fig. 4c).



(a) Step 1. (b) Step 2. (c) Step 3. (d) Step 4.

Figure 4. Example of KMeans clustering algorithm performance with three clusters.

2.1.5 Graph Theory

In discrete mathematics and graph theory, a graph presents a set of elements that have connections between certain pairs. The elements are called vertices or nodes, and the connections among them are called edges or links. Usually, a graph is defined as an ordered couple $G = (V, E)$, where V is a set of vertices and E is a set of edges.

Edges can be directed or undirected. For example, a connection between a company and a product that it sells is directed because "company A sells product B" is a one-way relationship that cannot be reversed. Graphs with this type of link are "directed." Otherwise, a connection between two similar products is undirected: for example, "product C is similar to product D" is equal to "product D is similar to product C", so it is a two-way relationship. Graphs with this type of link are "undirected."

In the graphs created during the current work, both company–product and product–product connections are presented. Hence, the graphs have both directed and undirected links. Such graphs are called "mixed" and are defined as ordered triples $G = (V, E, A)$, where V is a set of vertices, E is a set of undirected edges, and A is a set of directed edges.

One particular type of the graph that should be mentioned is a complete graph. In a complete graph, each pair of nodes is connected. Hence, a complete graph contains the maximum possible amount of links N that can be calculated with Formula (4):

$$N = \binom{n}{2} = \frac{n!}{(n-2)!2!} = \frac{n(n-1)}{2}, \tag{4}$$

where n is the number of nodes in the graph.

2.2 Related works

This subchapter presents an overview of several works that are related to the current topic or are solving similar problems.

In a master's thesis by Koppel [Kop17], a pipeline was developed whose purpose it was to use linked data obtained through web crawling as input and then apply feature extraction on the extracted RDF data. The purpose of the algorithm presented in the current work is to contribute input features for a machine-learning model for credit-scoring. The need for this algorithm arose from the needs of the company Register OÜ, which specializes in the development of IT solutions for credit management and sales. The use case of Koppel [Kop17] was obtaining linked data about the products of Estonian companies and connecting them with the companies themselves. The pipeline designed to identify situations in which two products from different companies belong to the same type of product or even are the same product. All companies and their products are represented on a graph, from which a set of network metrics was extracted by applying a number of graph processing methods (for example, Average Nearest

Neighbour Degree and PageRank) on it. They were saved as a time series with further derivatives calculations.

In general, all the steps of the proposed pipeline are as follows:

1. Resource Description Framework data about Estonian products are selected as N-quads from the web.
2. The obtained data are attached to the respective Estonian companies.
3. Products are linked to discover the same or similar products from different companies.
4. A linked company–product graph is created based on product SKUs.
5. Network metrics are calculated based on the obtained graph.
6. Features are extracted from the obtained metrics.

The entire pipeline was created to process terabytes of data, and for this purpose, Apache Spark was used. The thesis [Kop17] focuses on the performance of the whole pipeline from beginning to end. The presented algorithm satisfies all the requirements established by the company—speed and scalability.

But the presented approach also has some issues. As mentioned in the conclusion of Koppel [Kop17] thesis, only 13,699 products with unique SKU were connected to some companies, that correspond to 1.9% of all source products. Among those 13,699 products amount of products that were linked to the more than one company is less than 300 products. This implies that the approach can be improved by creating a larger number of links between products.

In the paper, a couple of ways to improve performance were suggested:

- Reduce duplicates in the source data.
- Use more advantageous methods to create more product links between products in the graph.

The second suggestion is the basis of the current thesis, and the first one was implemented during the data preparation. As a more advantageous technique to create more product links, the chosen method was topic modeling because, as was proposed by Koppel, products have names and descriptions more often than they have Stock Keeping Unit, so it makes sense to apply an NLP approach to the descriptions. The results of these improvements are described in chapter 4.1 Results.

During the process of related works investigation, papers were taken into account if they have common tasks related to entity linking and graph building, preferable with using of text analyzing methods.

Several works are solving the task of entity linking ambiguity. Entity linking itself is an ambiguous task because one word in a different context could have different meanings. The task is to define an appropriate option depends on the current context.

In X. Han, L. Sun, and J. Zhao article [HSZ11] consider a task of Collective Entity Linking (CEL). It is a type of entity linking, where links between entities are built based on interdependence between them. Authors describe the graph-based approach of Collective Entity Linking, that can choose the most appropriate option among variants proposed by EL, based on interdependence among entities. X. Han, L. Sun, and J. Zhao proposed to investigate global interdependence between all entities, rather than perform comparison pairwise. To compose those global meaning they used Referent Graph representation, based on which algorithm of finding the best EL option was built. Based on authors conclusion, obtained results are sufficient for solving entity linking ambiguity task and the proposed method could be competitive to traditional approaches of entity linking. Proposed approach is more useful for holistic texts, such as Wikipedia, where proper nouns could be used in different meaning, so graph based on interdependence among entities could be built. In the current work, entities in products descriptions are not so ambiguous to provide enough data for such graph. Mostly, they are concern some product characteristics, so this approach is not appropriate for products linking task.

The H. Chen et al. research in [CWL+18] also focused on solving the problem of entity linking ambiguity. They consider EL as a ranking problem, and especially for that developed the new embedding approach. Author called it bilinear joint learning model (BJLM). This model determining embeddings of word and entity, that is determined in various distributed spaces. Results show that the proposed approach improved the quality of EL algorithm. The proposed approach is not really useful in the current work, because in H. Chen et al. paper suggested that words and entities are located in different spaces and compliance should be established. In the current task words from products descriptions and entities are the same, so the described method is not really needed.

Another work considers entity linking ambiguity problem is paper [HRN+13] by B. Hachey et al., where three famous EL systems (cross-document coreference resolution (CDCR), wikification and named entity linking (NEL)) were re-implemented and evaluated. Those systems are working in two phrases: they define a set of possible meanings for the entity, and then choose the one. Authors used Wikipedia Linked data for testing and discovered that the second step is more important than it was considered before. Usually, for EL systems first phase is quite similar, when exactly the second makes the main differences between them. Described results have potential, but not really useful in the current work because each entity from products descriptions have a single meaning.

In P. Ristoski and H. Paulheim [RP16] paper proposed a new technique for open linked data processing. Authors consider that linked data have a graph structure to develop RDF2Vec. This approach is based on language modeling, with a purpose to extract features from sequences of words in an unsupervised way and finally converts them to

RDF graph. Authors divided the process into two main steps: graph transformation to the set of sequences and using those sequences to train language model. This model evaluates the probability of each sequence to be presented in the graph. In the output, every entity will be presented as a vector. P. Ristoski and H. Paulheim in their paper tested the proposed approach on the classification and regression tasks. RDF2Vec is efficient because it uses not just words order, but links between entities of the graph. In the current thesis, source data are already extracted from the web and saved as N-quads, so it is easier to work with it as a data frame, so RFD2Vec algorithm does not fit.

An example of work [GPL+06] where authors consider the task of entity linking using text descriptions is paper by R. Ghani et al. They worked with retail companies and figured out that new products are adding to the database manually. To make this process automatic and faster, authors developed an approach for extracting from products descriptions attribute-value pairs for further linking with other products. Hence, each product will correspond with the attribute-value pair. This representation is useful in tasks, where handling products as such pairs are more beneficial than a single structure. R. Ghani et al. presented two approaches for pairs extraction: one extracts implicit attributes from products, another - explicit ones. Authors state that their approach could be applied to the huge amount of non-labeled data and provide good results. They describe how to use this method for adding items to the recommendation system and show that it provides sufficient results. This approach is useful when the task is to fit some unstructured products into a structured database. But in the current work, all products are unstructured so attribute-value pairs extraction is meaningless.

In [KGAF11] A. Kannan et al. authors also consider a task of product linking based on their descriptions. They investigated linking products descriptions from unstructured textual data to entities from the structured product database. Authors proposed to use semantic parsing on descriptions. This parsing approach is based on the dictionary, produced by structured product database. Also, A. Kannan et al. updated matching function, so it considers not only possible matches but also mismatches and missing attributes. Authors implemented their solution as a test search engine for well-structured products set with detailed attributes. Their approach showed high F1 score[8], comparing to the results of standard approaches. In general, the proposed method is efficient for the structured products set when data in the current thesis is unstructured. Hence, this approach cannot be used in the current use case.

In products matching, duplicates detection is an actual problem. R. van Bezu et al. in their paper [vBBR+15] consider this problem and propose their approach. It takes into account the product's title and attributes similarities separately and then combine them in one general similarity score. To handle with abbreviations, authors use n-grams[9]. To handle with a various assortment of web-shops, authors use a hierarchical clustering approach, instead of clustering for just two shops. They evaluated the proposed approach

[8]F1 is a score of test accuracy.

on a set of products from web-shop and obtained results show better F1 score, than more popular methods. The size of test data for proposed approach is pretty small: 1,629 products against 450,096 products in the current thesis (their extraction descried in subchapter 3.1 Dataset description). Such methods as n-grams and hierarchical clustering are slow, so they are not appropriate for the current use case.

Regarding topic modeling approaches, they also were investigated in paper [VP14] by K. Vorontsov and A. Potapenko. Authors take a look at the background of Bayesian topic modeling methods and try to offer a non-Bayesian alternative. They consider probabilistic topic modeling approaches such as pLSA and LDA and decided to reduce their disadvantages with an alternative. Particularly, LDA may be considered as a two-layer Bayesian generative model, that assume about the distribution of topics over the words and documents over topics as about a prior Dirichlet distributions. That cause some disadvantages, such as no valid linguistic motivations, disagreement of sparsity and complexity of topic model which tries to contain all proper requirements. K. Vorontsov and A. Potapenko as an improvement of this approach propose a non-Bayesian method called Additive Regularization of Topic Models (ARTM), which should reduce the disadvantages of Bayesian methods and makes it easier to implement. Bayesian topic modeling approaches are based on the factorization of the stochastic matrix, that is an inverse issue with a variable solution. The target of improvement is to decrease a probably unlimited collection of answers and to choose a suitable one, which fits into all provided conditions. In the current work is important to find an appropriate way of topic modeling that also should be easily performed in the business environment. In this case, classical LDA is more suitable and its disadvantages are not so crucial.

Applying the text analyses approaches on big data can leads to memory problems. R. Řehůřek and P. Sojka in their paper [RS10] considered the problem of limited RAM during performing NLP tasks that include text vectorization. In such methods, parts of the text are presented as a vector in multidimensional vector space, which obviously requires a lot of memory for big data. To deal with this problem authors propose NLP software framework, based on document streaming approach. This framework was developed for Python programming language and allows to perform various text analytic tasks. To test the developed solution, R. Řehůřek and P. Sojka implemented with its LSA and LDA topic modeling methods. According to the authors, the proposed framework allows performing text parsing, transforming and processing for the huge size of corpora that doesn't fit into RAM. This research are potentially useful, but in this work for the current amount of data (450,096 products) is not needed.

In summary, in the current thesis would be used only results of Koppel thesis [Kop17] as the predecessor of this thesis. All other papers provided various methods, mostly not related to topic modeling, but those that were related imposed restrictions on the input data, inappropriate for the current task. All those methods won't be applicable in the

[9]N-gram is a connecting sequence of n elements, usually, words.

current work but could be used for its extensions. For example, for a bigger amount of data, for a structured dataset, for more various textual information, etc.

3 Methodology

This chapter provides an overview of the selection of source data and a description of the methods used in the current work.

3.1 Dataset description

The current subchapter presents an overview of the data provided by Register OÜ and the process of textual data selection and analyses for the further application of topic modeling methods.

3.1.1 Source data

As source data for the current thesis, 2,370,122 N-quads[10]were obtained from Register OÜ. The N-quads contain linked data about Estonian products gathered in August 2017 in microdata[11]format. The data were collected by the company using schema.org, an open-source collaborative community whose aim is to design, support, and develop structured data techniques using the Internet.[12]

Initially, it was unknown how many web pages, companies, or products were stored in the obtained data. An example of the source data is presented in Fig. 5.

Link	Node	Characteristic	Content
<http::www.dreamstuudio.ee::/::product=man-sho...	_:node6343f372257f756b3cd891fa70876a	http://schema.org/Product/description	\n\tPellentesque habitant morbi tristique sene...
<http::www.dreamstuudio.ee::/::product=man-sho...	_:node6343f372257f756b3cd891fa70876a	http://schema.org/Product/name	Man shoes@et
<http::www.dreamstuudio.ee::/::product=man-sho...	_:node6343f372257f756b3cd891fa70876a	http://schema.org/Product/offers	:nodeb2c5584e63aecec4ecbf181283f8cd2d
<http::www.dreamstuudio.ee::/::product=man-sho...	_:node6343f372257f756b3cd891fa70876a	http://schema.org/Product/aggregateRating	:node9ec24f30f2a319962d2d65b8e51bf90
<http::www.dreamstuudio.ee::/::product=man-sho...	_:node6343f372257f756b3cd891fa70876a	http://schema.org/Product/review	:node5c3c518c0ed541055221f2df661ae85

Figure 5. Examples of source N-quads.

The first component of an N-quad is a web link of the product, which presents the seller company; the second is the product node, which presents the unique id; the third is the name of product characteristic; and the fourth is content of the characteristic named in the third column.

The intention was to rewrite the data from the N-quads into Pandas DataFrame, a more appropriate format for further analysis. In the source data, each product characteristic

[10]N-quad is a line-based plain text format for encoding an RDF dataset.
[11]A HTML specification for fitting metadata inside existing content of websites.
[12]https://www.w3.org/community/meat/2016/02/04/more-about-our-mission/

is separate, so it is unknown how many and what kind of characteristics each product has or even how many products are presented. To make it easier to work with the data and collect statistics about the products, it makes sense to rewrite the data as follows: Each row represents one unique product, and each column represents one characteristic among all possible characteristics of all products in the source data.

The results of DataFrame are illustrated in Fig. 6, where the first two columns are the same as the first two parts of the N-quads—links and the nodes of unique products. The other columns are the unique characteristics of products from the third part of the N-quads, and their content comes from the fourth part of the N-quads.

Link	Node	Image	...	URL
<https::www.paradiis.ee::/ru/22-tv-alused::nul...	node7a429218c67498c6039c111e9dddd3a	https://www.paradiis.ee/5654-home_default/mont...	...	https://www.paradiis.ee/ru/tv-alused/2371-mont...
<http::tamberi.eu::/73-ehtekaardid::null::2017...	node6883b5caa683772129c48f93ac5e99	http://tamberi.eu/707-home_default/ehtekaart.jpg	...	http://tamberi.eu/ehtekaardid/1780-ehtekaart.html
<http::mulgimoto.ee::/::product=eelsuute-relee...	node62d9afc3e4cd82a16e3f201ac4ea92	http://mulgimoto.ee/wp-content/uploads/2016/01...	...	http://mulgimoto.ee/?product=eelsuute-releeyanmar
<https::nailin.ee::/et/27-akruulvarvid::null::...	node517e8478b3641044f1e2e2bbc31afa	https://nailin.ee/2012-home_default/akruulvarv...	...	https://nailin.ee/et/akruulvarvid/3201-akruulv...
<http::www.kalastussport.ee::/136-taliridvad::...	nodef5d0383018d768c2279c3541adc8f1a	http://www.kalastussport.ee/img/p/et-default-h...	...	http://www.kalastussport.ee/taliridvad/466-win...

Figure 6. Part of the performed DataFrame analysis with one product per row.

The results obtained from DataFrame contained 450,096 rows and 59 columns.

3.1.2 Data preprocessing

To apply topic-modeling methods, it is necessary to select among the entirety of the presented data non-empty columns with textual data, detect the language of the text, and merge the information from those columns. Those steps were performed to obtain a solid textual monolingual description for each product.

This subchapter presents the selection of non-empty textual columns with further language detection.

In the beginning, it was necessary to select among all 59 columns non-empty columns with enough data for further analysis. Fig. 7 presents 24 columns that have at least 0.1% non-empty content.

Most of the columns have a value of “null” because each product has a particular set of characteristics, as presented in the third and fourth elements of the N-quads. All products have “link” and “node” characteristics from the first and second elements of the N-quads as identification keys that describe each product as a unique item. However, the other characteristics are not so common or even are unique to one only product. For example, only approximately 1% of the products have a “price” characteristic. Nonetheless, there are 10 characteristics columns with an amount of non-empty data larger than 5%. They

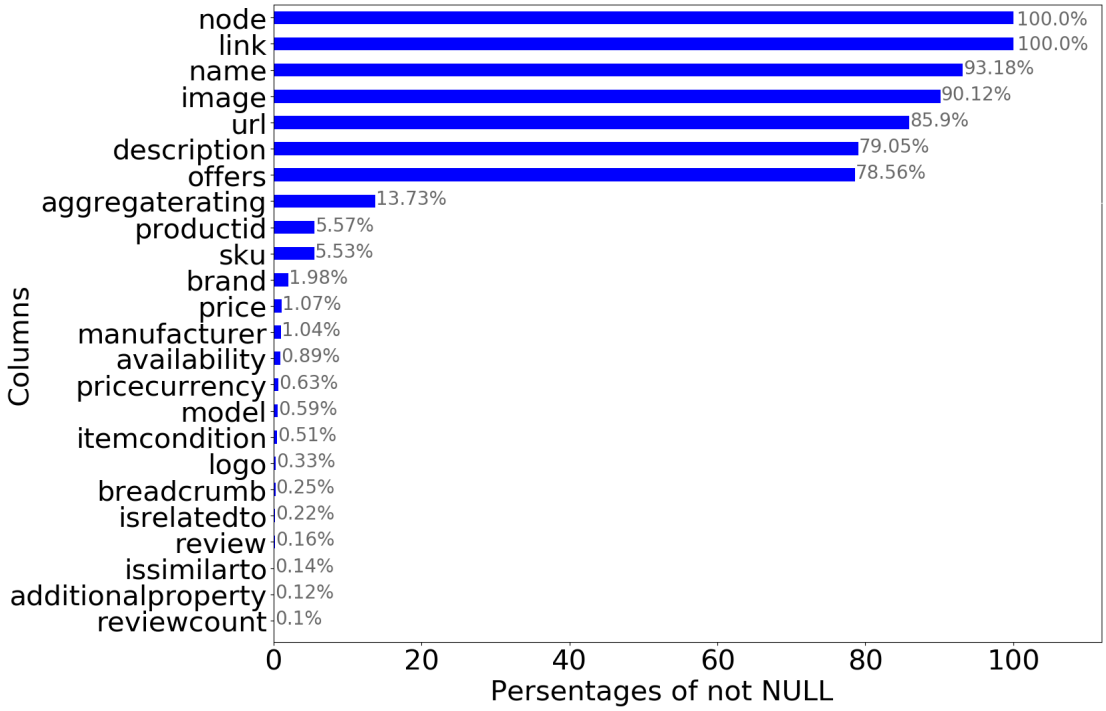


Figure 7. Product characteristics that have at least 0.1% of non-empty data.

provide some data about the product and can be selected for further analysis. Among them are three columns with textual data: "Description," "Productid," and "Name."

To ensure that the data from the textual columns are appropriate to use in the topic modeling and will provide meaningful information, the percentage of useful data was calculated for each column. Useful data are considered data with potential information about the product that makes sense to a human reader and contains at least three letters. Table 1 presents the size of such useful data in percentages, the average number of words, and one random sample of text for each of the three textual columns. To ensure that the data from the textual columns are appropriate to use in the topic modeling and will provide meaningful information, the amount of useful data was calculated for each column. Useful data are considered data with potential information about the product that makes sense to a human reader and contains at least three letters. Table 1 presents the size of such useful data in percentages, the average number of words, and one random sample of text for each of the three textual columns.

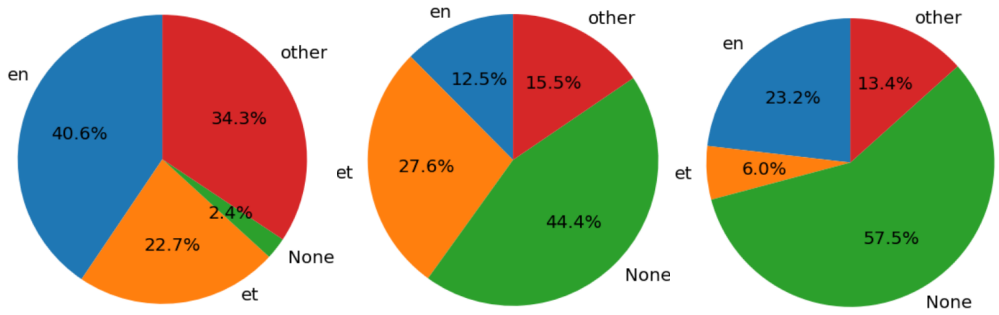
As can be observed from Table 1, the "Name" column has the largest percentage of useful data but only a small average number of words in each instance. "Productid" has the smallest percentage, but it contains keywords that can be useful for further topic modeling, as presented in the "Example of instances" column in Table 1.

Table 1. Description of text columns.

Column name	Size of useful data, %	Average number of words	Example of instance
Name	93	6	Digital pressure sensor BMP280
Description	55	38	See esinduslik aiamööbel on valmistatud romantilises ja ajatus lossipargi stiilis!
Productid	1.5	4	Bird Garden Duck Egg

Regarding the next step, to determine whether it is possible to merge information from textual columns together to form a solid description of the product, the language distribution of the text inside the selected three columns was explored. For each instance in each column, three main languages were detected. The algorithm of the language detection is described in detail in subchapter 3.2.1 Language detection.

As a result of the language detection process, it was discovered that almost all instances in all columns have text only in one language. Illustration in Fig. 8 presents results of the language for each instance.



(a) "Name" (b) "Description" (c) "Product id"

Figure 8. Language distribution for textual columns.

For the "Name" column (Fig. 8a), the "None" result is almost empty because nearly 97% of data is useful. The most popular languages are English and Estonian. Field "other" corresponds to the sum of languages, where each language has a size less than 5%.

For the "Description" column (Fig. 8b), the "None" is the most popular result, because only half of the data is meaningful, and the main languages are, again, English and Estonian. Field "other" corresponds to the sum of languages, where each language has a size less than 3%.

For the "Product id" column (Fig. 8c), the results are similar to those for the

"Description" column. As expected, the "None" result is the largest because only 1.5% of the data in this column are useful. Field "other" corresponds to the sum of languages, where each language has a size less than 2%.

Almost all the language results for all three columns have a probability of more than 0.8. This score means that for any particular language detection result, a certain proportion of the input text definitely belongs to the detected language. For example, if the results suggest that the presented text is English with a score of 0.8, at least 80% of the text is definitely English and the remaining 20% is unknown—it could be company or product names, words that are common in different languages, etc. So, in the current situation, it can be concluded that the obtained results are reliable.

3.1.3 Merging data

This subchapter presents the merging of data from the textual columns based on language to obtain a solid monolingual description for each product.

The text was merged in the following way:

1. The languages of the “description,” “name,” and “productid” columns were compared, and if they were the same with a probability larger than 0.5, the corresponding texts were merged.

2. If, for a given product, no matches were found, text in English or Estonian with a probability larger than 0.5 was chosen as the joined text.

In the end, for 73.6% of products, the textual information contained text from more than one column were used to form the joined text.

As expected, English and Estonian are the main languages. All other languages combined make up only 3% of the non-empty joined text columns. They were omitted in subsequent steps because this 3% of data would have provided only a minuscule amount of useful information, but for each language, it would have been necessary to use separate libraries and create separate models. It would not have been effective to, for example, implement the clustering model on several products that would not provide much useful information. Proper nouns that present organization names were recognized as English and weren’t deleted.

In general, joined text was obtained for 71.4% of all initial products (those that contained Estonian or English text with a high probability). In numbers, this constitutes 321,368 products from the original 450,096 products. The number of instances with English joined text is 58.2% of all selected instances, which equals 187,036 products, and the remaining 1.8% are Estonian - 134,332 products.

A word cloud was generated to shed light on the context of the obtained data. Creating a word cloud is not a real method of topic modeling because the output of this method is a single picture with the set of words with the highest frequency in the input text (words

with a larger frequency appear larger in the image). The results cannot be used in further analyses or be applied in any method, but they can give the user an overview of the data context. This method can be especially useful if the user does not know what the data is actually about, as in the current case. The data obtained from the company were without any initial topic specification, but creating word clouds (Fig. 9, 10) helped give an overview of the topics.

Figure 9. Word cloud for products with English description

Figure 10. Word cloud for products with Estonian description

As can be observed from results, the products with English text mostly seem to be connected to computer or mobile equipment and those with Estonian text to clothes and their characteristics.

3.2 Methods

This subchapter presents the methods of text preparation, topic modeling, and graph-building used during the project.

After obtaining a text description of the products, the next step is text preparation before using it as input for topic modeling (subchapter 3.2.2) and language detection (subchapter 3.2.1). After that topic modeling - a statistical model whose purpose is to discover the abstract topics presented in a set of documents, was performed. Topics are represented by a set of keywords that describe the main information in the source data.

The standard approach for topic modeling is LDA, which requires a specification of how many clusters should be obtained. This amount of clusters is not obvious from the beginning, usually, domain experts could provide an approximate number. But when data content is not well-known, as in the current work, the number of clusters should be measured with some technique. For this purpose, a coherence measure was developed.

Because the target of this thesis is to improve entity linking, it is worthwhile to test different methods of topic modeling to determine the best one. Besides latent Dirichlet allocation, three more methods were performed:

- Keywords extraction with KMeans clustering, performed on words from products descriptions.
- Keywords extraction with TF-IDF scores, calculated for words from products descriptions.
- Products description clustering with DBSCAN algorithm.

All those methods are described in chapters 3.2.3–3.2.6.

3.2.1 Language detection approach

For efficient language detection of the input text, a function was developed for separating texts into several subtexts based on language. This function is useful in the current thesis because different product descriptions are written in different languages or even contain several languages. Since it is necessary to apply different language-based techniques, such as part-of-speech tagging, described in the subchapter 3.2.2 Text preparation, to preprocess the text before using it in topic-modeling methods, it is important to know what the language of the text is and whether it is uni- or multilingual.

The following is a usage example of the language separation function: a user has some text from a web page and needs to perform part-of-speech (POS) tagging on it. The user needs to know the language of the text to choose the correct package. However, the text contains paragraphs in Estonian, English, and Russian. The output of the developing function will be three different texts: the first will contain only Estonian text, the second

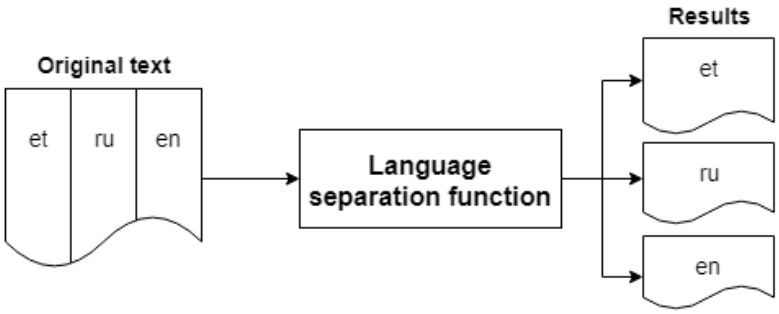


Figure 11. Example of language separation function usage.

only English, and the third only Russian, as presented in Fig. 11. Then, the user can apply POS tagging using the appropriate packages for each output text separately.

For this task, Python (V3.) was used, and different text analysis libraries were evaluated. The aim was to determine the best library or the combination of libraries (the way of defining the best result among obtained). The following packages were chosen as the most popular ones and tested: Langdetect[13], Polyglot[14], Langid.py[15] and Spacy CLD[16].

The process of the task is as follows:

1. Different Python libraries were investigated and tested with the following criteria: possibility to detect several languages in a single text, recognition of characters used in Nordic and Baltic languages, performance speed, and minimum text-size limitations.

2. The best library, or the best combination of libraries, was selected.

3. A function for language-based text separation was designed and tested.

The main measurements in the method comparison were as follows:

- The minimum amount of symbols needed to define the text as belonging to a particular language.

- Recognition of Nordic and Baltic characters (because the source data are about Estonian companies, which could have products descriptions in those languages).

- Possibility to detect different languages in a particular text.

[13]https://pypi.org/project/langdetect/
[14]https://pypi.org/project/polyglot/
[15]https://github.com/saffsd/langid.py
[16]https://github.com/nickdavidhaynes/spacy-cld

- Language recognition speed.

After the packages were investigated and tested, the best ones were determined to be Polyglot and Langdetect. Both methods were better than the others in all listed criteria - Langid.py and CLD showed slower performance, higher minimal amount of symbols needed for correct language detection and difficulties with multilingual text recognition.

Polyglot and Langdetect have advantages and disadvantages that complement each other. They are described in more detail in Table 2.

Table 2. Comparison of the Polyglot and Langdetect language detection methods.

Tool	**Advantages**	**Disadvantages**
Polyglot	- Can recognize different languages in one given text (for example, works well for the combination of English, Estonian and Russian) - The fastest method	- Has problems recognizing Nordic and Baltic characters - Does not work well for short texts - Has a bottom border of length for words equal to three characters
Langdetect	- Can recognize Nordic and Baltic characters correctly - Almost as fast as Polyglot - Does not have size limitations	- Has difficulties with multilingual texts (for example, in the combination of English, Estonian and Russian, it can only recognize the biggest one)

Neither of those methods was sufficient by themselves, so it was decided to combine them to complement the disadvantages of each other. For example, Langdetect can detect the language in short texts with Nordic/Baltic characters, while Polyglot can detect several languages in one given text. Also, they both are quick. The way this thesis combines both methods is illustrated in the diagram in Fig. 12, where "poly" represents the results of the Polyglot method and "lang" those of Langdetect.

Polyglot is prioritized because this method works better for multilingual texts, but for very short texts, it does not work at all. The first step is to check whether Polyglot can even generate a result, and if so, whether it is equal to the result of Langdetect. If they can both generate results and are not equal, the best method is the one whose result has the larger probability. This probability represents the approximate proportion of the input text that definitely belongs to the obtained language.

Before using this approach, it is recommended to convert all text to lowercase letters and clear numerals, punctuation marks, and other non-letter characters.

To separate the provided text data into subtexts by language, a separation algorithm was developed:

1. Clean data was generated by converting text to lowercase letters and eliminating all non-letter characters.

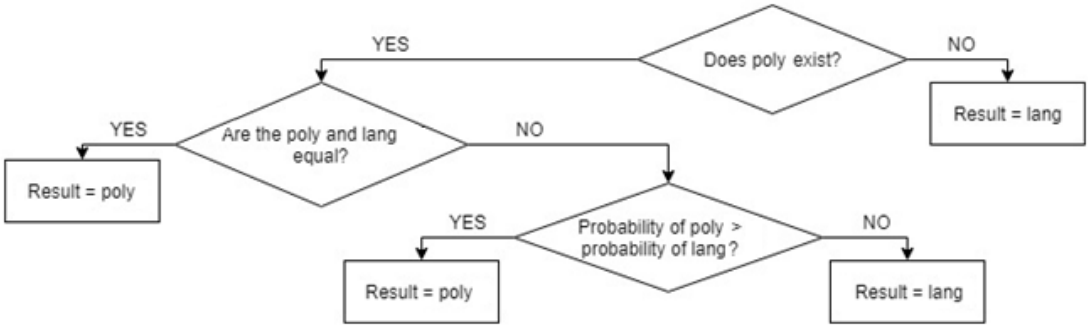


Figure 12. Results of the combination of the Polyglot and Langdetect methods.

2. The text was split into tokens (words).
3. Language detection using Polyglot and Langdetect was performed for each token.
4. A language was selected for each token based on the schema shown in Fig. 12.
5. A token in the corresponding language was added and its probability saved.
6. The three largest lists of obtained languages were selected as the main separated languages.
7. The probabilities for each of the three texts were calculated as the average probability of the tokens inside each text, and the results were saved.

In general, this function works well and was used in the current thesis during the data preprocessing stage.

3.2.2 Text preparation

After all steps described in the dataset description part were conducted, a total of 321,368 products with a textual description (187,036 products in English and 134,332 in Estonian) were obtained. Before these descriptions could be used in topic-modeling approaches, the texts had to be cleaned and prepared.

Before starting any data manipulation, it is vital to delete duplicates in the obtained noun lists to not put added weight on subsequent calculations. Duplicates are products from the same company with identical descriptions.

Examples of duplicates:

- In the English data: a product with the description "Mizon Original Skin Energy Hyaluronic Acid" from the company "koreacosmetics.eu" has 21 copies.

- In the Estonian data: a product with the description "Majapidamispaberi hoidja" from the company "freshdesign.ee" has 34 copies.

Deleting duplicates was mentioned by Koppel [Kop17] as a possible improvement of the proposed pipeline, so it is important to pay attention to and explore it. There is a large number of duplicates in the source data because the same information about the same product is located on different web pages, and the same web page can appear several times in the web crawling results. Therefore, the same product can be presented in different triples and have multiple entities in the source data.

Regarding the duplicate detection results, for English, 23.2% of the original amount of text was unique descriptions, and for Estonian, the percentage was 23.1%. After removing duplicates, 43,392 products with English descriptions and 31,165 with Estonian descriptions remained. Following the deletion of duplicates, subsequent steps should be executed faster.

Afterward, numerals and punctuation marks were deleted from product descriptions. Then, as the main step of the data preparation, part-of-speech tagging was performed. The POS tagging technique tags every word in the text with the appropriate part of speech. This method is based on the meaning of the words and the general context of the text. In the current thesis, to compose a set of topics, it is necessary to extract only nouns. Nouns are needed because in further topic modeling, each topic is presented as a set of keywords where each keyword is a noun to avoid meaningless verbs or adjectives out of context. Furthermore, this approach clears stopwords[17]from the source text, which might have too large of a frequency and appear in topics that they should not appear in.

[18]Stopwords - words that are useless for analyses and should be deleted from the text before processing. Because two languages—English and Estonian—are presented in the data, it is necessary to perform POS-tagging for each language separately. Different libraries and sets of POS tags exist for different languages, but the sense remains the same. For English, the Natural Language Toolkit (NLTK) library was used, with noun codes “NN,” “NNS,” “NNP,” “NNPS,” which correspond to common and proper nouns in singular and plural forms. For Estonian, the Estonian Natural Language Toolkit (ESTNLTK) library was used, with nouns codes “H” and “S,” which correspond to common and proper nouns. For subsequent steps, only words with these tags were selected from each dataset.

Afterward, for each product, a set of nouns was obtained as the description and will constitute the data for further topic-modeling approaches.

3.2.3 Topic modeling with LDA

A more detailed description of the LDA approach and its predecessors is presented in chapter 2.1 Background. In the current subchapter, a practical application of this approach and its evaluation method in the current work are presented.

For the current thesis, a realization of the latent Dirichlet allocation model implemented in the Gensim[19]package was chosen. This is the most popular approach, enabling the measurement of coherence and interactive visualization.

Before the LDA method can be applied, it is necessary to detect and specify the number of output topics that the model should find. The typical evaluation method for determining the optimal number of topics that should be specified before the training of the LDA model is coherence. The coherence score illustrates how strongly connected the obtained topics are and whether they overlap. In other words, coherence answers the question of how close the objects are, particularly terms inside the same cluster. A larger coherence value corresponds to a better number of topics.

Measurements such as coherence should be used because the results of topic-modeling techniques do not promise interpretability. They just provide a way to organize and summarize a huge number of data, and for this reason, different evaluation approaches are used.

As the method of coherence calculation, the coherence verification (CV) measure was used. Hinneburg et al. in their paper [RBH15] describe CV as the best-performing coherence measure. This approach was discovered in 2015, the same year as Hinneburg et al.'s publication, during a methodical study of coherence measures. The CV measure is based on a combination of the following techniques:

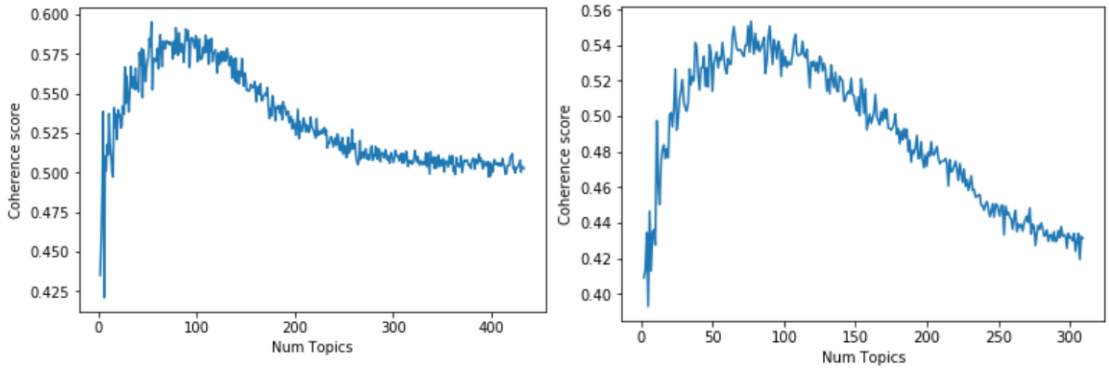
- Boolean sliding windows.
- Top-term segmentation.
- Indirect cosine similarity.
- A verification measure that uses normalized pointwise mutual information (NPMI).

To measure coherence in the current thesis, the CoherenceModel[20]method from the Gensim package was used, as was proposed by K. Kumar [Kum18]. This implementation specifies the CV measure as well.

To define the optimal number of clusters, several LDA models were built, and their coherence scores were saved, with topic sizes starting at two and ending at the size equal to 1% of the source data (434 topics for English and 310 for Estonian). Those numbers were chosen to be large enough to cover all possible hidden topics and, if needed based on the results, have the opportunity to increase (but as seen from the results, increasing the number of topics is not necessary). As illustrated in Fig. 13, the largest value of coherence for the English data is equal to 0.59 and corresponds to 54 topics, and for Estonian, it is 0.55, or 77 topics. Thus, 54 and 77 are the optimal number of topics for the LDA model training for the respective languages.

[19]https://pypi.org/project/gensim/

[20]https://radimrehurek.com/gensim/models/coherencemodel.html



(a) English.

(b) Estonian.

Figure 13. Coherence score for English (a) and Estonian (b) data.

As described in S. Li paper [Li18], LDA can use word embedding based the BOW or TF-IDF scores, both of which are described in chapter 2.1 Background.

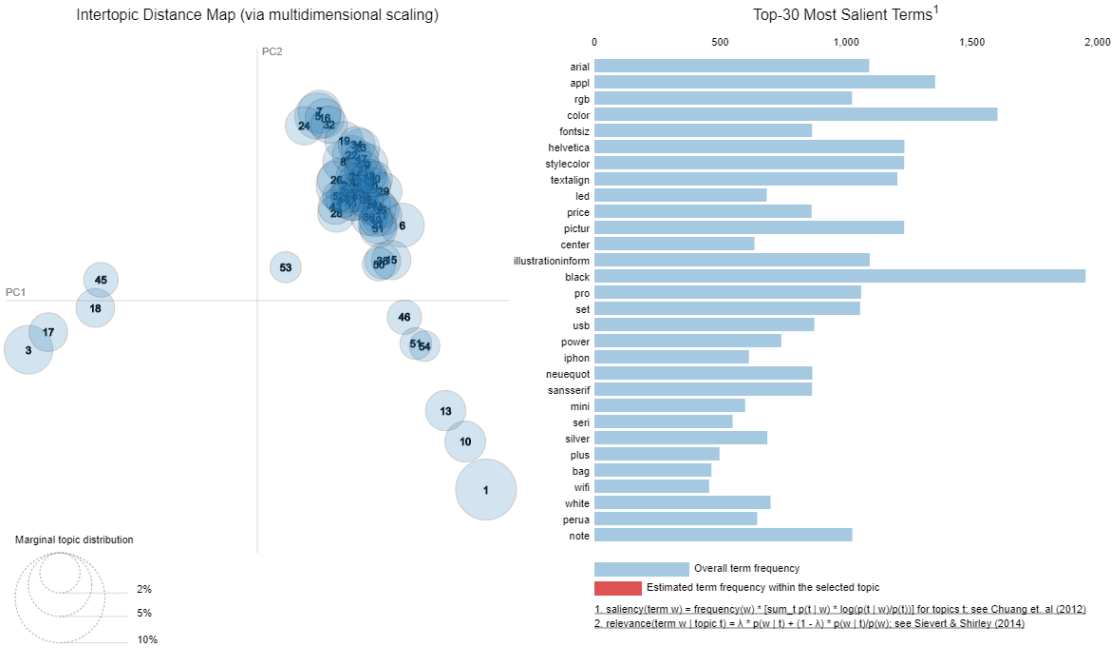
The BOW and TF-IDF scores are used to build vectors from corpora that will further be used as parameters for the LDA model setting. In the current thesis, both approaches were implemented and tested, and the appropriate visualizations are shown in chapter 4.1 Results.

Visualization of the obtained LDA models topics was carried out with pyLDAvis[21], a Python tool for building interactive topic-modeling visualizations. The results of the pyLDAvis visualization are presented in Fig. 14a-14b for English topics and in Fig. 15a-15b for Estonian.

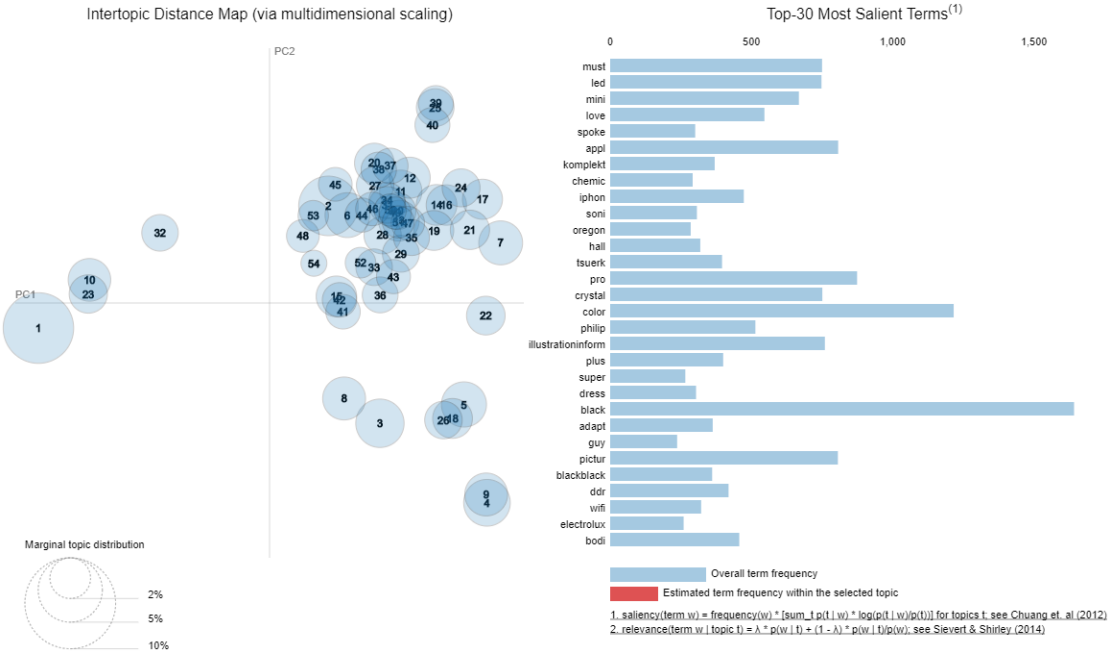
As can be seen in Fig. 14- 15, overlapping topics exist in each case. The largest amount of overlapping is illustrated in Fig. 14a, which shows the English topics based on the BOW approach. Otherwise, in every other picture, at least a couple of topics are located separately at a large distance from the others. They probably represent noise topics, which are collected products that do not fit in any other category. An investigation of the obtained results and a comparison with other approaches is presented in chapter 4.1 Results.

Obtained topics can be conducted by defining clusters because each topic contains a set of texts with common points. Therefore, products in each cluster can be connected as similar products. The output graph is presented in chapter 4.1 Results in Fig. 23 for English and Fig. 24 for Estonian.

[21] https://pypi.org/project/pyLDAvis/

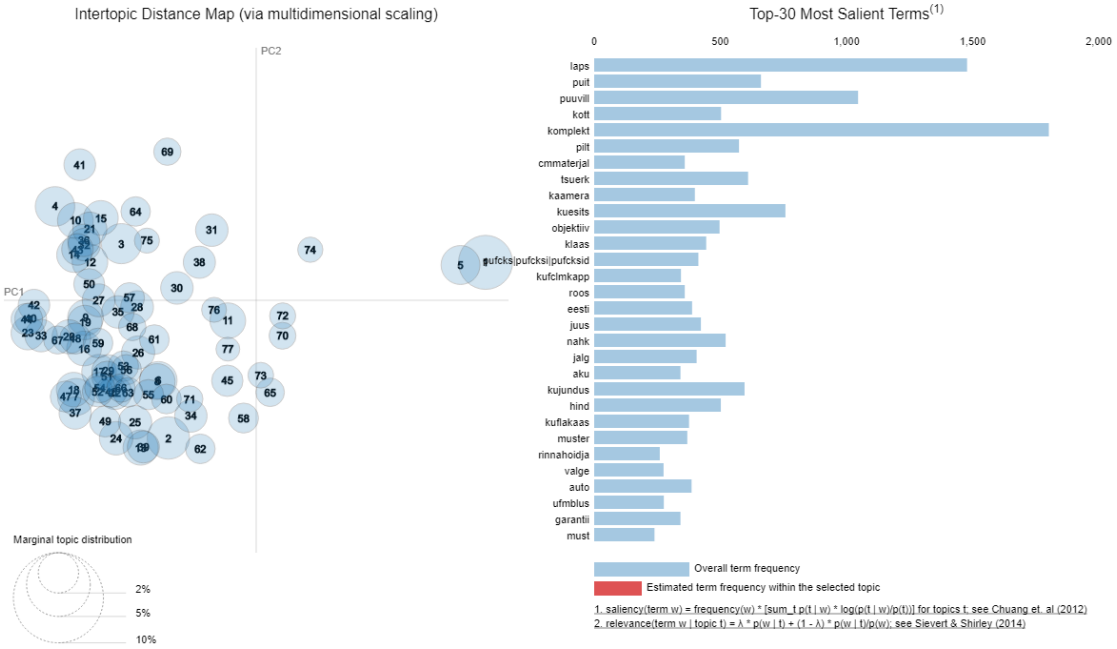


(a) BOW based.

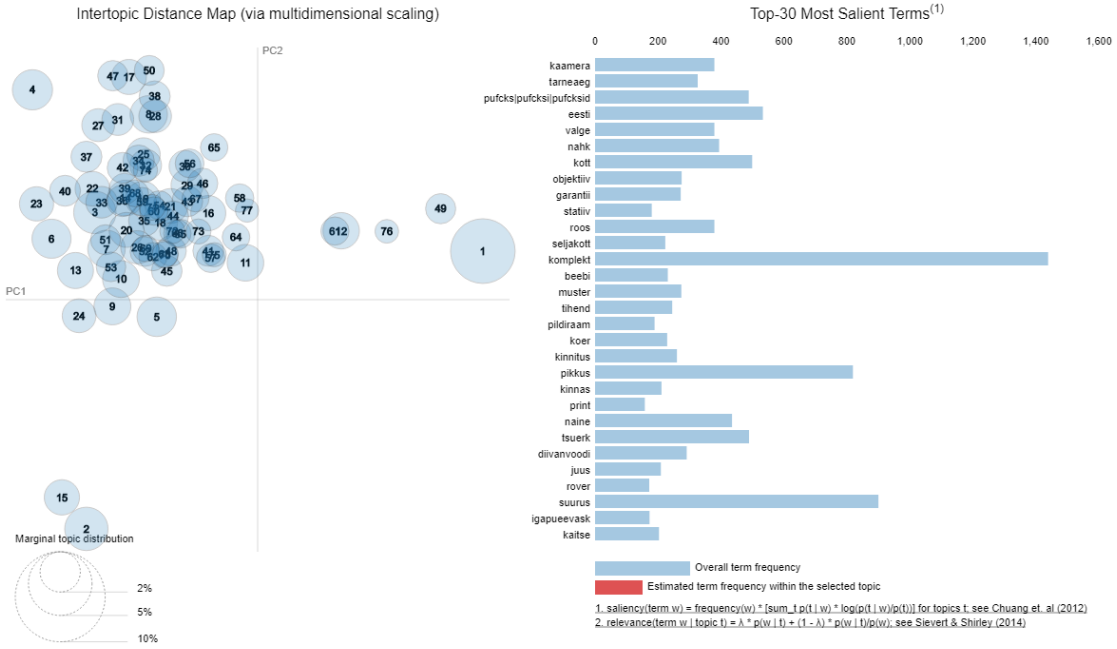


(b) TF-IDF based.

Figure 14. Topics obtained with LDA for English data.



(a) BOW based.



(b) TF-IDF based.

Figure 15. Topics obtained with LDA for Estonian data.

3.2.4 Texts clustering using DBSCAN

The main idea of text clustering with DBSCAN is to perform text vectorization on product descriptions and then apply a clustering method on the obtained vectors. A set of product clusters where all products from the same cluster are connected to each other is obtained as a result.

Particularly, vectors obtained from descriptions describe similar products in similar ways and are presented as vectors close to each other in the chosen multidimensional vector space. Hence, the cluster represents a group of products with similar meanings that are connected to the graph.

To obtain vectors for product descriptions, Doc2Vec was used. In the current thesis, product descriptions were taken as a set of texts such that every vector represents one product. For the experiments, the Gensim implementation of Doc2Vec[22]was chosen. As was illustrated by D. Mishra paper [Mis18], Gensim provides a simple method of Doc2Vec model training. The Doc2Vec model is built on the base of tagged data obtained from the source data with Gensim's TaggedDocument method, which represents an input document as a tag appropriate for the Doc2Vec model. To perform text vectorization appropriately, separate models should be used for each language - one for English and one for Estonian. Both models were built in a 300-dimensional space, which is the usual size of Word2Vec or Doc2Vec models. A representative vector in 300-dimensional space was obtained as a result for each product description.

After the pre-trained models were obtained, a distance matrix was built for each language corpus. A distance matrix is a square matrix wherein the rows and columns present the same set of entities (in the current case, a set of products), and the values are the distance between the element in the row and the element in the column. Distance matrices are used in the current thesis because of their sparse format, which decreases the data size and expedites clustering. It is a practical approach when the source data are large web-linked data.

As a distance measurement in distance matrices, Euclidean distance is usually used. Euclidean distance represents the distance between two particular data elements within Euclidean space. However, when the source data are textual, cosine metrics, which illustrate the similarity between texts, should be used instead of Euclidean distance. Cosine metrics is useful when the vectors' location in multidimensional space, not their size, matters. As was described by C. Emmery paper [Emm17], cosine metrics are also applied to data where some features of elements have different weights without having any other differences.

Mathematically, cosine represents the angle between a pair of vectors that corresponds to a pair of texts, while Euclidean distance measures the actual distance between those vectors. The cosine metrics on the diagonal elements of a distance matrix are 1s because

[22]https://radimrehurek.com/gensim/models/doc2vec.html

the cosine between equal vectors is 1, and the metrics on the non-diagonal positions consist of values from 1 to 0, where 0 corresponds to opposite vectors. Additionally, distance matrices are symmetrical matrices because the distance between element 1 and element 2 is equal to the distance between element 2 and element 1.

In the current approach, to extract distances between vectors, the Gensim method of the Doc2Vec model called most_similar was used. In this method, for a given vector, a list of ten vectors with cosine metrics from each of them to the input vector is returned, with the values of the cosine metrics sorted from largest to smallest. For vectors that are not in the top 10, their value in the distance matrix is 0. Hence, a matrix with many zero values was obtained as a result and was saved using the sparse format.

The most popular clustering algorithm performed on distance matrices is the density-based spatial clustering of applications with noise (DBSCAN). A description of this method and its parameters is presented in chapter 2.1.4 Clustering methods. For the current thesis, the Scikit-learn implementation of DBSCAN[23]was used.

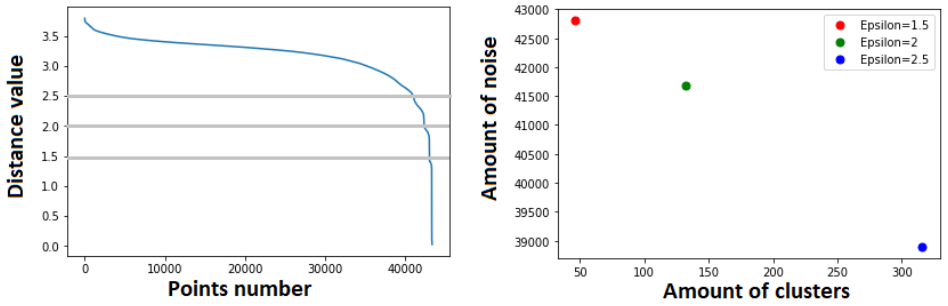
To implement DBSCAN for current source data, for both the English and Estonian data, a model was separately trained and the parameters eps and $minPts$ specified. The evaluation of each parameter was performed in a different way.

There is no one single popular opinion about obtaining $minPts$ measurements. The best way to obtain $minPts$ values as put forth by domain experts is impossible for the current work because of the unknown context of the extracted linked data. Furthermore, this value should not be less than 3 because otherwise noise points will be grouped into meaningless clusters. In the current work, the heuristic approach of D. Birant paper [BK07], who suggested setting up a $minPts \approx ln(n)$ where n is the size of the corpus, was used. The same heuristic approach also informs the eps parameter evaluation.

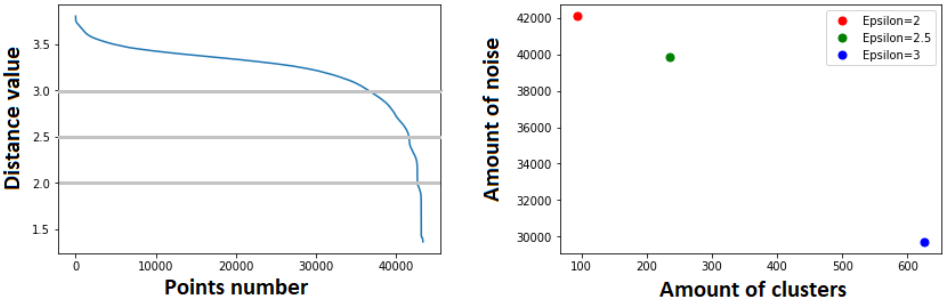
The value of eps is dependant on the value of $minPts$. To measure the approximate optimal eps value for a given $minPts$ value, it is first necessary to calculate the distance to the k-nearest neighbors for every element from the input data, where k is equal to the $minPts$ value. Afterward, the obtained distances should be sorted from largest to smallest and represented as a plot, where the y-axis represents the distance value and the x-axis the number of points with that distance. Then, a "knee" point, where the value of X increases significantly, as Y decreases (the appropriate value of the y-axis responds to the eps value), should be defined on the plot. In general, a plot can have several "knees," but the eps value should not be larger than the value corresponding to the first "knee" from the top. The approach for eps evaluation used in the current thesis is also described and illustrated in M. Ester et al. paper [EKS+96].

For the current thesis, a number range between $ln(n) - 1$ to $ln(n) + 2$, taking into account that $ln(n) - 1$ is bigger than appropriate minimum equal to 3, was chosen as the $minPts$ values to investigate different results. For the English data, $n = 43330, ln(43330) \approx 11$, and for the Estonian data, $n = 31043, ln(31043) \approx 10$. For

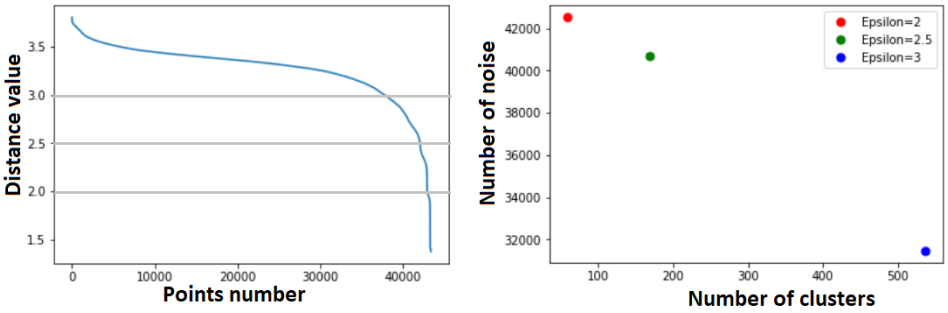
[23]https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html



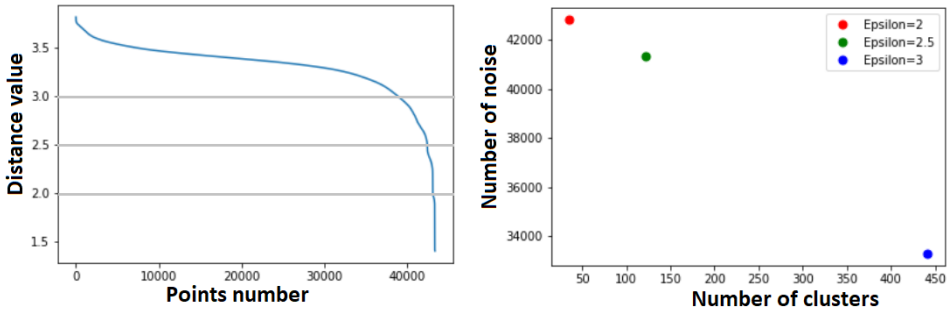
(a) minPts = 10



(b) minPts = 11

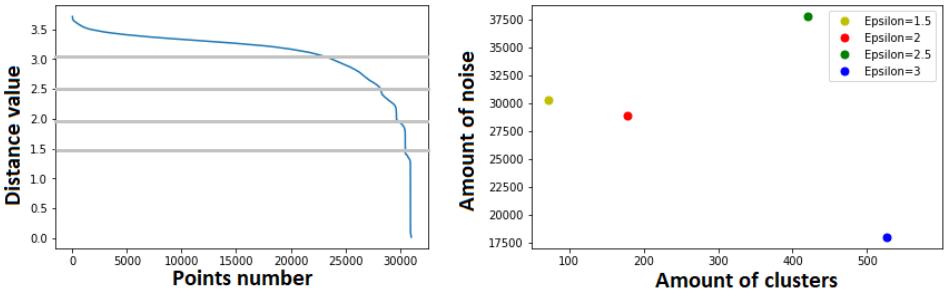


(c) minPts = 12

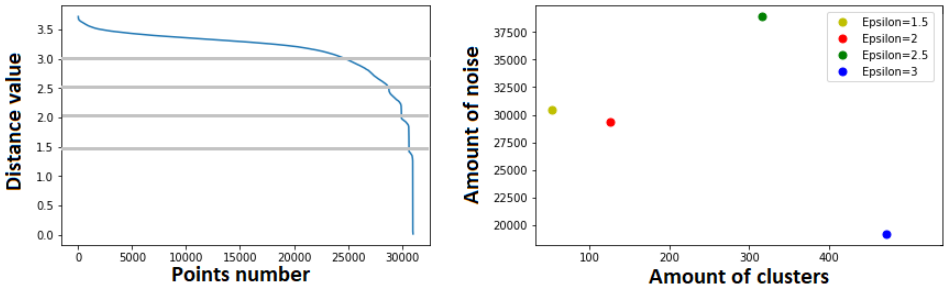


(d) minPts = 13

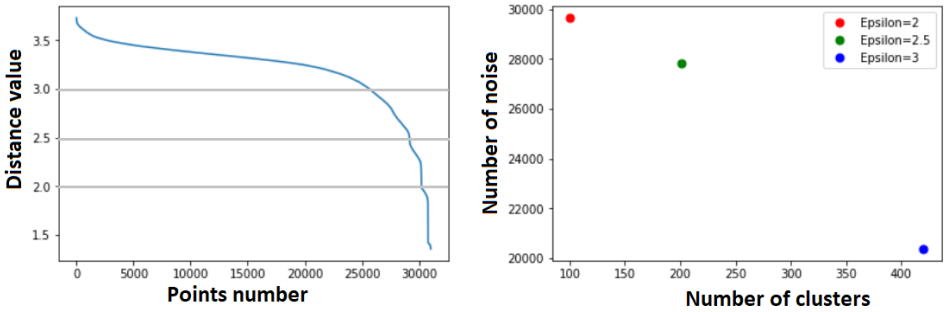
Figure 16. Epsilon and minPts evaluation for English.



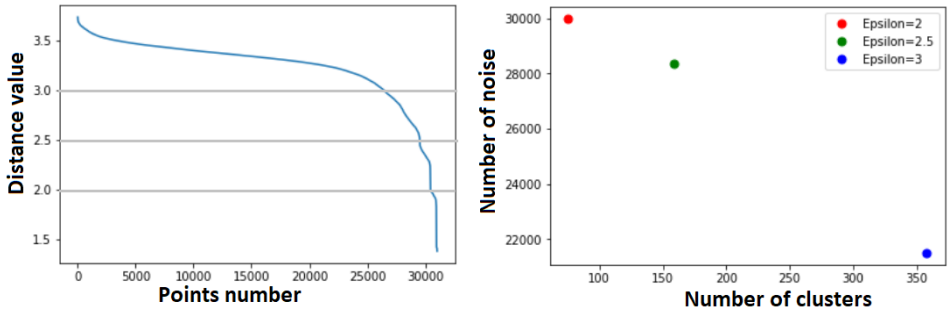
(a) minPts = 9



(b) minPts = 10



(c) minPts = 11



(d) minPts = 12

Figure 17. Epsilon and minPts evaluation for Estonian.

each $minPts$ value, the distance to the k-nearest neighbors was calculated for every element to build the plot. The results for the English data are presented on the left parts in Fig. 16a-16d and those for the Estonian data in 17a-17d.

As can be observed in plots, there exists for each plot several "knee" points that can be used for further evaluation. In most cases, there are three “knee” values, but for a couple of plots in Fig. 17, four were used to cover all cases. Afterward, three to four possible values of the eps were obtained for each of the proposed eps values.

Next, DBSCAN clustering was performed on the appropriate dataset (English or Estonian) for each combination of $minPts$ and eps values, and subsequently, the number of clusters and their noise size was measured. The obtained values were built into the plot separately for each $minPts$, where one dot corresponds to a particular eps value specified in the legend (on the right halves of Fig. 16a-16d for English and of Fig. 17a-17d for Estonian).

As illustrated in plots, in general, larger eps value correspond to a larger number of clusters and a smaller amount of noise. The number of clusters cannot be used as an appropriate evaluation measurement of the algorithms because it depends on the source data. Regarding the amount of noise, it is suspicious when the value is too low and the number of clusters is too large. Most likely, some noise points were grouped into some meaningless clusters. On the other hand, clustering with too large an amount of noise generates poor results and is not overly useful. In the current case, as presented in Fig. 16 and Fig. 17, the amount of noise is large, usually larger than 50%, so it makes sense to choose clusters with the minimum amount of noise.

For both languages, a value of $minPts$ equaling $ln(n)$ and the maximum of the corresponded eps values were chosen:

- for English $minPts = 11$ and $eps = 3$, with 626 clusters and a proportion of noise equal to 68.5%
- for Estonian $minPts = 10$ and $eps = 3$, with 472 clusters and a proportion of noise equal to 61.8%

Through this approach, product clusters for both language datasets were obtained. The products inside each cluster were connected to each other except for the noise points, which do not have any connections.

The part samples of output graphs are presenting in chapter 4.1 Results in Fig. 27a For English and Fig. 28a for Estonian. Respectively, in Fig. 27b-28b examples of data that were labeled by DBSCAN algorithm as a noise are presented.

3.2.5 Keywords extraction using KMeans clustering

The proposed method was inspired by Chauhan’s [Cha18] web article, which was based on Padmakumar and Saran’s [PS16] paper. For the current work, both sources were processed, and the presented approach is an adapted version of the proposed pipelines.

The main idea of this method is to apply word vectorization to product descriptions, further clustering and extracting as keywords words that correspond to the vectors closest to the centers of the obtained clusters. The purpose of this approach is to obtain a particular number of keywords for each product to connect products with at least one keyword in their description.

This method, word vectorization is used, but not in the same way that it was used for the previous approach, where each product description in the entire corpus was presented as one vector and all product descriptions were further clustered such that the elements of the clusters were products from the source data. The product clusters were obtained as output, where the products inside each cluster were connected based on the topic represented by the cluster. In the current approach, clustering occurs inside each product separately, where the elements of clusters are vectors that represent words from the same product description.

Clusters of the words inside each product description were obtained as a result of this clustering, and subsequently, the centers of the obtained clusters were extracted as keywords such that each product description number of the obtained keywords was equal to the number of clusters. For each product, a particular number of keywords was obtained as the output, so products with at least one keyword in common were connected.

The pipeline of this approach can be described as follows:

1. The text was cleaned by deleting numerals and non-letter characters other than punctuation marks.

2. The language detection was carried out, for further deleting language-specific stopwords and tokenization.

3. The sentences were tokenized, further deleting punctuation.

4. The obtained tokens were converted into a vector format.

5. The clusters were applied on the obtained vectors.

6. The centers of the clusters were extracted as output keywords.

Steps 1 to 3 were performed in the previous thesis and are described in chapters 3.2.1 Language detection approach and 3.2.2 Text preparation. In the current subchapter, an overview of the steps 4 to 6 is presented.

For the word vectorization in the current work, it was appropriate to use Word2Vec models from FastText[24]because they were pre-trained on a large number of data and were designed specifically for separate word vectorization as needed. Different models should be used for different languages. FastText is a language-analyzing project by Facebook

[24]https://fasttext.cc/

that contains many pre-trained text analytics model, including word vectorization models for 157 languages, as well as word to vector models for English[25]and Estonian[26]. On the FastText web page, a model in two different formats (binary file and text file) is presented for each language, along with a description on how to use it. For the current work, the text-file versions of the English and Estonian models were used. A 300-dimensional vector was obtained as output for each word in each product description. One property of the vectors is that vectors close in 300-dimensional space represent words with similar meanings.

Next, clustering was performed on the obtained word vectors. The number of clusters should be equal to the number of keywords necessary to extract from the original text. KMeans was used as the clustering method for this approach. KMeans was chosen because it can easily extract the centers of clusters obtained in this way. KMeans is described in chapter 2.1.4 Clustering methods.

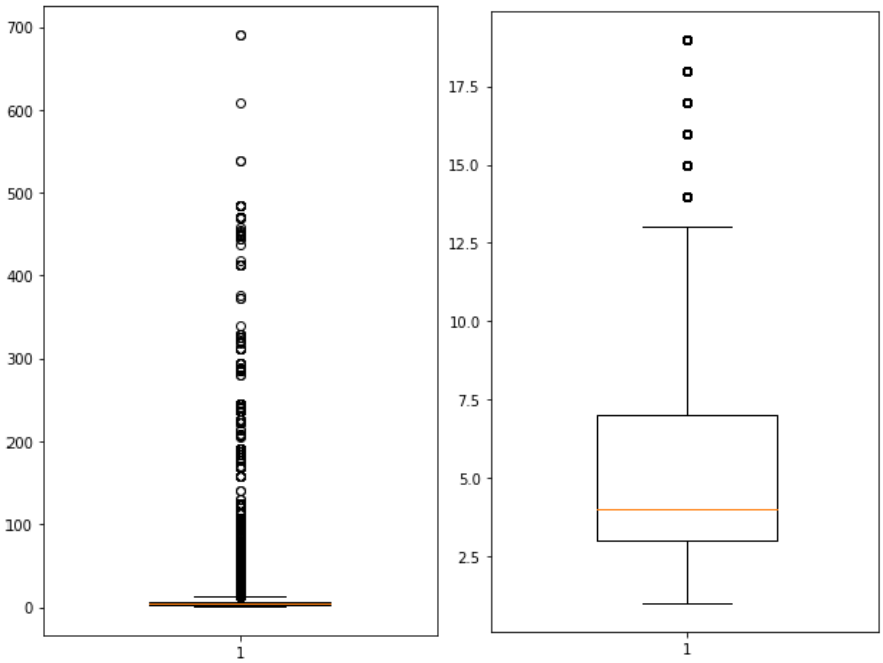
In this work, the implementation of KMeans from the Scikit-learn[27]library was used. In the Scikit-learn library, the centers of clusters can be easily extracted using the model.cluster_centers_ method. However, before the appropriate model can be trained, the number of clusters should be specified. In this pipeline, the number of clusters is equal to the number of keywords that should be extracted from the product description.

To define the optimal number of keywords for each product in the dataset, it is necessary to investigate the distribution of the number of words in product descriptions. For this purpose, two separate box plots[28]were built for the English and the Estonian descriptions using the Mat-plotlib[29]library.

As illustrated in Fig. 18a and 19a, the values in the rectangle correspond to the values between the first and third quartiles, the orange line represents the second quartile, and the dots beyond the upper border are outliers. To clearly indicate the quartiles on the plot, a zoomed-in version of the box plots were built, one for the English (Fig. 18b) and one for the Estonian (Fig. 19b) products. As can be seen from plots, for both languages there are many outliers – descriptions with an extremely large number of words, much more than average.

As an optimal amount of keywords the second quartile (the median) was chosen, because as was presented in Fig. 18 and 19, this value is small enough for both languages. If the number of keywords is smaller than the median, then many descriptions will lose words from their already short descriptions, which is not necessary. More important is to decrease the number of words in the outlier products to extract only the most important words and make their descriptions the same size as the majority of descriptions. In this way, product descriptions with a size smaller than or equal to the median stay the same,

[25]https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.en.300.vec.gz
[26]https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.et.300.vec.gz
[27]https://scikit-learn.org/stable/
[28]https://matplotlib.org/api/_as_gen/matplotlib.pyplot.boxplot.html
[29]https://matplotlib.org/



(a) Original picture. (b) Enlarged picture.

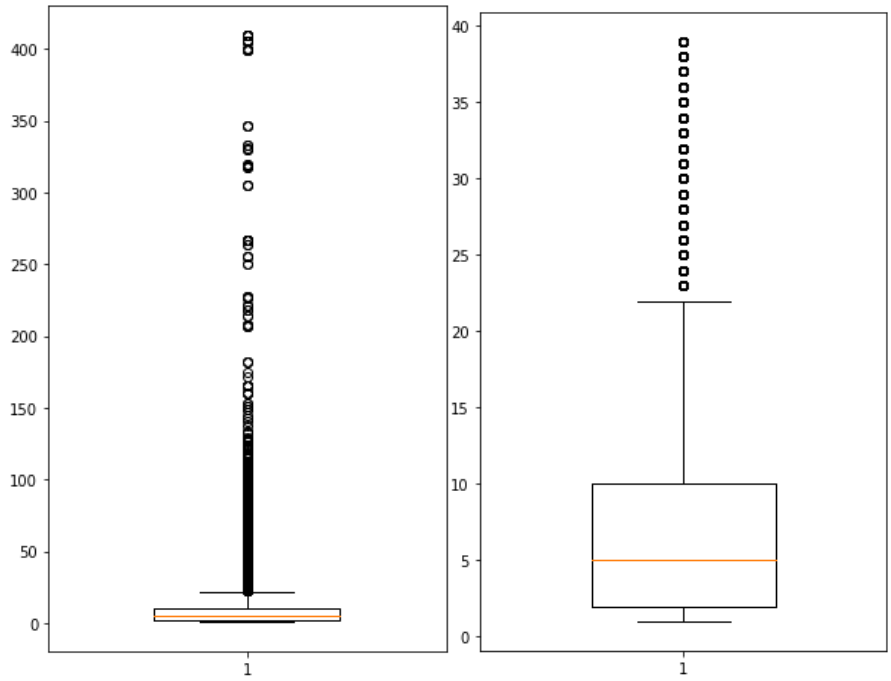
Figure 18. Words number for English products descriptions.

but those bigger than the median are shortened.

To print out the median values for both box plots, the “median” parameter was used. It was found that the median number of words in the English descriptions was four and in the Estonian descriptions five - it is numbers of clusters for KMeans models, and for each language a separate model was built.

After generating clusters for each product description, the words that corresponded to the vectors closest to the centers of the obtained clusters were extracted as keywords. The process was possible because each cluster of word vectors represented a collection of semantically related words whose meaning can be represented by one keyword. The word whose vector is the closest vector to the center of the cluster was chosen as the keyword for each cluster. Keywords matching each cluster formed a separate keyword dataset for each product description.

Through this approach, four keywords were extracted for each English description and five for each Estonian description. Products with at least one keyword in common were connected. The output graph is presented in chapter 4.1 Results in Fig. 25b for English and Fig. 26b for Estonian.



(a) Original picture. (b) Enlarged picture.

Figure 19. Words number for Estonian products descriptions.

3.2.6 Keywords extraction with TF-IDF metric

In this method, the main aim was to extract keywords for each product and then connect the products with at least one keyword in common.

This method was chosen as the appropriate method for the current thesis because, as is shown further in the description of the steps of the method, this approach is similar to one of the basic topic-modeling methods, LSA, which was described in detail in the Background subchapter. The LSA approach builds a text–term matrix with TF-IDF scores as values, as does the proposed method, and afterward applies SVD to decompose it. However, the proposed method does not use the SVD approach or any other decomposition technique; instead, it takes a particular number of words with the highest TF-IDF scores for each text and returns them so they can be further used for graph linking.

The main steps, which should be applied separately for the English and the Estonian descriptions, are as follows, also presented by K. Ganesan [Gan18] post:

1. A dictionary based on the entire corpus of descriptions is created using CountVectorizer, with vocabulary terms counted, where the output is a text–term matrix.

2. The IDF values are computed using TfidfTransformer, with the sparse matrix from

the previous step as input.

3. The TF-IDF values are computed for every description using the pre-trained TfidfTransformer.

4. For each document based on the obtained TF-IDF values, the terms are sorted in descending order and the number of keywords is extracted.

The idea of the current method is to calculate the TF-IDF metric for each word in each product description and then to select the number of keywords for each description.

Keywords are the terms from the original text that have the largest weights and therefore most accurately represent the gist of the text. The typical measurement used to define such weights is TF-IDF, which is described in detail, accompanied by a formula (3), in chapter 2.1 Background. In a nutshell, term frequency-inverse document frequency shows the importance of a word in a document's corpus, where a word has a large weight when it occurs often across one particular text but rarely across the entire set of texts. The methods CountVectorizer and TfidfTransformer from the Scikit-learn package were used to calculate the TF-IDF scores.

To define the optimal number of keywords, it is advisable to explore the size of the current product descriptions and choose the median word size to ensure the same number of keywords for all products. To investigate the sizes of descriptions, two separate box plots were built for the number of words in English (Fig. 18) and Estonian (Fig. 19) product descriptions. This step was described in chapter 3.2.5 Keyword extraction using KMeans clustering. For English, the optimal number of keywords was determined to be four and for Estonian five.

The four most important keywords were obtained for each English description, and for the Estonian descriptions, the five most important keywords were chosen. Afterward, a link was built between each couple of products that share at least one keyword. The output graph is presented in chapter 4.1 Results in Fig. 25a for English and Fig. 26a for Estonian.

4 Validation

In this chapter, the results of using the topic-modeling techniques described in subchapter 3.2 Methods to build graph are presented. The quality of the results and the suitability of the topic-modeling approaches for the current task are also discussed.

4.1 Results

In this subchapter, the results of the topic-modeling approaches used to build graphs are presented. Statistical characteristics and a partial sample of the obtained graphs are provided. Separate graph was built for each topic modeling approach and language combination. In general, ten graphs were obtained for five methods and two languages.

Different methods of graph comparison were proposed in this work, as well as thoughts about the similarity of the graphs obtained in this work to the one proposed by M.-K. Koppel [Kop17] thesis. A final summary of the obtained results and their quality is presented at the end of the chapter.

4.1.1 Graph building

After performing the approaches described in subchapter 3.2 Methods, for both English and Estonian separately, several sets of product clusters were obtained separately. A graph was built for each set of clusters. The package GraphFrames[30]in the Apache Spark environment was used to build the graphs. GraphFrames was chosen because it offers the ability to build DataFrame-based graphs, which is a suitable approach for data saved in Pandas DataFrame[31].

Apache Spark was chosen as an appropriate environment because it allows users to work with a large number of data quickly. The data in the current task are large because almost 322,000 products and 1,300 companies are presented across both languages, which together make up nearly 188,000 nodes for the English data-based graph and nearly 135,000 nodes for the Estonian one. In the beginning, it was unknown how many edges would be made in each graph. The worst-case scenario would be complete graphs, where all vertices are connected to each other. The number of connections in such a graph could be calculated with the combinatorial Formula (4) described in subchapter 2.1.5 Graph Theory. According to Formula (4), for the English data, a complete graph would have 188,000 nodes and 17,671,906,000 links, the Estonian data would have 135,000 nodes and 9,112,432,500 links. To calculate even just one such graph among the several needed and to allow the company to deal with larger data, the efficiency of a big data environment like Apache Spark is necessary.

[30]https://github.com/graphframes/graphframes

[31]https://pandas.pydata.org/pandas-docs/version/0.21/generated/pandas.DataFrame.html

4.1.2 Statistical characteristics of graphs

After the graphs are built, the particular characteristics for comparison should be defined. Nowadays, there is no fixed, automatic method for graph comparison, so such measures and experiments must be designed in a case-by-case basis.

For the current work, the following characteristics were chosen:

- Number of clusters (how many categories [topics, clusters, or keywords] were created with the proposed method). The small number of clusters could correspond to the big portion of clusters with too vague topics. Otherwise, the big amount of clusters could correspond to the big portion of single-node clusters.

- Percentage of companies linked to at least one other company and the percentage of products linked to at least one other product (the more such connections, the better the approach).

- Number of links created among all products. This measure also can be presented as a percentage of the maximum number of possible links described in Formula (4). For English, it is 17,491,139,130 links among 187,036 products, and for Estonian, 9,022,475,946 links among 134,332 products. For this measure, a larger value does not necessarily mean a higher-quality graph because products from different fields cannot be connected to each other. Otherwise, if this number is too small, it is a signal that the method of clustering was not appropriate.

- The smallest number of elements among all clusters. If the approach generates many clusters with only a single product, it is not efficient.

- Average number of products in clusters. This measure illustrates well the real size of clusters, with the minimum and maximum showing border cases. The average number should be large enough to not contain single-product clusters and small enough to avoid noise clusters.

- The biggest number of elements among all clusters. Clusters with a huge number of products could just be clusters full of noise, and a high-quality clustering approach should avoid generating such clusters. This measure, as well as the previous two (minimum and average numbers of products in clusters), can be presented as a percentage of the total number of products.

All these measures are calculated and listed for all methods from chapter 3.2 Methods in table 3 for the English data and table 4 for the Estonian data.

As can be seen from the obtained results, the trends for the English and Estonian data are remarkably similar. The “bigger” and “smaller” relationships for values of different measurements for all methods are almost the same. Accordingly, all conclusions about

Table 3. Comparison of graphs based on different clusters sets for English.

Characteristics	**Methods**				
	Topics, LDA (BOW)	Topics, LDA (TF-IDF)	Keywords, TF-IDF	Keywords, Kmeans	Clusters, DBSCAN
Number of clusters	54	54	9,155	30,536	626
Companies linked to others	100%	100%	100%	100%	100%
Products linked to others	100%	100%	100%	100%	100%
Number of links among products	838,587,236 11.2%	768,522,074 10.6%	227,998,426 3.0%	340,468,854 4.5%	6,697,643 0.1%
Minimal number of products in clusters	1,225 0.7%	1,545 0.8%	1 0.0005%	1 0.0005%	10 0.005%
Average number of products in clusters	3,452 1.8%	3,452 1.8%	58 0.03%	19 0.01%	96 0.05%
Maximal number of products in clusters	33,450 17.9%	30,922 16.5%	14,718 7.9%	13,932 7.4%	1,593 0.9%

Table 4. Comparison of graphs based on different clusters sets for Estonian.

Characteristics	**Methods**				
	Topics, LDA (BOW)	Topics, LDA (TF-IDF)	Keywords, TF-IDF	Keywords, Kmeans	Clusters, DBSCAN
Number of clusters	77	77	9,512	31,089	472
Companies linked to others	100%	100%	100%	100%	100%
Products linked to others	100%	100%	100%	100%	100%
Number of links among products	352,275,700 3.9%	336,674,876 3.7%	30,707,240 0.3%	44,550,814 0.5%	31,359,084 0.4%
Minimal number of products in clusters	647 0.5%	543 0.4%	1 0.0007%	1 0.0007%	12 0.009%
Average number of products in clusters	1,703 1.3%	1,703 1.3%	41 0.03%	13 0.01%	105 0.08%
Maximal number of products in clusters	23,054 17.2%	22,310 16.6%	1,846 1.4%	2,733 2.0%	7,351 5.5%

the obtained measurements described in the next paragraphs are applicable for both languages.

Immediately, it is clear that all methods were able to link each company to at least one other company as well as link each product to at least one other product. Therefore,these metrics can be disregarded when comparing resulting graphs.

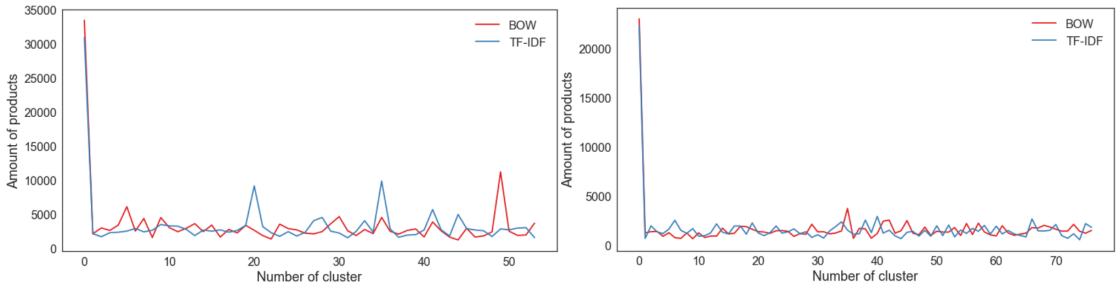
Both LDA based approaches, BOW and TF-IDF, generate an equal number of clusters and an equal average number of products in clusters. The minimum and maximum numbers of products in clusters for those solutions are also very close for both approaches. Because the main LDA pipeline and data were the same in both cases, the BOW and TF-IDF approaches were concerned with only text preparation before using the LDA topic calculation. The number of clusters for the LDA approaches is the smallest among all the methods used. This result is natural, considering that cluster sizes for this method are larger than for other methods. Therefore, for a large number of clusters, their sizes will be small and vice versa. The minimum and average numbers are large enough such that no single-node clusters are generated. Otherwise, the maximum number is quite huge, but the large size could be caused by outliers, so to further explore the real distribution of cluster sizes, a corresponding plot is created (Fig. 20 - 22). Those plots illustrate the number of products in each cluster for different distributions. The ordinal numbers of the clusters are presented on the x-axis of the plot, and the number of products in the corresponding cluster is represented on the y-axis. The number of links among products for the LDA methods is the largest among all approaches, but it is still quite small. To further investigate the quality of the links created, it was decided to perform manual validation of the results, which is provided in subchapter 4.1.4 Comparison of common products between two companies. For this reason, common and not common products between two similar and two different companies were investigated.

The other two methods that showed similar results are clustering based on keyword extraction with the TF-IDF metric and the KMeans algorithm. All their measurements are similar except for the number of clusters. For the KMeans approach, the number of clusters is in almost 3.5 times larger than the number of clusters extracted with the TF-IDF approach. Hence, the set of keywords obtained with KMeans is much more diverse than the set obtained with TF-IDF. These methods present the largest number of clusters among all the methods, which was expected because these approaches extract clusters not as combinations of words but as words themselves, which makes the clusters much more diverse. Otherwise, the number of links among products is quite small (in 40-130 smaller then for LDA approaches), as well as the minimum, average, and maximum numbers of products in clusters, which is suspicious. This must be checked further with plots (Fig. 20 - 22).

The DBSCAN approach stands separately from the others. It provides a smaller number of clusters than the keyword extraction methods but a larger one than the LDA approaches. However, what is important is that one particular cluster is the biggest one – the noise cluster. The proportion of noise for the English data is 68.5% and for Estonian 61.8%. Hence, this approach does not take into account more than half of the

data because labeled it as noise and not using into products connections. Nonetheless, it is still able to connect every company to at least one other company. Because a large number of data is ignored, the number of links among products is smaller than for other approaches, as expected. Regarding the minimum, average, and maximum numbers of products in clusters, the numbers are closer to the analogical values for the keyword extraction methods than to the results of the LDA approaches.

To investigate the size of the clusters obtained by different methods, appropriate plots (Fig. 20-22) were generated.



(a) English.

(b) Estonian.

Figure 20. Number of products in clusters obtained with LDA approaches.

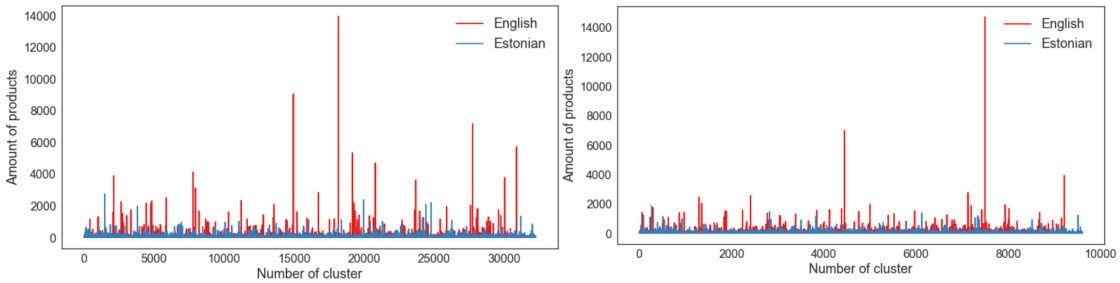
As can be seen in Fig. 20a-20b, the general tendency of the cluster size distribution is similar for both languages with both the BOW and TF-IDF approaches.

The results for the English data are presented in Fig. 20a. In this dataset, topics mostly have sizes below 5,000 products, with a couple of outliers that have close to 10,000 products and one huge outlier with more than 30,000 products. For the Estonian data, shown in Fig. 20b, the situation is similar: most of the topics have sizes below 5,000 products, and there is one huge outlier with more than 20,000 products. For every distribution, the biggest outlier has a cluster number of 0. Therefore, the conclusion could be made that this zero cluster actually is a noise cluster for the LDA method. Typical examples of product descriptions inside the zero cluster include the following:

- For English: "kitchenaid kthcbkitchenaid kthcb", "comfort course views outdoors", "uuuuu".
- For Estonian: "uuuub callista", "nõrutusrest", "kontinentaalvoodi" ("uuuub callista", "slack", "continental bed").

After zero clusters were checked manually, it can be assumed that they collected products with meaningless descriptions, descriptions where words have been merged together or incorrectly written, and short descriptions that do not match other topics.

The distribution of non-zero topic sizes looks natural. The distribution of clusters obtained with keyword extraction is presented in Fig. 21. The results of the KMeans



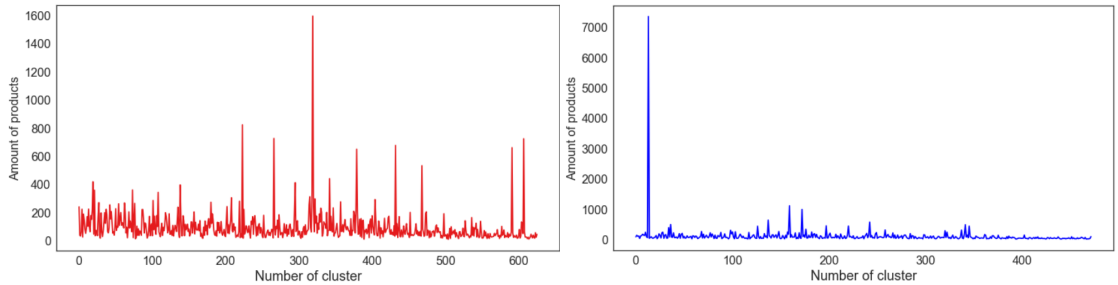
(a) KMeans. (b) TF-IDF.

Figure 21. Number of products in clusters obtained with keywords extraction methods.

method for both languages are illustrated in Fig. 21a and those of the TF-IDF method in Fig. 21b. The English clusters are marked in red and the Estonian ones in blue. Both languages are illustrated on the same plot because the number of clusters is highly similar and the outliers are still evident.

For the Estonian data, the cluster size values obtained with KMeans are mostly below 2,000 products, with no major outliers. The largest cluster has 2,733 products. For English clusters, the situation is similar: the sizes are mostly below 4,000 products, with several larger clusters. The biggest one has 13,932 products, but compared to the clusters around it, the biggest cluster does not seem like an outlier.

For TF-IDF Estonian clusters, the sizes are below 2,000 products, with no outliers. For the English clusters, the main number of clusters is below 3,000 products, with a couple of larger clusters and one outlier of 14,718 products. Inside this outlier, the following products are contained: "door stopper mm valfezn," "mortice lock mm epz," and "panic device pad bolt mm stainless val." Therefore, it does not look like a noise cluster.



(a) English. (b) Estonian.

Figure 22. Number of products in clusters obtained with DBSCAN.

Clusters obtained with DBSCAN are presented in Fig. 22a for English and 22b for Estonian. The English clusters mostly have sizes above 800, with one large cluster of

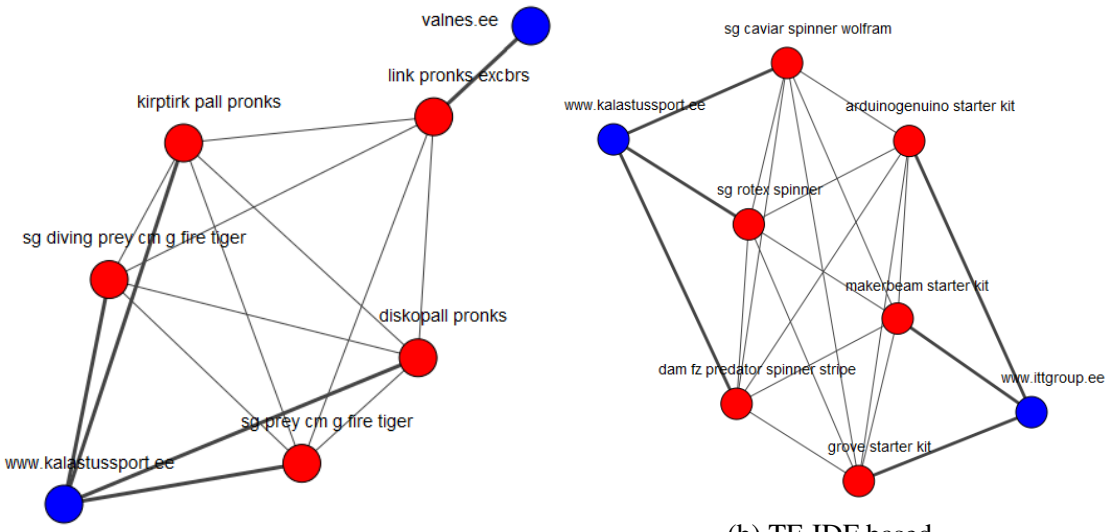
1,593 products. This cluster contains such products as "mm crystal pastel," "mm crystal celsian," and "handwatches michael kors runway crystal." It does not look like a noise cluster.

For the Estonian data, the clusters mostly have a size below 1,000 products, with one outlier of 7,351 products. Examples of products from this cluster include "sõõgilauakomplekt hert sõõgilauda tooli laua," "kreedidi visiidikardi hoidja," and "kauss puuviljadele" ("hertz board table chair," "credit card holder," and "bowl of fruit"). Those products have understandable descriptions but are not connected in meaning. Most likely, this cluster just gathered products that did not fit in other clusters.

4.1.3 Subgraphs samples

The parts of the obtained graphs were visualized to investigate their content. To illustrate the elements of the graphs, the package igraph[32], an analytic tool for different programming languages that allows graph visualizations to be built, was used.

An example of the parts of the graphs built from the clusters obtained with the LDA approaches is illustrated in Fig. 23 for the English data and in Fig. 24 for the Estonian data. One subplot shows the results of the BOW-based approach and the other the TF-IDF-based approach.



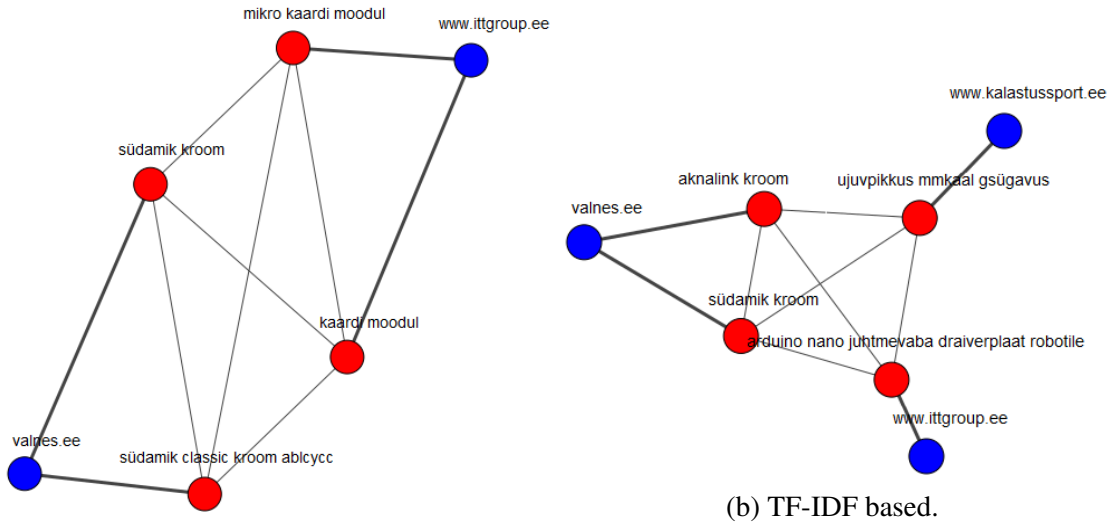
(a) BOW based.

(b) TF-IDF based.

Figure 23. Example of graph elements for English data based on LDA topics.

The visualization in Fig. 23a shows the English clusters based on the BOW approach. In the graph, two companies are connected: valnes.ee (security equipment shop) and

[32]https://igraph.org/python/



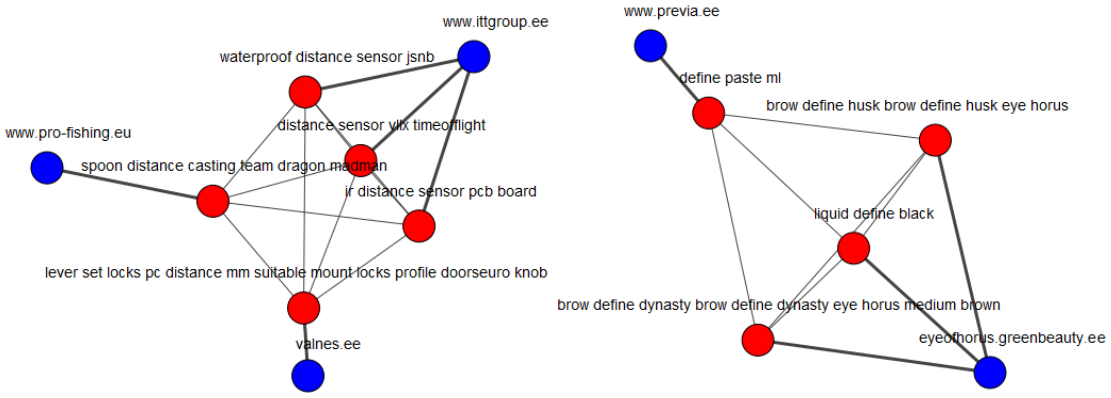
(a) BOW based.

(b) TF-IDF based.

Figure 24. Example of graph elements for Estonian data based on LDA topics.

kalastussport.ee (fishing equipment shop). Their products are different, but they are connected based on similar characteristics, such as "pronks" ("bronze").

The Estonian clusters based on the BOW approach are illustrated in Fig. 24a. The same company - valnes.ee - is now connected with ittgroup.ee (computer equipment shop). Products such as "südamik kroom" ("door lock") and "kaardi moodul" ("card module") connect the two. They are different but are probably connected based on common materials.

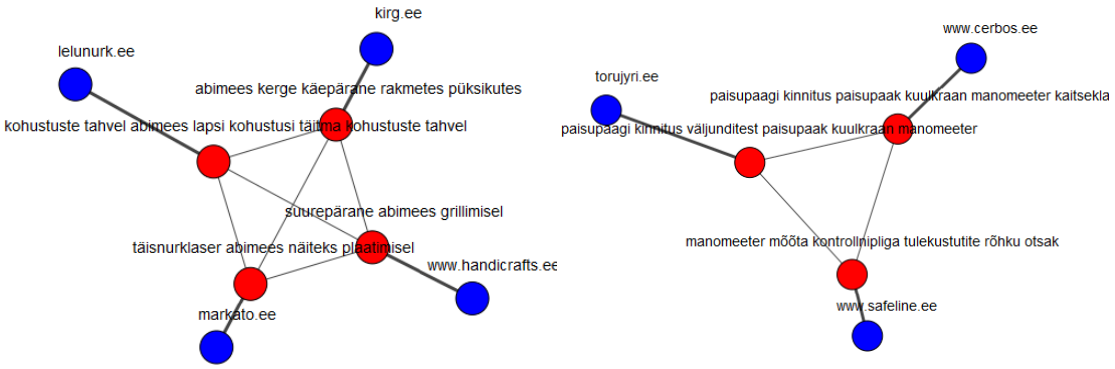


(a) With TF-IDF.

(b) With KMeans.

Figure 25. Example of graph elements for English data based on keywords extraction.

An example of the parts of the graphs built from the clusters obtained with keyword



(a) With TF-IDF.

(b) With KMeans.

Figure 26. Example of graph elements for Estonian data based on keywords extraction.

extraction methods is illustrated in Fig. 25 for the English data and in Fig. 26 for the Estonian data. One subplot shows the results of the KMeans approach and the other those of the TF-IDF approach. The clusters in these graphs have the property that all products in a given cluster have one common keyword.

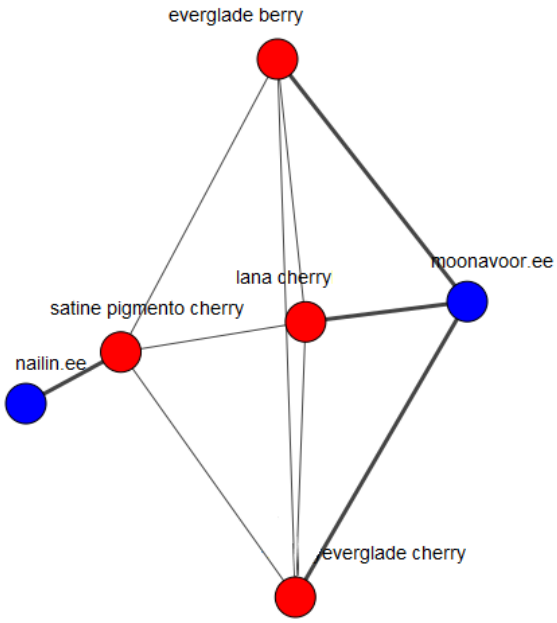
For example, an English graph based on keyword extraction with KMeans is illustrated in Fig. 25b. Two companies are connected: previa.ee and eyeofhorus.greenbeauty.ee (both beauty and cosmetic shops). The first company offers hair paste products and the second liquid eyeliner pens. Both products have the same keyword: "define".

But the same word could be used in different product descriptions. For example, an Estonian graph based on keyword extraction with TF-IDF is illustrated in Fig. 26a. There are four different companies with completely different products: kirg.ee (adult shop), lelunurk.ee (toy shop), markato.ee (repair and construction tool shop), and handicrafts.ee (handmade wooden craft shop). Their products all have one word in common - "abimees" ("helper") - but it is used in a different context each time.

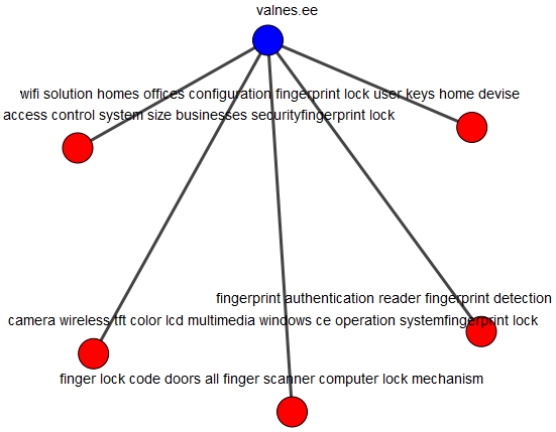
An example of the graphs built based on the clusters obtained with DBSCAN is illustrated in Fig. 27 for the English data and in Fig. 28 for the Estonian data. One subplot shows the results of meaningful clusters and the other an example of non-linked data from noise points.

An example of linked data for English is presented in Fig. 27a. Two companies are linked: moonavoor.ee (wallpaper shop) and nailin.ee (beauty and cosmetic shop). They offer different products, but their products have a common keyword - "cherry" - in the descriptions.

An example of non-linked data for Estonian is presented in Fig. 28b. There is only one company, ittgroup.ee, with four products, all types of cards and receivers. Potentially, they could be connected with some other products.

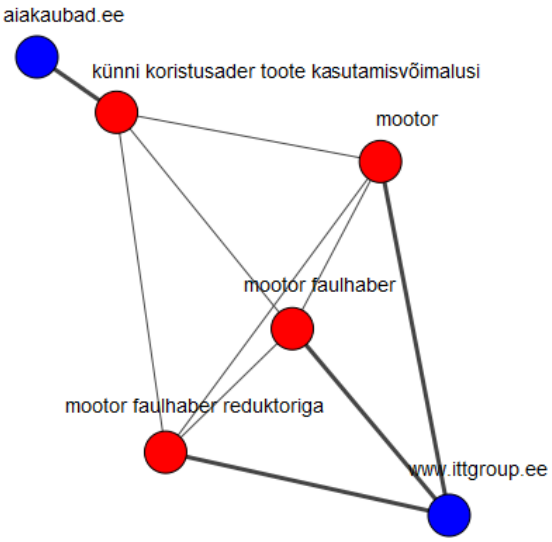


(a) Linked cluster.

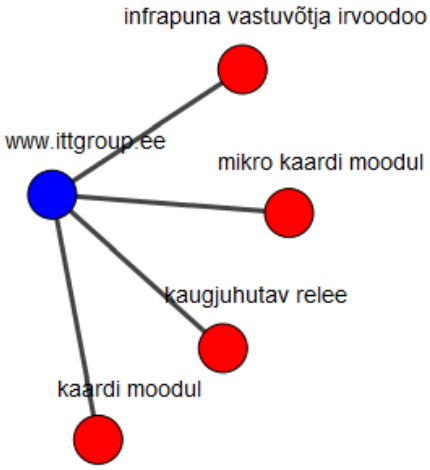


(b) Part of noise.

Figure 27. Example of graph elements for English data based on DBSCAN clusters.



(a) Linked cluster.



(b) Part of noise.

Figure 28. Example of graph elements for Estonian data based on DBSCAN clusters.

4.1.4 Comparison of common products between two companies

One more handmade test was performed to assess how the approaches connect products of companies from the same field and from the different fields.

In the first experiment, two random companies from the one field were chosen: nailin.ee and biotrend.ee, both beauty cosmetic shops. They have products from the same field, and the number of products is similar in both languages. The first company, nailin.ee, has 1,558 products with English descriptions and 612 with Estonian descriptions. For biotrend.ee, there are 1,232 descriptions in English and 114 in Estonian.

For all 10 graphs in both languages, the number of connected products for the two companies was calculated. Companies are considered connected if at least one product from the first company has a link with at least one product from the second company.

Hence, the following metrics were calculated for each graph:

- The number of products connected to both companies among the total number of products from both companies.
- The percentage of products for each company that are connected to at least one product of the other company.

After that, random samples were chosen from the connected and non-connected sets of products. All results are presented in Table 5.

The first column presents the percentage of products from nailin.ee that have at least one connection to a product from biotrend.ee. The second column presents the percentage of products from biotrend.ee that have at least one connection to a product from nailin.ee. The third column presents the percentage of products that are connected for both companies among the total number of their products. The fourth column lists two samples of product descriptions from the common products for both companies. The fifth column displays two samples of product descriptions from unlinked products. The first product is from nailin.ee and the second from biotrend.ee.

As can be seen from Table 5, the LDA approaches found the largest number of common products for both languages. Their percentages and the content of the linked and non-linked products descriptions are highly similar. In contrast, the results for the KMeans-based approach are ambiguous. For the English products, the results are almost the same as the LDA results, but not for the Estonian data. The percentages for the TF-IDF-based approach for the English data are higher than those for the DBSCAN results but lower than the others. For the Estonian data, the TF-IDF-based approach has the lowest numbers. DBSCAN shows low percentages because this method marks more than half of the data as noise, which significantly narrows its circle of possibilities.

Regarding the proposed product descriptions samples, most of the different methods have the same products in the linked part. However, methods with lower percentages missed more products. Some parts of the description samples from non-connected

Table 5. Measurement of connected products from two similar companies.

Methods	**Characteristics**				
	Linked nailin.ee	Linked biotrend.ee	Linked amoung all	Samples linked	Samples non-linked
English					
Topics LDA BOW	76.3%	98.8%	86.2%	"acrylic color powder" "pigment rosa"	"metallic pastel pink" "compact makeup"
Topics LDA TF-IDF	74.0%	96.6%	83.9%	"professional acrylic paint" "rainbow top coat"	"gel brush flat dark" "oil lavender skin"
Keywords Kmeans	72.1%	85.4%	77.9%	"sunny orange" "vitamin bomb"	"smaller bullionsbullions" "glitter multicolor"
Keywords TF-IDF	53.8%	61.3%	57.3%	"hand repair cream" "combination skin cleansing gel"	"beautiful confetti flakes" "sample body"
Clusters DBSCAN	21.8%	50.6%	30.4%	"hand antiseptic" "nail art detail"	"berry juice" "combination skin night cream"
Estonian					
Topics, LDA (BOW)	54.4%	89.5%	59.9%	"pilti värvitoon tipil" "näogeel"	"läikega glitter" "sussid pakis polyethylene"
Topics, LDA (TF-IDF)	34.9%	84.2%	42.7%	"hüdrosool nahale laven" "akrüülpulber"	"nail pintsel" "huulepulk"
Keywords, Kmeans	22.9%	10.1%	20.9%	"puhastaja nahale" "mask nahale"	"tooniga glitter" "silmakreem"
Keywords, TF-IDF	0.4%	14.9%	1.6%	"kontsentraat nahale" "sussid pakis"	"värvitooniga geel" "päevakreem"
Clusters, DBSCAN	8.5%	37.5%	10.4%	"pärlid multicolor" "kunstküünteleua-tavavalguses geellakk"	"hologramm-teemandid" "näogeel pinguldav"

products contain noise or not enough words. Logically, it seems that such products are not connected and could probably be allocated to the noise clusters.

The proposed companies are from the same field (cosmetics), so it is expected that they will have many products in common. Furthermore, they offer a similar number of products, but based on the results, some conclusions about their differences can be made. It seems that biotrend.ee mostly has products linked to nailin.ee, whereas nailin.ee has more links to other companies.

In the second experiment, two random companies from the different fields were chosen: smarta.ee (electronic devices accessories shop) and all4pet.ee (pet supplies). They have a similar number of products in both languages. The first company, smarta.ee, has 240 products with English descriptions and 5 with Estonian descriptions. For all4pet.ee, there are 272 descriptions in English and 4 in Estonian. Numbers of products with Estonian descriptions are too small for sufficient scores calculating, so they were skipped during the experiment. Besides, as illustrated in experiment one for two similar companies, methods results for English are analogical as for Estonian.

Set up of experiment and metrics are the same as for the first company pair. Results are presented in Table 6, which has the same structure as previous Table 5.

Table 6. Measurement of connected products from two different companies.

Methods	**Characteristics**				
	Linked smarta.ee	Linked all4pet.ee	Linked amoung all	Samples linked	Samples non-linked
English					
Topics LDA BOW	17.5%	13.6%	15.4%	"otterbox commuter series case"	"fashion case iphone aurora red" "royal canin kitten"
Topics LDA TF-IDF	15.8%	94.5%	57.7%	"acana dog pacifica"	"magnet wallet iphone plus black" "orijen dog puppy"
Keywords Kmeans	11.3%	72.5%	43.9%	"royal canin jelly"	"startpakett basic" "orijen dog senior"
Keywords TF-IDF	2.9%	46.1%	25.9%	"hillubs feline adult ocean fish"	"magnet wallet iphone" "brit care cat missy"
Clusters DBSCAN	16.6%	6.3%	9.5%	"royal canin mini exigent"	"start kit monitor" "large breed lamb rice"

As can be seen from Table 6, the graph based on the DBSCAN algorithm provides a minimal number of common products. But it doesn't mean that this is the best approach because DBSCAN labeled too many products as a noise. It used only 17.5% of products from smarta.ee and 34.9% from all4pet.ee. Otherwise, LDA based on BOW approach

showed the second result and used the whole data from both companies. In contrast, LDA based on TF-IDF approach showed the worst results. Probably, this solution creates a lot of links between the similar and between the different companies equally. Two other approaches based on keywords extraction also showed a big amount of linked products, likely, because different products could have the same keywords, but with different meanings.

Regarding linked and not linked products, some conclusions could be made. If take into account only solutions with a small number of connected products, some interesting examples could be found. For example, product "otterbox commuter series case" from smarta.ee has the word "otter" inside, that easily could be connected with pet topic by LDA model.

Overall, the LDA-based approaches showed the best results regarding both the obtained percentages and the manual checking of connected and non-connected products.

4.1.5 Summary

At present, no one universal approach for graph comparison exists. In this thesis, several criteria for graph evaluation have been proposed.

Based on the statistical characteristics, for both languages, the results of the LDA approaches were better than the results of the other methods. The LDA methods created the largest number of links between products but not so many that it is suspicious. Furthermore, according to the distribution of cluster size, the LDA clusters have only one exceptionally large zero cluster (most likely noise), whereas no other clusters have outliers that are too large or too small. In addition, this method did not create any clusters with a single product, as some of the others did.

The examples of subgraphs illustrated in Fig. 23-28 cannot provide any conclusive results regarding the graph comparison, but they illustrate that any method can generate correct and incorrect links among products, even the LDA approaches.

The investigation of the common products of two similar and two different companies also illustrated that LDA is the most efficient approach. LDA approaches found many common products from similar companies, as was expected from companies of the same field. For companies from different fields, BOW based version of LDA was much better, than TF-IDF based. Some thought about theoretical differences between those methods are presented below.

Regarding the three other topic-modeling approaches, some conclusions can be drawn. Keyword extraction based on KMeans clustering and TF-IDF scores did not generate very good results because the clusters are based on particular keywords. The same word in different descriptions could have a different meaning, which makes the quality of product connections quite low. Also, DBSCAN failed to present high-quality results because, even with the best input parameters, it labeled more than half of the data as noise. In this way, it discarded many useful products, which also made the quality of

connections quite low.

In the method comparison task, which ultimately suggested that LDA is the most appropriate approach, some uncertainty still exists - whether the BOW-based or TF-IDF-based approach is the best and what is the difference. Based only on the numbers, BOW showed better results. Therefore, some theoretical reasoning can be conducted.

First, the TF-IDF approach is usually used to reduce the influence of stopwords on the obtained topics. Stopwords are mainly considered words that appear frequently but do not provide any value to a topic, such as articles. However, in the current thesis, before topic modeling techniques were applied, text cleaning was performed. In the process, only nouns were selected for further examination, which means that most stopwords were automatically filtered out. Hence, the TF-IDF approach was not as effective as it could have been. Second, based on the theoretical background of LDA, described in chapter 2.1.2 Topic modeling, this method works with plain integers in the BOW approach. In contrast, TF-IDF provides scores as floats but can still be used because the Gensim implementation of LDA smooths this process and allows both integers and floats to be used.

According to all listed results and thoughts, LDA based on the bag-of-words approach was chosen as the most appropriate topic-modeling method for the current work.

4.2 Discussion

The graphs obtained during the current work show different the statistical characteristics and the quality of connections among products. The results were mostly based on how the approach creates clusters and how many data it defines as noise. According to the conclusions from chapter 4.1 Results, LDA based on the BOW approach was chosen as the most appropriate topic-modeling method.

The graphs obtained in this work cannot be directly compared with the graphs obtained in Koppel [Kop17] thesis. First, the particular universal way of graphs comparison doesn't exist. Second, the graphs are based on different approaches. In the graphs created in this work, every product of every company is a unique item. If products from different companies are connected as common products, they are still different products and should simply be similar in nature, based on the link. The companies connected via those products have three edges between them. In the graph created by Koppel [Kop17], one SKU corresponds to one product. Hence, if products have the same SKU code, then they are connected and actually represent the same product. Therefore, companies connected through a product share one common node and have two edges between them.

For these reasons, it is difficult to make a direct comparison of the graphs. The only way to compare them is to rely on the numbers and the quality of connections.

Regarding the graph presented by Koppel [Kop17], the number of created links is described. Because only products with SKUs were explored, the author discovered that only 13,699 products were connected to the same companies. This number corresponds to 1.9% of all products from the source data. More important was that among those 13,699 products, the number of products linked to the more than one company was lower than 300, which is only 2.2% of products with SKUs.

In the current work, the graphs based on the best LDA approach created 11.2% the possible links and connected 100% of products for the English data as well as 3.9% of the possible links and 100% of products for the Estonian data. Hence, the number of links obviously increased, but their significance changed.

The goal of products linking was to model relationships between companies and use those relationships to generate additional features. In Koppel [Kop17] case, the relationships were "companies sell the same product." In the case, presented in the current work, relationships correspond to "companies sell similar products". At the same time, it was clear from Koppel [Kop17] work that it is not possible to use collected data for the main goal without improving the number of links. So, this work solves this problem, but the results need further study in the context of the company use case, which will be very domain specific tasks.

According to the obtained results, topic modeling could be used as a method of entity linking, and it could improve the quality of the graphs presented in the work of Koppel [Kop17].

5 Conclusion

The current work was performed with the purpose of improving the entity linking in the pipeline presented by Koppel [Kop17] thesis. The pipeline collected linked data from the web and transformed it into features for a machine-learning model for credit-scoring. For one of the steps of the proposed pipeline, the products obtained from the web data were linked, but the results were poor. As a basis for the linking, the author used stock keeping units (SKUs), but only 1.9% of the products even had one. Hence, only around 0.02% of products were connected to two or three companies.

In the current thesis, textual information about the products was chosen as the source data for the linking. Product descriptions were separated based on topic using different topic-modeling techniques. Products considered to be similar were clustered together.

The following topic modeling techniques were evaluated to separate the products into clusters:

- The most popular approach nowadays, the latent Dirichlet allocation (LDA) approach, in two forms: one based on a corpus embedded with a bag-of-words model, and one with a TF-IDF approach, which provides a set of topics for the proposed text corpus.
- The DBSCAN algorithm, where product descriptions are transformed into vectors using the Doc2Vec approach.
- Keyword extraction with a KMeans clustering algorithm, where for each product, the words in the product description are presented as vectors using the Word2Vec technique, the keywords from each description are extracted, and products with intersecting sets of keywords are connected as similar.
- Keyword extraction with TF-IDF scores, where for each product description, the words with the largest TF-IDF scores are extracted as keywords and products with intersecting sets of keywords are connected.

After performing topic modeling techniques, for each of them, a set of topics (clusters) was generated. Products in the one cluster consider to be similar ones and they could be connected. Hence, graphs were built based on the obtained clusters. The Apache Spark environment and packages were used to build the graphs. There is no one single way to compare graphs, so the graph evaluation approaches were defined manually. For each graph, the statistical characteristics were calculated, random samples were investigated, and tests for the intersection of two particular companies were conducted. According to the obtained results, the best topic-modeling approach is the BOW-based LDA approach. It created 11.2% of all possible links among all products as well as connected each product with at least one other product and every company with at least one other company.

The obtained results significantly increased the number of links from Koppel [Kop17] paper. However, it is important to note that the graph from the previous work connected only products that are exactly the same. The graph in the current work connects products that may be different but that belong to the same category of products. Regarding the use cases of company linking considered in the current work, the obtained graph could provide more features for a machine-learning model for credit-scoring and improve its quality.

The main contribution of the current thesis was investigation whether topic modeling approach can be applied to solve the problem with previous work [Kop17] and evaluation which topic modeling approach gave the best result. The current work provided a graph that can be used to generate more features. However, whether the additional links are usable for the original use case of the company needs more investigation and evaluation by the domain experts.

In future research, the quality of LDA topic modeling could be improved. For example, a future study could manually check the obtained topics to verify whether they are significant and whether or not their meaning overlaps. Overlapped topics could be connected together.

Some other approaches for topic modeling or keyword extraction could also be explored to improve the proposed methods. Furthermore, more advanced methods of graph comparison could also be proposed, such as checking variations in graph similarity and subgraph matching algorithms or manually checking larger numbers of connections in the graphs.

References

[BK07] Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[Cha18] Kushal Chauhan. Unsupervised Text Summarization using Sentence Embeddings, 2018. `https://medium.com/jatana/unsupervised-text-summarization-using-sentence-embeddings-adb15ce83db1`, Last visited on May 2019.

[CWL+18] Hui Chen, Baogang Wei, Yonghuai Liu, Yiming Li, Jifang Yu, and Wenhao Zhu. Bilinear joint learning of word and entity embeddings for entity linking. *Neurocomputing*, 294:12–18, 2018.

[EKS+96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[Emm17] Chris Emmery. Euclidean vs. Cosine Distance, 2017. `{https://cmry.github.io/notes/euclidean-v-cosine},LastvisitedonMay2019.`, urldate = 2019-16-04.

[Gan18] Kavita Ganesan. Keyword Extraction with TF-IDF and Python's Scikit-Learn – Full Working Example, 2018. `http://kavita-ganesan.com/extracting-keywords-from-text-tfidf/#.XLOpu5gzY2y`, Last visited on May 2019.

[GPL+06] Rayid Ghani, Katharina Probst, Yan Liu, Marko Krema, and Andrew Fano. Text mining for product attribute extraction. *ACM SIGKDD Explorations Newsletter*, 8(1):41–48, 2006.

[HRN+13] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R Curran. Evaluating entity linking with wikipedia. *Artificial intelligence*, 194:130–150, 2013.

[HSZ11] Xianpei Han, Le Sun, and Jun Zhao. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM, 2011.

[KGAF11] Anitha Kannan, Inmar E Givoni, Rakesh Agrawal, and Ariel Fuxman. Matching unstructured product offers to structured product specifications. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 404–412. ACM, 2011.

[Kop17] Madis-Karli Koppel. Large Scale Feature Extraction from Linked Web Data. Master's thesis, University of Tartu, 2017. https://www.semanticscholar.org/paper/Large-Scale-Feature-Extraction-from-Linked-Web-Data-Koppel/65c8dffe8001b0f4a48e377e8de86fe175a1f8bb.

[Kum18] Kamal Kumar. Evaluation of Topic Modeling: Topic Coherence, 2018. https://datascienceplus.com/evaluation-of-topic-modeling-topic-coherence, Last visited on May 2019.

[Li18] Susan Li. Topic Modeling and Latent Dirichlet Allocation (LDA) in Python, 2018. https://towardsdatascience.com/topic-modeling-and-latent-dirichlet-allocation-in-python-9bf156893c24, Last visited on May 2019.

[MCCD15] Tomas Mikolov, Kai Chen, Gregory S Corrado, and Jeffrey A Dean. Computing numeric representations of words in a high-dimensional space, May 19 2015. US Patent 9,037,464.

[Mis18] Deepak Mishra. DOC2VEC gensim tutorial, 2018. https://medium.com/@mishra.thedeepak/doc2vec-simple-implementation-example-df2afbbfbad5, Last visited on May 2019.

[MMK03] David JC MacKay and David JC Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[PS16] Aishwarya Padmakumar and Akanksha Saran. Unsupervised Text Summarization Using Sentence Embeddings. 2016.

[RBH15] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408. ACM, 2015.

[RP16] Petar Ristoski and Heiko Paulheim. Rdf2vec: Rdf graph embeddings for data mining. In *International Semantic Web Conference*, pages 498–514. Springer, 2016.

[RS10] Radim Rehurek and Petr Sojka. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer, 2010.

[vBBR+15] Ronald van Bezu, Sjoerd Borst, Rick Rijkse, Jim Verhagen, Damir Vandic, and Flavius Frasincar. Multi-component similarity method for web product

duplicate detection. In *Proceedings of the 30th annual ACM symposium on applied computing*, pages 761–768. ACM, 2015.

[VP14] Konstantin Vorontsov and Anna Potapenko. Tutorial on probabilistic topic modeling: Additive regularization for stochastic matrix factorization. In *International Conference on Analysis of Images, Social Networks and Texts*, pages 29–46. Springer, 2014.

[Xu18] Joyce Xu. Topic Modeling with LSA, PLSA, LDA & lda2Vec, 2018. `https://medium.com/nanonets/topic-modeling-with-lsa-psla-lda-and-lda2vec-555ff65b0b05`, Last visited on May 2019.

Appendix

I. Glossary

Bag of Words is a method of text vectorization that represents text as a bag or set of words, ignoring grammar and word order but taking into consideration multiplicity. 15

Coherence is a score that illustrates how strongly connected the obtained topics are and whether they overlap. 35

Collective Entity Linking is a type of Entity Linking, where links between entities are built based on interdependence between them. 20

Complete graph is a type of the graph, where each pair of nodes is connected. 18

Cosine metrics is a measurement that illustrates the similarity between two particular elements in the vector space, in the way of measuring cosine of the angle between those two elements. 39

Density-based spatial clustering of applications with noise is a density-based clustering approach that links together the cluster elements located close to each other (in other words, the ones that have many neighbors). 16

Dirichlet distribution is a set of continuous multivariate probability distributions, that can be described as "the one distribution above the rest," and it answers the question of whether real probability distributions can be observed with this particular type of distribution. 14

Distance matrix is a square matrix wherein the rows and columns present the same set of entities (in the current case, a set of products), and the values are the distance between the element in the row and the element in the column. 39

Document to Vector is a technique that works like Word2Vec, but it is processes not each word separately but rather sets of words, so the obtained vector represents an entire text. 16

Entity Linking is the method which the main goal is to define the correct meaning of the entities mentioned in a particular text. 8, 9, 12

Euclidean distance is measurement that represents the forward distance between two particular data elements within Euclidean space. 39

KMeans is a clustering method, based on the division of elements into a particular number of clusters, where each element lies in the cluster with the closest average value, represented as a cluster prototype. 17, 45

Latent Dirichlet Allocation is a topic modeling approach that presents a Bayesian extension of pLSA and uses the Dirichlet priors for both distributions (text–topic and topic–term). 14, 30, 35

Latent Semantic Analysis is a topic modeling method, which main idea is to take a matrix of texts and terms and decompose into split text-topic and topic-term matrices. 13, 47

Linked data is a method of publishing structured data such that it can be interconnected and more useful through semantic queries, but instead of using it only to serve web pages for human readers, it is also used to exchange information in a way can be read automatically on computers. 8, 11, 20

N-quad is a type of N-Triple with an optional "context" component in the fourth position. 11

Named Entity Recognition is a task of text analysis that attempts to find and classify proper names mentioned in the unstructured data. 9

Natural language processing is a field of computer science related to the interaction between computer and natural human languages. 8

Open Data is information that can be openly used and copied by anybody. 8

Part-Of-Speech tagging is a technique that tags every word in the text with the appropriate part of speech. 30, 34

Probabilistic latent semantic analysis is a topic modeling approach that represents an improvement of LSA that uses a probabilistic method instead of the SVD approach. 13

Resource Description Framework is a group of specifications that was created as a metadata data model, used as a common approach for data representation and structuring in the web. 11, 19, 73

Singular Value Decomposition is a method that factorizes a matrix into the product of the three different matrices. 13, 47

Stock Keeping Unit a separate item type for sale, presented in current data as set of numbers. 9, 19

Term Frequency-Inverse Document Frequency is a numerical statistic designed to show the importance of words in a document's corpus. 13, 15, 48

Topic Modeling is an approach to building a model that analyzes a collection of text documents and determines which topics each document contains. 9, 12

Triple is a set of three components — "subject", "predicate", and "object" — that encodes a statement about the semantic data. 11

Word to Vector is a technique that represents words as vectors in a multidimensional space with one important characteristic: words with common meanings are represented with vectors in close proximity to one another. 15, 45

II. Acronyms

ARTM Additive Regularization of Topic Models. 22

BJLM bilinear joint learning model. 20

BOW bag of words. 15, 36

CEL Collective Entity Linking. 20

CV coherence verification. 35

DBSCAN density-based spatial clustering of applications with noise. 40

Doc2Vec document to vector. 16

ESTNLTK Estonian Natural Language Toolkit. 34

IT information technology. 8

LDA latent Dirichlet allocation. 13, 14, 30, 34

LSA latent semantic analysis. 13, 47

NER named entity recognition. 9

NLP Natural language processing. 8

NLTK Natural Language Toolkit. 34

pLSA Probabilistic LSA. 13

POS part-of-speech. 30, 34

RDF Resource Description Framework. 11, 18

SKUs stock-keeping units. 9, 19

SVD Singular value decomposition. 13, 47

TF-IDF term frequency-inverse document frequency. 13, 15, 36, 48

Word2Vec Word to vector. 15, 44

III. Code

Source code for the current thesis is located by the following GitHub repository:
https://github.com/dil-delada/master_thesis

IV. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Olha Kaminska**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,
Entity linking via topic models in Apache Spark,
supervised by Pelle Jakovits and Peep Küngas.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Olha Kaminska
16.05.2019