Clustering of lithuanian news articles

Vilius Pranckaitis, Mantas Lukoševičius
Faculty of Informatics
Kaunas University of Technology
Kaunas, Lithuania
e-mail: vilius.pranckaitis@ktu.edu, mantas.lukosevicius@ktu.lt

***Abstract*—There is arguably more research done on clustering of English texts than of any other language. In this article, the process of clustering Lithuanian news articles is studied. For text preprocessing, the effect of stemming, term frequency metrics and feature filtering is investigated. In addition, following clustering algorithms are compared: k–means, bisecting k–means, and three linkage method variations of hierarchical clustering. The results show that k–means algorithm gives best overall results and that only one of the three hierarchical algorithms produces comparably good results. Term frequency–inverse document frequency (TF–IDF) with stemming significantly increased clustering quality compared to not doing stemming and/or using TF. Feature filtering by IDF helped to optimize the k–means algorithm, but reduced the quality when using hierarchical clustering.**

Keywords—document clustering; feature selection; k-means; hierarchical clustering; Lithuanian news articles

I. Introduction

The way people transfer information has changed drastically throughout the history. The initial methods were slow and had low capacity, e.g. a messenger carrying as many scrolls as he is physically able to carry. Throughout the centuries new ways to store and transfer information were invented. Nowadays information travels at the speed of light in amounts so large that humans can hardly comprehend them.

While the amounts of information transferred increased, human capabilities to consume this information did not improve as much. This only increases the need for automated means to process information: group, categorize, find duplicates, etc. One of such means is document clustering.

This work studies clustering process of Lithuanian news articles. The purpose of this study is to examine how different text preprocessing steps and clustering algorithms affect the quality of the clustering produced. This article describes the work done and the results of it. Similar previous work includes [1, 2, 3].

This document is divided into several sections. Section II introduces reader to feature selection and clustering of textual data. Section III describes the data set used in this study. Section IV lists the metrics used to evaluate results. In sections V and VI it is described how features were selected and what experiments were performed. Sections VII and VIII provide analysis of results and conclusions.

II. A Brief Introduction to Document Clustering

Document clustering is an application of cluster analysis to textual documents. It is unsupervised learning and can be used for finding similar documents, organizing large document collections, detecting duplicate content and search optimization [4].

Document clustering process can be separated into three stages: feature selection, feature extraction and an application of clustering algorithm. A more detailed description of these stages is given in the following paragraphs.

A. Feature selection

Feature selection is a process of creating feature vectors from the text of the documents. It includes various text processing steps such as splitting text into tokens, stemming and lemmatization, removing stop words, and calculating term frequency values.

Text data has a few properties, which requires different approach than clustering other kinds of data [5]. While the dimensionality of text data is large, documents usually contain a relatively small number of distinct words (i.e. the data is sparse). In addition, the number of words in two documents may differ by orders of magnitude (e.g. a tweet versus a chapter of a book). Also, some of the words might be common in all kinds of texts while others would only appear in specific type of documents. Measures for these problems can and should be taken in feature selection step.

Stemming and lemmatization can be used to reduce the problem of high dimensionality. Multiple forms of the same word induce multiple dimensions in a feature vector. By stemming words multiple dimensions would be joined to a single dimension corresponding to a base form of the word. In addition to decreasing dimensionality this would also reduce noise caused by the grammar of a language (which is especially true for Lithuanian language, in which a word can have many different forms).

There are multiple ways to control the input of a term to a clustering process. A simple stop words list can be used to remove stop words from the text. Other statistical techniques can be used for less significant term removal, such as *term strength*, *entropy-based ranking* and *term contribution* [5]. A common approach to control how much a term affects the clustering process is *term frequency–inverse document frequency* (TF–IDF). A numeric value is assigned to every term as a product of TF and IDF values. TF is calculated per

document and is directly proportional to the number of times a term in a document. IDF is calculated per corpus and is inversely proportional to number of documents in which a term appears.

There are multiple variants of TF–IDF formula. In this work fallowing formula is used:

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t,D) = f_{t,d} \cdot \log\frac{N}{n_t}, \qquad (1)$$

where $f_{t,d}$ is the number of times term *t* appears in document *d*, *N* is the number of documents in *D*, and n_t is the number of documents in *D* containing term *t*.

B. Feature extraction

While feature selection is a process of filtering relevant features of a text, feature extraction uses the original feature set to build new features. This includes methods for dimensionality reduction, such as *Principal Component Analysis*, *Non-negative Matrix Factorization* and *Latent Semantic Indexing* [5]. The clustering itself can also be used to extract features. For example, in [6] it was shown how to reduce dimensionality and noise of the features by clustering the words first.

The importance of feature extraction comes from a fact that words tend to correlate with one another. This means that the feature space is bigger than the number of concepts [5]. Dimensionality reduction can be used to transform this big feature space to a smaller space of concepts.

C. Clustering

There are many different approaches how to cluster textual data [5]: distance–based clustering, probabilistic methods (e.g. *topic modeling*), co-clustering (clustering words and documents simultaneously), clustering with frequent phrases. In this work variations two distance–based algorithms are studied: *k–means clustering* and *hierarchical agglomerative clustering*.

K–means is a simple algorithm, producing flat clustering. During initialization, algorithm selects *k* means, which corresponds to *k* clusters. Then two steps are repeated: (1) for every data point choose the nearest mean and assign the point to the corresponding cluster; (2) recalculate means by averaging data points assigned to the corresponding cluster. The algorithm terminates, when assignment of the data points doesn't change after successive iterations.

The clustering produced by k–means algorithm highly depends on the initial means chosen. A common approach is to run algorithm multiple times with randomly chosen initial means. In [7] there is proposed a simple randomized seeding technique, which improves speed and accuracy of k–means.

Bisecting k-means algorithm is a variation of k–means algorithm. This algorithm takes top–down approach. It starts with single cluster containing all the data points. Iteratively the biggest cluster is chosen and split into two parts by running inner k–means on the data points of the cluster. The algorithm terminates when the chosen number of clusters is reached.

Hierarchical agglomerative clustering is another well-known technique for clustering. Algorithm starts by assigning every data point to a separate cluster. Then iteratively two closest together clusters are joined into a single cluster. This step is repeated until single cluster is left. The output of the algorithm is a tree which describes how the clusters were joined.

There are multiple ways to describe the distance between two clusters in terms of distances between data points in those clusters:

- **Single linkage**. The distance between two clusters is equal to smallest distance between two data point crossing these clusters (i.e. closest pair of points across clusters). This resembles Kruskal's *minimum spanning tree* algorithm.
- **Group average linkage** (also known as *Unweighted Pair Group Method with Arithmetic Mean*, UPGMA). In this method, the distance between clusters is an average of distances between every pair of data points across clusters.
- **Complete linkage**. This is similar to single linkage method, but instead of smallest distance the largest distance between a pair of data points is taken (i.e. the most distant pair of points across clusters).

The distance–based clustering algorithms requires a metric to evaluate the distance between data points. In this work a cosine distance was chosen:

$$\mathrm{CosineDistance}(\vec{x},\vec{y}) = 1 - \frac{1}{|\vec{x}| \cdot |\vec{y}|}\sum_{i=1}^{n} x_i \cdot y_i, \qquad (2)$$

where $\vec{x}$, $\vec{y}$ are feature vectors, *n* is dimensionality of the vectors. Cosine distance is a commonly used distance metric and gives good results for textual data [8, 9]. In [1] it was shown that cosine distance works significantly better than Euclidean distance when clustering Lithuanian texts using k–means algorithm.

III. Data Set Preparation

The articles for the experiments were taken from the three major Lithuanian news websites: *delfi.lt*, *alfa.lt* and *15min.lt*. A week's worth of articles were retrieved, starting from January 1st to January 7th, 2016, total of 3572 articles.

The news sites contain different sub-sections, e.g. "news from Lithuania", "crime", "business", etc. Every article is published in one of these sub-sections. To label the data the names of the sub-sections were used. One problem of such labeling is that different websites have different number of categories, some of which are more abstract than others (e.g. one website has a single sub-section "sports", while other has multiple sub-sections for "basketball", "football" and so on). To avoid labels being mismatched, they were normalized across the websites by grouping them to several categories. By investigating the websites, a common set of categories was noticed:

- Lithuania news (413 articles);

- World news (426 articles);
- Crime (321 articles);
- Business (337 articles);
- Cars (170 articles);
- Sports (602 articles);
- Technologies (129 articles);
- Opinions (99 articles);
- Entertainment (526 articles);
- Life (306 articles);
- Culture (56 articles);
- Other (187 articles, which doesn't fall into previous categories).

This division to categories shows that some categories are more popular than others. For example, 1/6th of the articles fall into "sports" category. The *standard deviation* of such composition is 173.6, and *median absolute deviation* is 135 (these metrics are also used for comparison of clustering algorithms further in this article).

Fig. 1 shows the article categories mapped to 2 dimensions using PCA. While the "sports" category lies in top right quadrant clearly separated from other categories, other categories are much more intermixed. This is probably caused by the fact that sports articles have a distinctive vocabulary and that "sports" category takes up a big part of the data set.

IV. Means of Evaluation

To measure the quality of the clustering several evaluation metrics were used: *precision*, *recall*, *F1 score*, *purity* and *entropy*.

A. Precission and recall

Precision and recall are well known measures in information retrieval. Both measures deal with notions of *true positive*, *false positive*, *true negative* and *false negative*. In case of clustering, every pair of documents are taken in account. If a pair of documents have same label and appear in the same cluster, then it is a true positive. The rest of the notions are described in Table I.

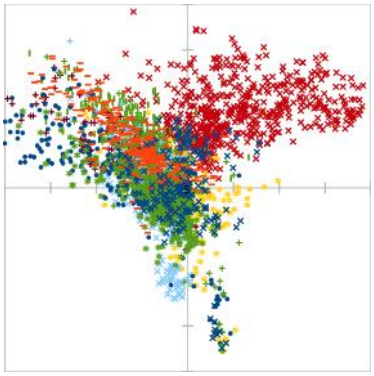

Fig. 1. Article categories, mapped to 2 dimensions using PCA. In the top right quadrant clearly separated from other categories lies "sports" category.

TABLE I. Confusion matrix for a pair of documents

	Same cluster	**Different clusters**
Matching labels	True positive	False negative
Different labels	False positive	True negative

Precision and recall range from 0 to 1 and are defined as:

$$\text{Precision} = \frac{tp}{tp + fp}; \tag{3}$$

$$\text{Recall} = \frac{tp}{tp + fn}. \tag{4}$$

B. F1 score

Precision and recall tend to introduce bias when the number of clusters reaches extremes (e.g. recall equals 1 when every item falls into a single cluster). F1 score reduces bias by combining both measures:

$$\text{F1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{5}$$

Because F1 score doesn't tend to favor high or low number of clusters as much as other metrics, a preference is given to this metric throughout the article.

C. Purity

Purity describes the homogeneity of clusters. Purity of a clustering ranges from 0 to 1 and is defined as

$$\text{Purity} = \frac{1}{N}\sum_{i=1}^{k} \max_j |c_i \cap g_j|, \tag{6}$$

where N is the number of items, k is the number of clusters, c_i is a chosen cluster, and g_j is a category which has the maximum number of items in c_i.

Purity has a bias for high number of clusters. For example, when every cluster consists of a single item, purity equals to 1.

D. Entropy

Entropy, same as purity, describes the homogeneity of the clusters. However, while purity only considers the number of items from dominating category, entropy considers the whole composition of the cluster. For a single cluster, entropy is defined as

$$\text{Entropy}(c_i) = -\sum_{j=1}^{m} \frac{|c_i \cap g_j|}{|c_i|} \cdot \log_m \frac{|c_i \cap g_j|}{|c_i|}, \tag{7}$$

where c_i is a chosen cluster, m is the number of categories, and g_j is a category which has items in cluster c_i. To calculate then entropy for a whole clustering, weighted average is used:

$$\text{Entropy}(C) = \frac{1}{N}\sum_{i=1}^{k} |c_i| \cdot \text{Entropy}(c_i), \tag{8}$$

where C is a set of clusters c_i, k is the number of clusters, and N is the number of items.

Entropy, same as purity, is biased towards high number of clusters (e.g. entropy of a single item cluster is 0).

V. Feature Selection Procedure

Text preprocessing and conversion to feature vectors consisted of the following steps:

1) Splitting text into tokens;
2) Switching characters to lowercase;
3) Stemming (*skipped in some experiments*);
4) Filtering a specific percentage of terms which are rarest according to IDF (*skipped in some experiments*);
5) Applying TF–IDF (*TF used in some experiments*);
6) Normalizing feature vectors.

More details about what feature selection steps were used in which experiments are given in the experiment descriptions.

VI. Experiments

Several experiments were conducted during this work, testing text preprocessing steps and clustering algorithms. Below is a detailed description of these experiments.

A. Stemming and term frequency experiment

In this experiment, it was studied how stemming and different term frequencies affected the clustering. 4 different configurations were used: (1) TF without stemming, (2) TF with stemming, (3) TF–IDF without stemming, and (4) TF–IDF with stemming. No term filtering was applied.

For clustering, k-means algorithm was used. Each stemming/term frequency configuration was tested by making multiple runs, trying out different numbers of clusters $k \in [9;13]$ and with 5 randomly generated initial mean sets (in total, 25 runs per stemming/term frequency configuration).

B. Clustering algorithms experiment

In this experiment, multiple clustering algorithms were tested. In every test the same feature selection procedure was used. The text preprocessing matched the steps mentioned in section "Feature Selection Procedure", except without any term filtering applied.

Below is the list of clustering algorithms tested:

- K-means;
- Bisecting k-means;
- Hierarchical, single linkage;
- Hierarchical, group average linkage;
- Hierarchical, complete linkage;
- Random (*for comparison*).

In case of k-means algorithms, previously mentioned configuration was used: 5 randomly generated initial mean set for each $k \in [9;13]$. The results of multiple tests were averaged.

Contrary to k-means, hierarchical clustering doesn't depend on a random factor. However, to extract flat clustering HDBSCAN extraction method [10] was used, which doesn't allow direct control of the number of clusters. In HDBSCAN, a single parameter is used—a minimum size of a cluster. This allows to imprecisely control the number of clusters produced. Because of this, multiple tests were run with different minimum cluster size parameter. The tests, which produces from 9 to 13 clusters, were picked and their results were averaged.

C. Term filtering experiment

In term filtering experiment, it was studied how clustering quality changes in response to increasing the number of terms removed from feature vectors. Filtering was done by sorting terms by IDF and removing the ones which appear in the least number of documents. The percentage of terms remaining after filtering ranged from 100% to 10%. K-means and group average linkage hierarchical algorithms were used, configured as in clustering algorithms experiment.

VII. Results and Analysis

A. Stemming and term frequency experiment

The average F1 scores of different stemming and term frequency combinations are displayed in Fig. 2. The worst results are produced by TF ($\overline{F1} = 0.19$ without stemming and $\overline{F1} = 0.21$ with stemming). While TF–IDF without stemming provides only a small increase ($\overline{F1} = 0.25$), the F1 score improves by more than 50% when applying stemming ($\overline{F1} = 0.39$).

The results confirm the intuition that TF–IDF and stemming improves clustering by disregarding stop words and reducing noise caused by language grammar. It is worth noting that stemming was much more effective when combined with TF–IDF. This shows that when selecting features, one misconfigured step can greatly diminish effect of other steps.

B. Clustering algorithms experiment

In Fig. 3 there are displayed average and maximum values of F1 score. The best results (both average and maximum) were shown by k–means and bisecting k–means algorithms (latter achieving the absolute maximum $F1_{max} = 0.44$ across all algorithms with no term filtering applied). Close results were shown by group average linkage hierarchical algorithm. Complete linkage and single linkage versions performed significantly worse, by value being closer to random clustering than to previously mentioned algorithms.

More quality measures are given in Table II. Different

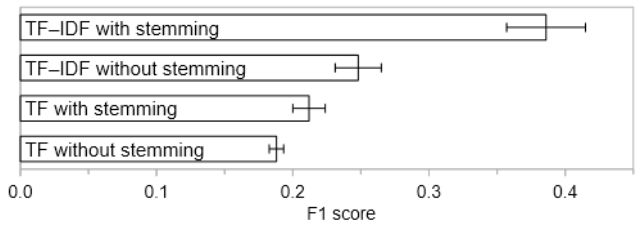

Fig. 2. F1 scores of a clusterings using different combinations of TF/TF–IDF and stemming. TF–IDF produce better results than TF with and without stemming. While stemming has small impact when combined with TF, there is a more than 50% increase in F1 when combining stemming with TF–IDF.

TABLE II. EVALUATION METRICS OF DIFFERENT CLUSTERING ALGORITHMS

Clustering algorithm	Quality measures (averaged)					Cluster size statistics (averaged)	
	F1 score	*Precision*	*Recall*	*Purity*	*Entropy*	*Standard deviation*	*Median absolute deviation (MAD)*
K–means	**0.386**	0.327	0.482	**0.522**	0.580	286.5	194.9
Bisecting k–means	0.375	0.293	0.532	0.479	0.617	356.7	150.9
Hierarchical, group average linkage	0.354	**0.330**	0.389	0.495	**0.578**	221.4	79.8
Hierarchical, complete linkage	0.194	0.152	0.277	0.291	0.806	355.1	71.0
Hierarchical, single linkage	0.187	0.111	**0.604**	0.287	0.811	853.0	14.0
Random	0.100	0.109	0.093	0.173	0.927	16.8	13.2

algorithms lead according to different metrics, but two of them stands out: k–means algorithm has the highest mean value of F1 score and purity, while group average linkage hierarchical clustering leads by precision and entropy.

Among all the algorithms, single linkage hierarchical clustering has the best recall value. This is probably caused by the fact that recall is biased to favor big clusters. In many cases this algorithm produced single cluster containing more than a half of articles along with multiple small clusters. The cluster size statistics confirms that—single linkage hierarchical clustering has the highest standard deviation, but lowest median absolute deviation (MAD). High standard deviation is caused by cluster sizes being distant from the mean. In case of MAD, a small number of outliers are irrelevant, so low MAD value means that the sizes of most of the clusters are close to each other.

When comparing cluster size statistics, group average hierarchical algorithm produces clusters which are close to each other in size. This observation is backed up by relatively low values of standard deviation and median absolute deviation. The cluster size distribution is not as equal in case of both k–means algorithms. While the k–means has lower standard deviation, bisecting k–means has lower MAD. In other words, the majority of clusters produced by bisecting k–means are more similar in size than those produced by k–means, but the outliers are more extreme too.

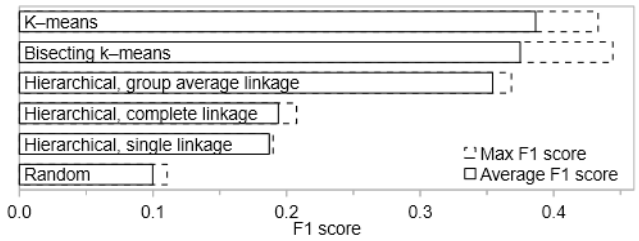

Fig. 3. Average and maximum F1 score of different clustering algorithms. The best results were shown by k-means and bisecting k-means algorithms with group average hierarchical clustering coming in a close third.

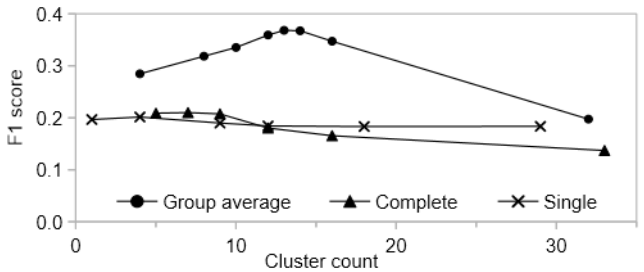

Fig. 4. Relation between F1 score and number of clusters produced by hierarchical clustering algorithms. When the number of clusters gets close to number of categories, the F1 score for group average hierarchical clustering tend to increase.

Fig. 4 displays the relation between number of clusters and the F1 score of hierarchical clustering algorithms. Group average clustering tends to show higher values when the number of clusters is close to the number of categories. This signals that the underlying concepts found by the algorithm are similar to the categories of the documents. In case of single and complete linkage methods, there is no such correlation.

C. *Term filtering experiment*

For filtering tests, k-means and group average linkage hierarchical algorithms were chosen. There were multiple runs starting with full feature space of 41 thousand dimensions to only 10% of terms which are most common across the documents. The F1 scores are displayed in Fig. 5.

Of the two algorithms, k–means shows less sensitivity to changes of feature space. While there is a small decrease in F1 score moving from 100% to 50% of the dimensionality (minimum at $\overline{F1}_{0.5} = 0.371$), the F1 score of 10% rises slightly above the full feature space with $\overline{F1}_{0.1} = 0.391$ versus $\overline{F1}_{1.0} = 0.386$.

Similar results were shown in [11]. This study compared different metrics for feature selection, including document frequency. The clustering was done on English texts using k–means algorithm. The variation of quality metrics moving from 100% to 10% was relatively small. A more rapid change occurred going from 10% down to 2%. Reference [2] tested multiple feature selection metrics on Lithuanian and Russian texts, and had best results at 7% of features remaining.

The results of group average linkage hierarchical clustering differ from the k-means ones. The variance of F1 score is significantly greater. At the beginning the score increases from $\overline{F1}_{1.0} = 0.36$ to $\overline{F1}_{0.9} = 0.39$. Later values drop reaching $\overline{F1}_{0.4} = 0.326$ and after a temporal increase eventually drops to $\overline{F1}_{0.1} = 0.321$.

The pruning of 10% of features might have reduced the noise in the feature space. This would explain the sudden rise of the F1 score. The further pruning might have started removing relevant features. However, this does not explain the increase in F1 score at 30–20% of feature space.

TABLE III. RELEVANT TERMS (IN LITHUANIAN) OF THE CLUSTERING WITH THE HIGHEST F1 SCORE

Cluster size	143	113	907	484	420	308	449	121	232	395
Relevant words	variklio dyzelinių gamintojų automobilių elektromobilių tesla motors fiat volkswagen lg	šalčio temperatūra laipsnių kritulių rajoniniai sniego provėžoti hidrometeorologijos plikledis prispausto	aktorė dainininkė muzikinę prodiuseris eurovizijos koncertinį meninės žanro scenoje režisierius	rungtynių ekipa žaidėjų turnyro taškų pergalę kamuolį treneris rezultatyvaus įvartį	karinių partijos sąjungininkų nimro šiitų obama narystės referendumas sirijos nato	policijos ugniagesiai vpk neblaivus ambulatoriškai prom patrulių komisariato girtumas vairuojamas	bendrovės barelį indeksas brent įmonės valiutos wti holding akcininkų aplinkosaugos	dakaro ruožas ralio ekipažas vanagas juknevičius lenktynininkai trasos lenktynių benediktas	jums organizmą vitaminų astrologė cukraus horoskopas riebalų jus mitybos ožiaragis	šulinį saviečių tragedijos smurto mažamečius smurtaudamas sugyventinę sumetė nužudė ekspertizė

When comparing both clustering algorithms, k-means seems to intrinsically diminish the value of rare terms. In case of group average hierarchical clustering, the impact of such terms to the clustering results is significantly higher.

D. A drill–down look at the clustering results

During this study, several hundred runs were made of various clustering configurations. From the top 50 tests sorted by the F1 score, every single one was done using one of the two k–means algorithms, only several of them being bisecting k–means. Most of these tests came from the filtering experiment, over a half having 40% or less features remaining.

In order to have a better idea if the methods actually produce meaningful clusters, a metric was proposed to measure which terms are more relevant to the cluster. It is a combination of two inverse document frequencies and is defined as

$$\text{Relevance}(t, c, D) = \frac{\text{idf}(t, D)}{1 + \text{idf}(t, c)}, \quad (9)$$

where t is a term, c is a set of documents in a cluster, and D is the documents in the data set. In other words, term relevance is high if the term is (1) relatively rare across the documents in the data set and (2) relatively common in the documents of the cluster.

The test with the highest F1 score came from the filtering experiment: k–means algorithm and 20% of remaining features. The most relevant terms of this test are displayed in Table III. As expected, the clusters don't seem to identically map to the categories in the data set. However, the themes of the clusters are easily noticeable (in Table III, some of the themes are "cars", "sports", "Dakar rally", "business", "foreign politics" and "crime"). Despite that, a deeper look at the clusters reveals that it is not uncommon for other themes to appear in the clusters, sometimes not very related to the relevant words.

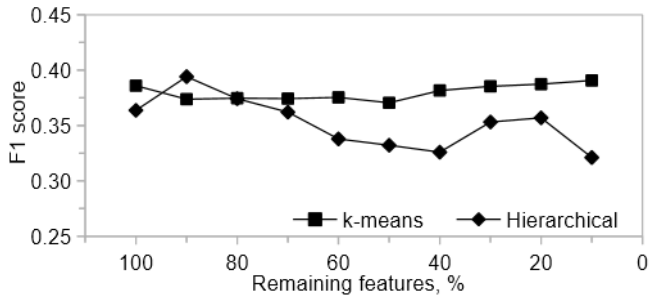

Fig. 5. F1 score dependency on the percentage of features used. K–means algorithm was less sensitive to the change of feature space dimensionality. F1 score of hierarchical clustering varies significantly more.

VIII. CONCLUSIONS

In this work, several different experiments were performed studying feature selection process and clustering algorithms. Term frequency and stemming experiment proved that TF–IDF with stemming is superior to other configurations. In addition, this experiment showed that one misconfigured step in the process can greatly diminish the effects of other steps. In the clustering algorithms experiment only 3 of 5 algorithms tested produced acceptable results, k–means performing arguably the best overall. In case of filtering experiment, k–means algorithm seemed mostly unaffected by removal of major part of the features, while the results of group average linkage hierarchical clustering varied more and was less predictable.

REFERENCES

[1] G. Ciganaitė, A. Mackutė-Varoneckienė, T. Krilavičius, "Text documents clustering," in *19-oji tarpuniversitetinė tarptautinė magistrantų ir doktorantų konferencija IVUS*, 2014.

[2] A. Mackutė-Varoneckienė, T. Krilavičius, "Empirical Study on Unsupervised Feature Selection for Document Clustering," in *Human Language Technologies – The Baltic Perspective*, pp. 107-109, 2014.

[3] E. Kebelytė, M. Lukoševičius, "Keywords analysis of Lithuanian news stream," in *20-toji tarptautinė magistrantų ir doktorantų konferencija "Informacinės technologijos 2015"*, Kaunas, 2015.

[4] N. Shah, S. Mahajan, "Document clustering: a detailed review," *International Journal of Applied Information Systems*, vol. 4(5), pp. 30-38, 2012.

[5] C. C. Aggarwal, C. Zhai, "A survey of text clustering algorithms," in *Mining Text Data*, New York, Springer-Verlag, pp. 77-128, 2012.

[6] N. Slonim, N. Tishby, "Document clustering using word clusters via the information bottleneck method," in *Proceedings of the 23rd annual international ACM SIGIR conference*, 2000.

[7] D. Arthur, S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007.

[8] H. Anna, "Similarity measures for text document clustering," in *Proceedings of the Sixth New Zealand*, 2008.

[9] A. Strehl, J. Ghosh, R. Mooney, "Impact of similarity measures on web-page clustering," in *In Workshop on Artificial Intelligence for Web Search (AAAI 2000)*, 2000.

[10] R. J. G. B. Campello, D. Moulavi, J. Sander, "Density-based clustering based on hierarchical density estimates," *Advances in Knowledge Discovery and Data Mining*, pp. 160-172, 2013.

[11] T. Liu, S. Liu, Z. Chen, W.-Y. Ma, "An evaluation on feature selection for text clustering," *Icml*, pp. 488-495, 2003.