CMST 24(1) 43–58 (2018) DOI:10.12921/cmst.2018.0000007

Open Stylometric System *WebSty*: Integrated Language Processing, Analysis and Visualisation

Maciej Piasecki[1*], **Tomasz Walkowiak**[2], **Maciej Eder**[3]

[1] *Faculty of Computer Science and Management*
Wrocław University of Science and Technology

[2] *Faculty of Electronics*
Wrocław University of Science and Technology

[3] *Institute of Polish Language*
Polish Academy of Sciences and Pedagogical University of Kraków

** E-mail: maciej.piasecki@pwr.wroc.pl*

Received: 14 April 2017; revised: 28 December 2017; accepted: 16 January 2018; published online: 31 March 2018

Abstract: The paper presents an open, web-based system for stylometric analysis named *WebSty*, which is a part of the CLARIN-PL research infrastructure. WebSty does not require local installation by users, can be used via any web browser, offers rich set-up, and runs on a computing cluster. We discuss the underlying ideas of the system, its architecture, a pipeline of language tools for processing Polish, and its integration with systems for clustering, visualizing the results of clustering, and identifying the features of the strongest discrimination power. The techniques used for feature weighting and text similarity measuring are also concisely overviewed. In conclusions, we present preliminary evaluation of WebSty on the corpus of 1000 literary works, and we report on the results of the first research applications of WebSty. Even if the system was initially focused on processing Polish texts, we also briefly discuss its development towards a multilingual system, which already supports English, German and Hungarian.

Key words: stylometry, language technology infrastructure, web application, authorship attribution, style analysis, CLARIN

I. INTRODUCTION

Most literary works are now digitised and available for research. Various tools for natural language analysis and the quality of their performance are growing at a considerable pace. It opens possibilities for more advanced and deeper application of the distant reading paradigm to text collections, e.g. investigating different types of associations between authors, styles, genres, the context of their creation, translation, etc. However, at the same time there exist several obstacles that make the potentially attractive combination of opportunities difficult:

1. a variety of technical solutions for data formats and communication make combination of language tools and data analysis tools non-trivial; especially those focused on stylometry are difficult to be operationalised,
2. application of language tools and data analysis tools usually requires specialised knowledge and technical skills from the users, which might be challenging for researchers in the area of the Humanities and the Social Sciences,
3. the entire workflow involving data analysis settings, language tools properties and error characteristics expresses a large number of hyperparameters whose in-

fluence on the overall results of the stylometric analysis is difficult to control and thus might lead to unpredictable outcomes of the experiment.

WebSty is an open, stylometric system with a web-based user interface that was proposed as a step towards overcoming the aforementioned obstacles.

WebSty initial prototype was focused on processing texts in Polish, using a selection of the most robust language tools for Polish. The goal of the work presented here was to develop an architecture of the WebSty system that would allow for a significant flexibility when it comes to the availability of different language tools combined with numerous settings of processing parameters applicable for different types of stylometric tasks. Moreover, several different visualisation methods were added to the system, so that clustering results can be supplemented by attractive visual plots.

II. STYLOMETRIC TASKS

Stylometry is usually associated with an analysis of language features extracted from a collection of texts, aimed at tracing similarities and dissimilarities between these texts. It is usually used to identify groups of texts that exhibit subtle similarities hidden to the naked eye but traceable by multidimensional statistical techniques. A classical type of analysis where the stylometric methods prove useful is authorship attribution, or an experimental setup where an anonymous (or disputed) text is compared against a set of texts of known authorship, in order to identify the nearest neighbourship relations between them [1–3]. A typical attribution scenario involves a corpus of a few "candidate" authors and some controls, but one might also face a much more complex setup, which is referred to as the open-set attribution problem, where the list of "candidate" authors cannot be assumed fixed.

Not only is stylometry attractive for its ability to solve attribution problems. In the Humanities, as well as the Social Sciences, text analysis is becoming an interesting methodological proposition to assess textual similarities beyond authorship. In the study of literature, one might be interested in distant reading techniques to pinpoint genre characteristics, authors' gender signal, literary period, intertextuality, and so forth. In sociology, one might want to analyse textual biases in press materials from different press agencies, in psychology one might trace change of style as a function of age, or correlations between textual properties and mental diseases [4].

Unlike traditional stylistics, however, stylometry is focused on language features that might seem "unattractive": these include articles, particles, pronouns, conjunctions, prepositions and similar function words. In English, these are the words "the", "and", "of", "in" etc., in Polish – "się", "w", "i", "za", etc. It is very difficult to observe these words' variance by a close reading, but it is surprisingly straightforward to extract them and count them using computational techniques. Simple as they are, however, function words have also their limitations, because they represent but a fraction of linguistic features that might be useful in stylometry. Recent studies in the field suggest that authorship attribution can be improved using frequencies of part-of-speech tags (grammatical classes), letter tri-grams, parse trees fragments, and similar features. Stylometry beyond attribution will certainly benefit from the usage of named entity recognition features, chronology identifiers, topic modelling characteristics, and many other sophisticated features. However, they cannot be simply harvested from input texts using, say, regular expressions. To have an insight into these sophisticated language features, one has to involve third-party natural language engineering software. What makes the WebSty approach attractive is that our system offers a variety of dedicated language processing tools that can be concatenated into the workflow.

III. RELATED WORK

Research on stylometry has a very long and rich tradition. Hundreds of papers have been published. However, when we look for systems supporting stylometric analysis, and especially systems that are publicly available, the situation is very different. Only a few systems can be found.

Signature (2017) [5] is an online application designed for author identification, so based on similar philosophy to WebSty. However, *Signature* works on the level of words only, and offers a limited range of methods for comparing texts.

StyleTool [6] is a quite rudimentary off-line, application for calculating word frequency finding clusters of documents and visualising them with the help of Principle Component Analysis (PCA).

AICBT (2017) is a web application described as an "online authorship attribution tool". It uses simple lexical features. Its description is very imprecise, but AICBT seems to work on the basis of an average number of words per sentence, lengths of sentences, punctuation frequencies, common words, and PoS automatically assigned to text with the help of NLTK[1] system [7]. Processing is based on k-means clustering from *scikit-learn*[2] library [8].

JGAAP (2017) [9] is an off-line application that offers a whole line of processing: automated conversion from different document formats to text, preprocessing techniques, e.g. removing numbers, simple NLP processing (e.g. coarse grained PoS) and dictionary-based, e.g. calculating frequency of function words. The prepared vectors for documents can be next used in machine learning packages.

[1] NLTK – Natural Language Toolkit URL: http://www.nltk.org
[2] scikit-learn – Machine Learning in Python http://scikit-learn.org/stable/

JStylo [10] is an off-line application which was developed as a part of research on Adversarial Stylometry. Several configurations of the features are discussed and some list are quite rich, e.g. Unique words, Complexity, Sentence Count, Average Sentence Length, Average Syllable Count, Character Count, Letter Count, Gunning-Fog Readability Index, Flesch Reading Ease Score. Vectors generated do document are next processed with the *Weka 3*[3] machine learning system [11] for data mining, supervised classification. However, JStylo is focused on one particular goal of recognition of obfuscation of authorship attribution.

Voyant 2.0 [12] is on-line tool for limited statistical analysis of texts. It was originally focused on counting word level statistics. Voyant has now rich functionality and a good User Interface with plenty of visualisation methods. A range of NLP tools was added on the a basis of the Stanford CoreNLP[4] [13], e.g. PN recognition. However most functionality of Voyant is based on tracing word forms and their relative frequencies across text. Only simple statistical measures: tf.idf and Z-score are available to compute them for word forms vs documents. Voyant is almost exclusively focused on English. It can be applied to other languages, but only on the level of word forms, e.g. without lemmatization.

Mallet[5] [14] is an off-line document classification system working on the basis of machine learning. It employs basic language tools for tagging and named entity recognition, but it is quite narrowly focused on topic analysis.

LATtice (2017) [15] is an off-line application providing several visualisation methods for showing distances between documents, where distances are calculated as the Euclidean distances between document vectors.

Stylo [16] is a library of functions in the R programming language (some of them supplemented with GUI) for different stylometric tasks. The package is designed to analyse shallow language features (function words and letter n-grams) harvested from locally stored plain text files, but it can also be used to analyse corpora preprocessed by third-party tools, such as taggers, parsers, topic modelling software, etc. The package offers both exploratory methods (PCA, MDA, hierarchical cluster analysis), and supervised machine-learning algorithms (nearest shrunken centroids, support vector machines, k-nearest neighbors, Burrows' Delta).

None of the systems is designed to work with large amounts of data and parallel processing. They offer only selected language tools and processing methods. All systems do not provide methods for the extraction of characteristic features or the available methods are rather simple, e.g. frequency-based. It is hard to find an online system offering as complex and comprehensive functionality as that provided by WebSty. It is hard to compare the system beyond the comparison of features and functions, as there is lack of benchmark data sets for stylometric tasks in Digital Humanities.

IV. LANGUAGE PROCESSING ARCHITECTURE

Natural Language Processing and Machine Learning tools are now widely available. However, they were developed in different technologies (like Python, Java, R, C++) and use a variety of different formats of data that are very often not compatible with each other. Many language tools, especially for languages different than English, are hard to be installed without skills in programming. They usually have a large number of different parameters to be set-up. Interpretation of the parameters mostly requires knowledge from Computational Linguistics. Therefore, their integration and building its own processing workflow is not a simple task especially for researchers without any experience in computer engineering.

CLARIN proposes making linguistic tools available on the Internet (e.g. as Web Services and simple Web Applications) and developing research web based applications [17], as a means to diminish the above mentioned problems. Thus, users can process data online and do not need to bother about technicalities, but still need to understand processing mechanisms on the level of their presentation.

However, building a multi-user, web system generates other set of problems connected with the system availability and performance. The system should be *scalable*, *responsive* and *available* all the time. Language tools have excessive CPU/memory consumption and a number of users or tasks at a given time is unpredictable. Therefore, development of a responsive and highly available web system for large scale text processing texts is challenging.

Main assumptions behind WebSty are:

1. no need for installation
2. no need for setting up the tools
3. providing preselected processing settings for different tasks typical for research Humanities and Social Sciences.

It is worth noting that in order to fulfill the last requirement we need different benchmark data sets representative for H&SS research tasks to tune on the system.

Many of NLP tools (like Named Entity Recognizers [18] or Word Sense Disambiguation tools) have large knowledge models. In such cases, the time of loading a model to the operating memory is much longer than processing a single text file. This can be avoided by running a tool as a service with pre-loaded data models kept later in memory. Each service is run as a separate process. The usage of services communicating with others by lightweight mechanisms also

[3] http://www.cs.waikato.ac.nz/ml/weka/
[4] https://nlp.stanford.edu/software/
[5] http://mallet.cs.umass.edu

```
urlzip("http://ws.clarin-pl.eu/public/teksty/2mini.zip")|any2txt|wcrft2|
fextor({"features":"base interp_signs bigrams ","filters":{"base": [{"type": "lemma_stoplist", "args":
{"stoplist": "@resources/fextor/nkjp360-meaningless-no-prep-freq-above-3500.txt"}}]}})|
dir|featfilt({"similarity":"cosine","weighting":"all:tf-idf","filter":"min_df-1 min_tf-1"})|
cluto({"no_clusters":2,"analysis_type":"plottree"})
```

Fig. 1. Exemplar LPMN for stylometric analysis

solves the problem of various programming languages used by language and ML tools, as there is no need for tight integration. It results in an architecture of the system based on microservices [19]. An architecture style following service-oriented [20] ideas that has recently started gaining huge popularity. The number of parallel run tools (microservices) is limited by hardware (size of memory/number of processors). That is why a queening system is required to perform such tasks effectively. Each type of a language tool has its own queue. NLP microservice collects tasks from a given queue and sends back messages when results are available. Such solution allows providing effective scalability capabilities. The required, mostly used, NLP microservices have to be run in several instances since a queuing system acts as a load balancer.

Having language tools implemented as microservices there is a need to describe cooperation of them to realize specific Text Mining tasks. Therefore, we have developed [21] a human readable orchestration [22] language that allows to describe different text processing tasks. It is called Language Processing Modelling Notation (LPMN). LPMN is a formal language defined in LL(*) gramma [23]. The LPMN statement for WebSty processing is presented in Fig. 1.

In the implementation we have used the AMQP[6] protocol for lightweight communication mechanisms and open source RabbitMQ[7] broker for a queuing system. AMQP protocol has clients for a large number of different software platforms as required by technologies used by language and ML tools. In the proposed architecture (Fig. 2) an additional server grants access from the Internet. It works as a proxy for the core system delivering REST API. Such an approach allows for easy integration with almost any kind of application. In addition, the engine for running workflows described in LPMN was developed. It allows to process a large corpus of text in a batch like mode.

The exchange of data between microservices, i.e. inputs to NLP tools and results of their processing, is made by a network file system. It makes integration of NLP tools easier since they are mostly designed in the manner that they expect a file as in input and produce files as an output. Moreover, the data formats used by NLP tools are very exhaustive and the size of the files is several times larger than input text data. Therefore using messages in AMQP protocol for sending processed data would be very inefficient.

To achieve high availability requirements the system was deployed on a scalable hardware and software architecture that forms a private cloud. The hardware consists of ten Blade Servers, connected by a fast fiber channel with highly scalable midrange virtual storage designed to consolidate workloads into a single system for simplicity of management. XENServer controls each machine and forms a private cloud. Each important, more frequently used NLP microservice is deployed on a separate virtual machine. Therefore, it is easy to scale up the system just by duplicating virtual machines as a reaction to a high number of requests for given types of tools. Virtualization provides a disaster recovery mechanism ensuring that when a virtualized system crashes, it will be restored as quickly as possible and virtual machines can be easily moved to another server as a reaction to any failure or resource shortage [24].

V. DATA ANALYSIS

Stylometric techniques are based on converting first text documents or fragments into vectors of numerical values and next processing the resulting vectors by various data analysis methods. From the technical point of view the ultimate goal of the analysis is to divide the vectors into similarity classes, e.g. documents of the same author. This can be achieved by clustering algorithms (unsupervised approach) or classifiers trained by ML on examples of text documents with known properties (supervised approach). As the vast majority of the clustering and ML algorithms used in stylometry are well known and not specific for this domain, the key issues are: definitions of features for text description and methods for their further processing. The features are defined on the basis of characteristic elements of the text linguistic structure or document, their initial values are frequencies of these elements, and processing techniques are used to clean the vectors from noise and to compare the vectors.

[6] https://www.amqp.org
[7] https://www.rabbitmq.com

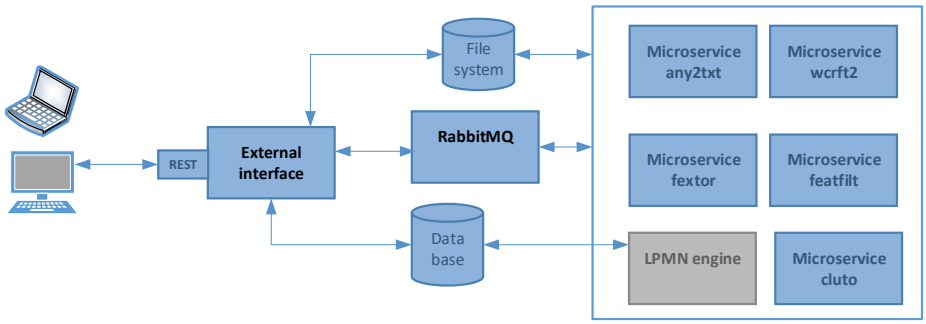


Fig. 2. System architecture

V. 1. Feature extraction

Features should reveal properties of text that are characteristic of an author or his/her style. They should be correlated with the semantic content of the text. Features can refer to any level of the language analysis or text form. However, they should be based only on language tools that express relatively small error, as errors produced by language tools increase the level of noise in data.

For WebSty we selected several types of features that can be generated for texts with the existing language tools for Polish:

1. word forms (words in text)
2. punctuation
3. lemmas[8]
4. grammatical classes (according to the detailed tagset)
5. Parts of Speech (as sets of grammatical classes)
6. bigrams and trigrams of grammatical classes (2 and 3 element sequences)
7. semantic types of Proper Names.

Initial values of features as extracted from text are frequencies of particular instances of the feature types, e.g. the frequency of a particular word, grammatical class or a bigram.

The lemmas and grammatical classes are obtained from WCRFT2 [25] tagger, occurrence and types of name entities by Liner2 [18] tool.

Feature vectors are generated separately for each document. For all documents they form a feature matrix with rows representing documents and columns frequencies of a feature occurrence in a given document.

V. 2. Feature filtering

Features which are suspected to introduce too much noise or to be irrelevant can be filtered out on the basis of: their instance, raw value or weighted value (after preprocessing).

It is possible to eliminate particular types of features, e.g. specified lemmas (a stop word list) or define a list of analysed lemmas (a list of analysed lemas), selected punctuation marks or given grammatical classes.

Two simple filtering methods were introduced in order to remove features that occur very rarely. They are based on the following two types of criteria:

1. the minimal number of the feature occurrences in the whole collection
2. the minimal number of the feature occurrences in the given document.

V. 3. Feature weighting

Raw frequencies are often skewed by document length, document content, text domain, selection of text or by general properties of the given feature instance in a given language (e.g. word *nowy* 'new' is much more frequent than most of the other adjectives). Therefore, in a vast majority of cases it is better to replace the raw frequency values with values that are normalised in relation to the document length and also express the relative importance of the occurrences of these features for analysed texts. Such mapping is called *weighting*. WebSty offers the following weighting methods (formal specifications of the methods are presented in Appendix A):

1. *tf* - Text Frequency, values normalised by the maximum frequency in a document
2. *tf.idf* – Text Frequency multiplied by the Inverted Document Frequency [26], often used in vector models of the Information Retrieval
3. *normalize* – vector normalised to the length 1
4. *mi* – Pointwise Mutual Information (PMI) measure calculated for feature vs document
5. *mi-simple* – PMI measure multiplied by the discounting factor which is intended to increase weights of more frequent features [27]
6. *tscore* – T-score, the scaled statistical standard score association measure.

Most weighting methods (except *tf* and *normalize*) have an intrinsic ability to filter out non-informative features. e.g. *tf-idf* filters out features that exists in all documents (by setting their values to zero), so they cannot be applied to the features based on the most frequent word or lemma in a text corpus.

[8] Lemmas are simply understood here as basic morphological forms representing sets of word forms that differ only in the values of grammatical categories.

It is important to note here that after all weighting schemes the achieved values of feature vectors are not negative (assuming that negative PMI are changed to zero, that is a typical practice). This observation has a big influence on the next steps of processing. Since, for example the most popular similarity metric for high dimensional data – cosine – is always positive (or equal to zero).

V. 4. Dimensionality reduction

The number of features and therefore the size of feature vectors can be very high, e.g. there can be several million words (due to Proper Names, foreign words, misspelled forms etc.) in a large collection of text files. This number very often goes much beyond 1000. However, features are very unevenly distributed across texts, e.g. words/lemma frequencies they are distributed according to a power law referred to as Zipf's distribution, and the matrix of text vectors can be very sparse, where the vast majority of vector values are zero or close to zero. The problem is not only the vector/matrix size, but also that low values are very often a result of accidental, non-informative occurrences.

Several transformations that transform the feature matrix to lower dimensionality are proposed in the literature. They are based on the assumption that the original matrix includes noise, but we can find subspaces of larger density inside it and a new orthogonal basis can be found of lower dimensionality but with orthonormal eigenvectors representing most significant directions of the largest density. A matrix of a hundred thousand features can be reduced to a couple of hundred with a minimal loss in the reconstruction error. However, the most important is that dimensionality reduction removes a lot of noise and results in a form of generalisation of the similarity structure (it can also have a negative effect, because some important subtle differences can get lost by the generalisation).Dimensionality reduction does not change individual values, but instead transforms the whole matrix of feature vectors. The new vectors provide some kind of generalisation in a way that emphasizes the most important similarities and differences between documents. Accidental associations are mostly reduced during this transformation. Consequently, the level of statistical noise should be lower. However, in many tasks the introduced generalisation goes too far and some distinctions that we are interested in disappear.

WebSty offers the following dimensionality reduction techniques:

1. *SVD* – Singular Value Decomposition [28]
2. *LSA* – Latent Semantic Analysis, a method of Distributional Semantics that utilises SVD, in WebSty LSA means feature transformation that combines entropy normalisation with tf-idf weighting and final SVD transformation [29, 30]
3. *Random projection* – a dimensionality reduction technique which assumes that vectors are located in Euclidean space, preserves distances [28]

It is important to state that the new calculated features are abstract and do not have any intuitive interpretation. Moreover, their values could go below zero, since they have no frequency interpretation. It could result in problems in the next step of processing similarity calculation, i.e. the similarity could have values below zero. In such cases the similarity value is truncated to zero.

V. 5. Similarities and distances

The similarity of texts is computed on the basis of similarity of their transformed vectors. Texts are grouped on the basis of the vector similarity or distances. So, similarity distance calculation methods are in the heart of stylometric techniques.

Weighted and transformed data vectors are compared against each other and a similarity matrix is computed. We have assumed that the similarity values are between 0 and 1, where 1 means high similarity of both texts (or rather weighted vectors representing each text) and 0 signalls very significant differences between text and thus their dissimilarity. There are dozens of different methods for calculating similarity between data vectors. Several of them have been implemented in WebSty on the basis of their performance in preliminary experiments done on the *1000 Novels* collection [31] (the measures' specifications are given in Appendix A):

1. *cosine* – a cosine of the angle between two vectors
2. *dice* – a heuristic measure, a ratio of the amount of shared features to all features in two vectors [32, 33]
3. *Jacquard* – a heuristic measure, a ratio of the amount of shared features to the features specific in two vector
4. *ratio* – heuristics that measures approximately the average ratio of commonality in the features of two vectors
5. *shd* – a heuristic measure of rendering precision of one vector by other, i.e. how one vector can repeat the feature values of the other and vice versa, the measure can be made non-symmetrical by changing the value of the parameter.

The counterpart of a similarity is a distance measure (required e.g. by some clustering algorithms), in which values close to zero mean that documents (feature vectors) are similar, and large values mean that they are dissimilar. We have implemented the following distance measures:

1. *euclidean* – L2 distance, when vectors are treated as points in the Euclidean space
2. *manhattan*- L1 distance
3. *Canberra* – a weighted version of Li distance
4. *Simple* – is done by normalizing the input dataset by a square root function, and then applying Li distance [34]
5. *Delta* – classical Burrows delta measure ([35]), defined as Manhattan distance applied to a scaled (z-scored) dataset

6. *Argamon* – relies on Euclidean distance combined with Z-score normalisation of the input dataset [36]
7. *Eder* – is a modification of the Burrows delta measure; it slightly increases the weights of frequent features and rescales less frequent ones in order to suppress discriminative strength of some random infrequent features [34].

The methods used in the next step of processing (i.e. clustering, multidimensional scaling and result visualization) are based on similarity or distance values. To process data regardless of the selected similarity/distance measure, there is a need to convert similarities (s) to distances (d) and distances to similarities. In case of all similarity measures regardless of cosine one the distance (d) is calculated as a simple negation of similarity (s):

$$d = 1 - s$$

The cosine similarity is converted to distance by the arc cosine function, see e.g. [37], i.e.:

$$d = 2 * \arccos(s)/\pi$$

The distance measures are converted to similarities by an inversion operation, i.e.:

$$s = \frac{1}{1+d}$$

V. 6. Clustering

Clustering is a task of grouping texts into, a mostly predefined number of clusters (groups) based on similarity or distance measures. The main assumption is that documents in the same cluster are more similar or closer (in a sense of used measure) to each other than to those in other clusters.

Among different clustering algorithms the *combined agglomerative-flat clustering method* implemented in the Cluto tool [38] was selected for WebSty as it provides two perspectives on the same data set: pairwise hierarchy of similarity and division of the set into a predefined, expected number of clusters. Moreover Cluto also presented good results in applications to text clustering.

In agglomerative clustering, in each step two most similar clusters are found and merged into a new one. Such a "bottom-up" process starts with the initial set of singleton clusters including one document each, i.e. in the beginning each document is treated as a cluster. As a result, a tree-like hierarchy of clusters (also referred to as a dendrogram) is established. Particular documents are put at the bottom, and any similarity relations between them, as well as any relation between larger clusters, are represented by appropriate links, see the example in Fig. 3. The clustering process is controlled by three parameters: number of clusters, similarity measure and clustering criterion function. Selecting the clustering criterion function is a complicated issue. Mostly it is done by tuning a method on a training set. WebSty provides some ready to use defaults that should provide reasonably good results in some typical tasks.

VI. DATA VISUALISATION

VI. 1. Available data and clustering results

The process of data analysis described in the previous section provides two different types of results for each input corpus and selected options. It includes results of similarity/distances calculation and results of clustering. The similarity/distance results are real numbers defined for each pair of input texts. They form two (similarity, distance) 2-D matrices of a size equal to a number of texts in input corpora. The number of input texts could be automatically enlarged if a user selects an automatic chunking of input data into smaller (with defined size) parts. The methods of presenting similarity results are presented in Section 6.2 whereas distances in 6.3. Presentation of the clustering results includes a group membership and a dendrogram. The group membership is shown to a user in two different ways. First of all, the user can download the relation between the input file name and the identifier of a cluster that it was assigned to in a form of an XLSX file. Secondly, in most graphical presentations of results, it is shown using the same colour for names of files belonging to the same cluster (see Fig. 3, 4). The dendrogram on Fig. 3 is presented as an interactive binary tree where each node (and a subtree) could be hidden.

The plotting of results in the web browser is done by the usage of D3.js[9] library. The server side application returns results in the JSON format, data are analysed by JavaScript code and results displayed on the client size. Such solution allows interactive presentation of data.

VI. 2. Similarity results

Similarity results are presented in two ways: in a form of a heatmap (Fig. 5) and schemaball (Fig. 6). The heatmap allows to analyse text-to-text similarities by colours (from red – almost identical – to light green – very different) and numerical values (by moving a mouse over rectangles representing two texts).

In the schemaball plot (Fig. 6) a user could select a file name and analyse similarity of texts by shown connections. The colour and thickness of the connection line depends on the similarity value.

VI. 3. Multidimensional scaling

One of the methods of data similarity visualisation based on the distance matrix is a multidimensional scaling. Its starting point is a nonlinear dimensionality reduction. It aims to present data in a low dimensional space (in our case in 2 and 3D space) in a such way that elements which are close

[9] https://d3js.org/

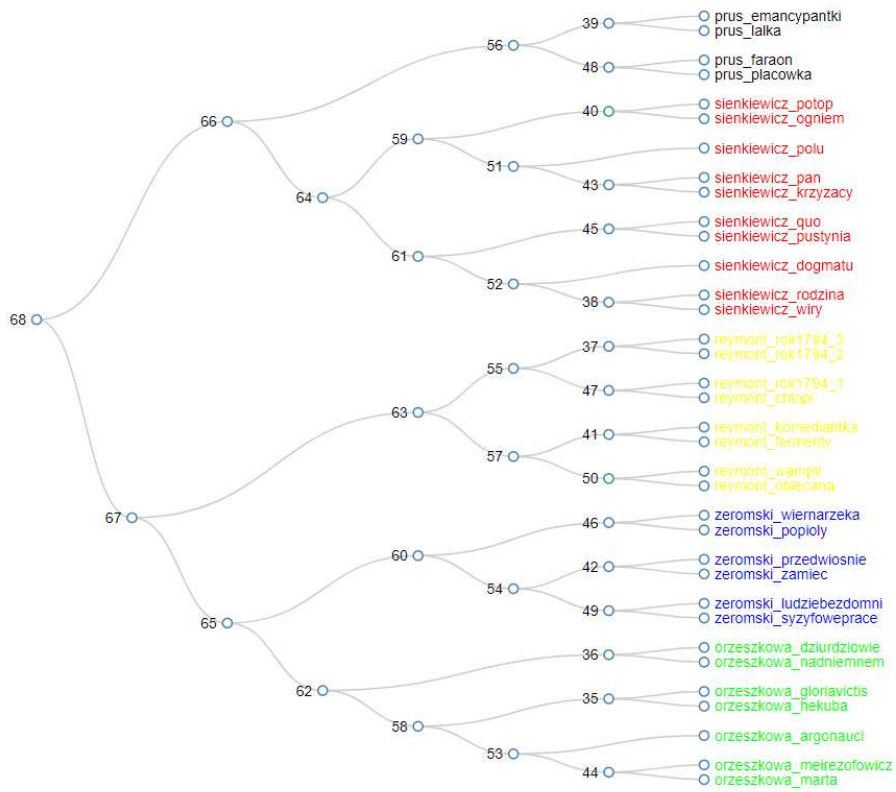


Fig. 3. Clustering results (dendrogram and cluster membership) in a form of interactive dendrograms

in relation to the original measure (in general distances) are as close to each other in the low dimensional space as possible. We have included into WebSty four different methods of multidimensional scaling methods:

1. *metric* [39] – preserves distances (relative values), Euclidean distance is used in low dimensional space,
2. *non-metric* [39] – preserves orders in distances, data ordered by distances in original space are in the same order in low dimensional space (Euclidean distance is assumed in low dimensional space),
3. *t-distributed Stochastic Neighbor Embedding* [40] – preserves similarities,
4. *spectral embedding* [41] – preserves local neighborhood.

The user is able to see the results – points with file name labels in 2D space (Fig. 7) or in 3D space (Fig. 8). The 2D plot is implemented with the help of the 3D.js library. The interactive 3D plot utilises the three.js[10] library which is based on WebGL[11] technology. It uses user computer graphic card 3D acceleration. The 3D space is slowly moving around to allow a perception of 3D geometry. On more distance in the presentation, in comparison to the 2D presentation, allows often to spot associations in the data that are not clearly visible in 2D. Animated rotation provides an opportunity to analyse the data from different angles.

VII. CONCLUSIONS AND FURTHER DEVELOPMENT

WebSty was implemented[12] as a part of the CLARIN-PL[13] infrastructure and made publicly available to the CLARIN-PL users. So far, WebSty has provided only analysis methods based on unsupervised clustering[14] of texts. Evaluation of the clustering methods is not straightforward due to the difficult of manually constructing a golden dataset of clusters to compare with. Moreover, small differences in boundaries of the clusters: automatically and manually built ones can be always expected. Concerning the first problem, some stylometric tasks are precisely defined, e.g. authorship attribution or author's gender recognition, while for some

[10]https://threejs.org/
[11]http://www.khronos.org/registry/webgl/specs/latest/
[12]http://ws.clarin-pl.eu/websty.shtml (also http://websty.clarin-pl.eu)
[13]http://www.clarin-pl.eu
[14]Works on expanding WebSty with analysis based on Machine Leaning and supervised classification are very advanced and the main problems are User Interface and asynchronous performance of lengthy processes.

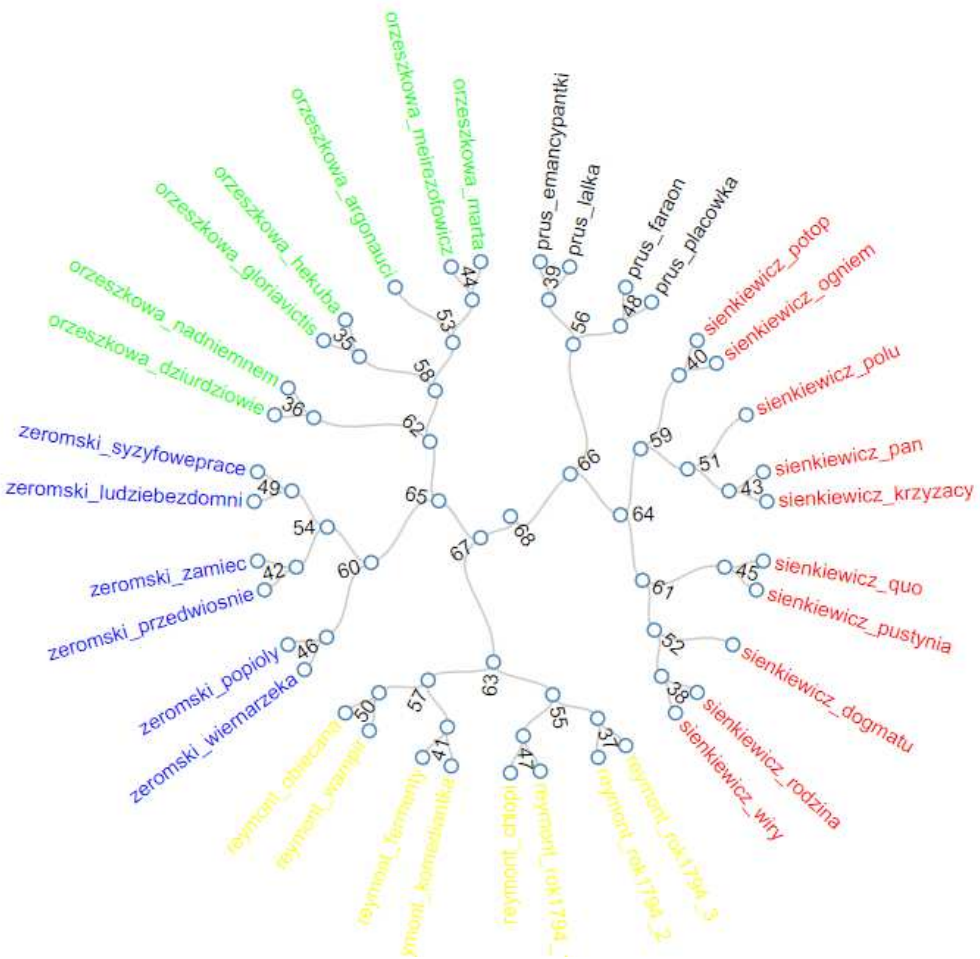


Fig. 4. Clustering results (dendrogram and cluster membership) in a form of circle (in the presented results two clusters were selected in contrast to results in Fig, 3 were five clusters were selected)

tasks, e.g. literary style recognition there are no established definitions. Due to the issue of copyrights, there is also a problem with building a standard golden data set for the stylometric analysis of Polish.[15]

As a basis for a preliminary evaluation of WebSty, we selected *1000 Novels Corpus*[16] [31], henceforth *1kNC*. It includes 1000 literary texts in Polish. Most of them are texts of classical Polish authors from the second half of the 19th century and the beginning of the 20th century. Texts in *1kNC* are not evenly distributed concerning their authors and size. Many of them are novels, but there are also many short stories. Several authors are represented by more than 20 texts, but the vast majority is represented by a few or even single texts. A large number of texts were translated from some other language than Polish. Texts were also published by several different publishers, which resulted, e.g., in different standards of correcting punctuation. Many of the texts include some small percentage of words written in old morphology or even archaic. They slightly decrease the performance of the morphosyntactic tagger. In the case of all texts we have manually removed all information concerning the publisher and moved it to the metadata attached to each file. Summing up, *1kNC* is a quite reasonable literary data to work on, but not perfect, so the results of the tests presented below must be treated as preliminary and providing only the first insights into the performance of WebSty on different settings.

During the tests on *1kNC* we used several groups of features supported by *WebSty*.

1. *lemmas* – a set of frequent Polish lemmas that seem not to carry the meaning directly related to the content of literary works; selected *punctuation* signs (i.e.: ".,:;?!"), *grammatical classes*, *bigrams* of grammatical classes – all classes and bigrams including classes representing Proper Names, i.e. *brev* and *xxx*, were ex-

[15]However, similar problems can be observed in the case of other natural languages.
[16]http://hdl.handle.net/11321/312

Fig. 5. Similarity results in the form of a heatmap

cluded[17] and trigrams of grammatical classes including only: adjectives, adverbs and nouns[18].

2. only features based on filtered grammatical classes and punctuation signs (as in A.): *punctuation* signs, *grammatical classes* and *bigrams* of grammatical classes.
3. only lexical features: *lemmas* and selected *punctuation* signs.

The list of the content insensitive lemmas was built on the basis of the list of the 500 most frequent lemmas extracted from the National Corpus of Polish [42]. However, this was not a perfect choice as this list included many con-

[17]Many words in old morphology or archaic were erroneously recognised as Proper Names. In addition, due to our preliminary experiments, the density of Proper Names in texts also seems to depend more on the content of the texts than on its stylometric features like author or literary style.

[18]Trigrams of these classes are intended to provide the description of the structures of noun phrases used by in a given text.

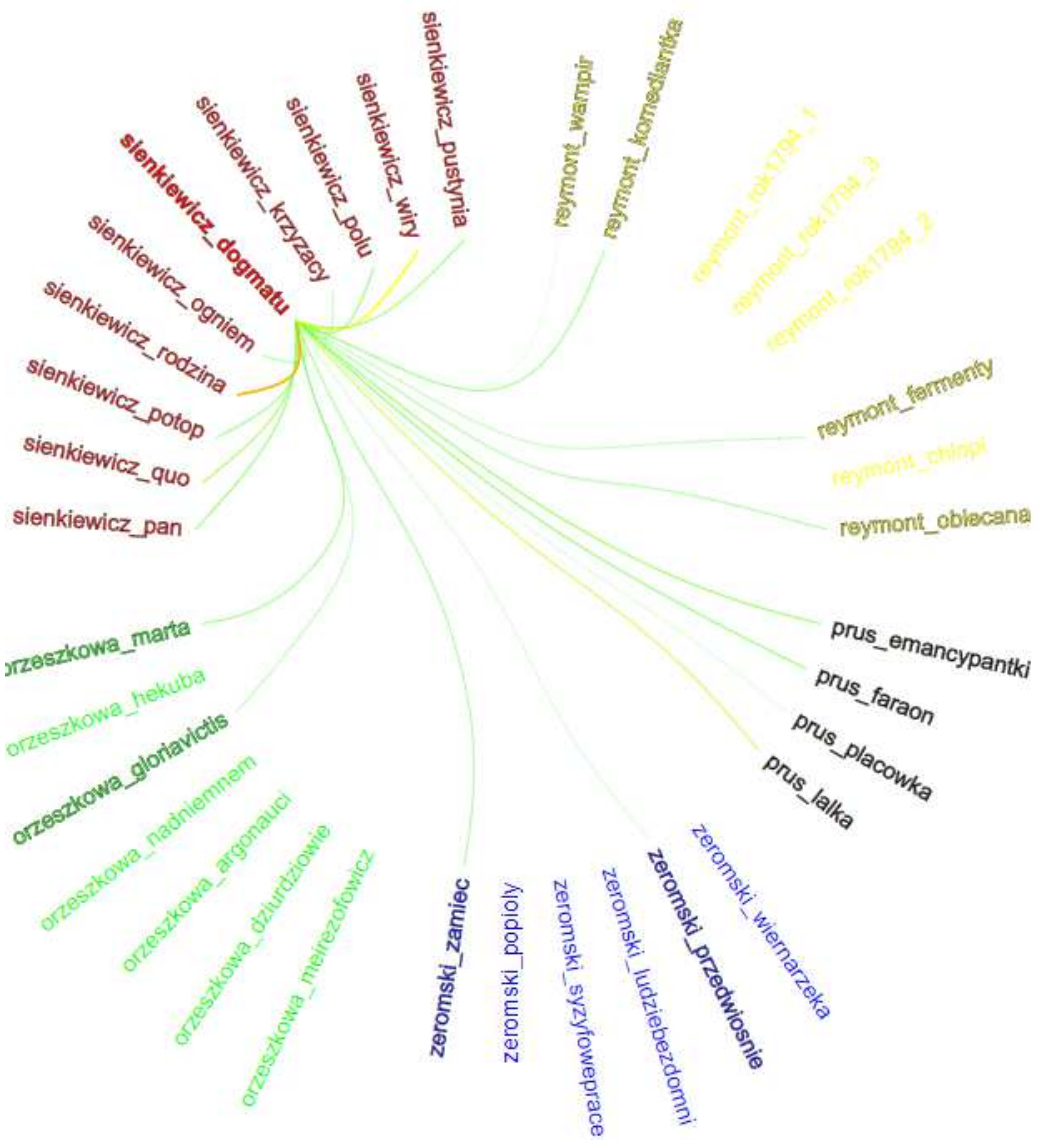


Fig. 6. Similarity results in the form of a schemaball

tent sensitive words. Some of them clearly resulted from the lack of balance in the corpus, e.g. words used during parliamentary debates. The list was manually filtered during clustering experiments performed on the collection of blogs [43], and also on *1kNC*. All words that appeared to be significant features in the clusters univocally representing some topics were manually eliminated. Next, several classes of words located on top of the reduced list, like conjunctions, adverbs and participles, were carefully expanded manually and the resulting expanded lists were tested again, e.g. in the case of adverbs, many quantifying adverbs were added, but not those related to time and space as they seemed to influence the clustering process towards the content of the literary works. As a result, we obtained several versions of the lemma list, starting with the list of 360 lemmas (finally offered in the WebSty user interface as one of the feature types) and ending with the list including only 212 lemmas.

In order to reduce the influence of too infrequent features that can cause accidental similarities between texts, we set the filter for the minimal number of occurrences to (*Tf*) to 100 and the filter of the minimal number of texts including the given feature (*df*) to 20 for all experiments. Graph-based clustering methods from Cluto were applied on the basis of the precomputed text-to-text similarity matrix.
Results of the selected experiments are presented in Tab. 1.

For the evaluation of the clustering results we used *entropy* and *purity* measures, e.g. [38]. Both are calculated in relation to the classes of texts, i.e. the authors of the texts. *Entropy measure* value is a weighted average of entropy calculated for all clusters, i.e. the entropy shows how diversified

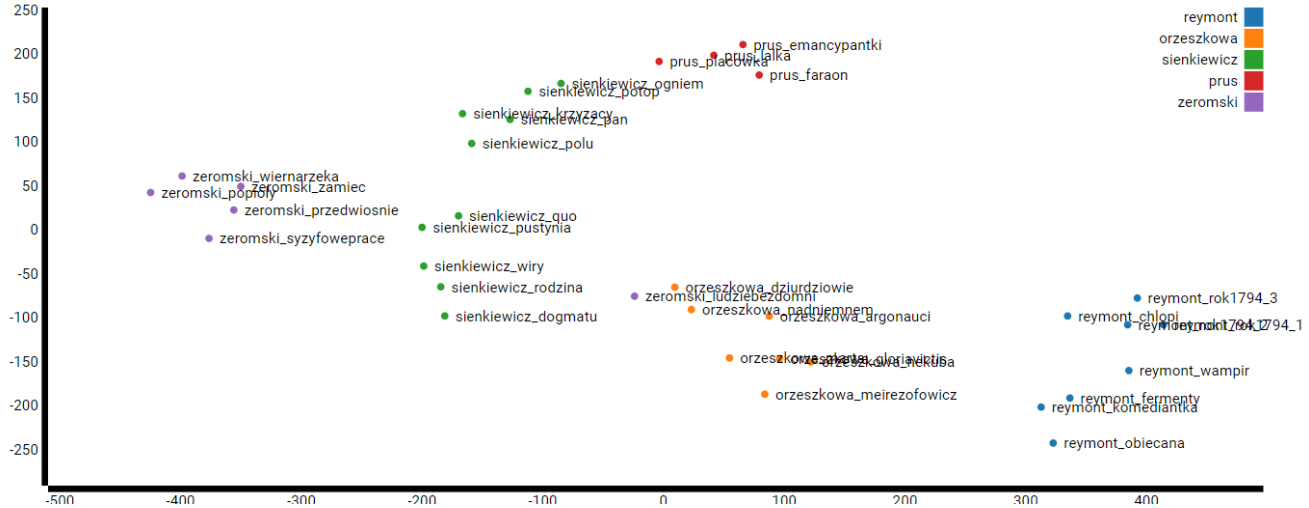


Fig. 7. Distance results in the form of 2D plot

Tab. 1. Results of the evaluation of different settings of WebSty in authorship attribution on *1000 Novels Corpus*, where: *W* is a weighting method, *Sim* – similarity measure between texts, RB+I2 – the clustering algorithm of repeated bisections with I2 criterion [38], Bagglo – agglomerative clustering biased by the initial partitional clustering, RBR – repeated bisections with the global optimisation at the end, *L* – is the size of the lemma list, 360+500 – the union of the manually selected lemmas with the list of the 500 most frequent in the National Corpus of Polish, *Ent* – the entropy measure for clustering evaluation and *Purity* – the purity measure

Features	No of clusters	W	Sim.	Algorithm	L	Ent.	Purity
A+trigrams from {adj, adv, subst}	40	PMI	Ratio	RB+I2	360	0.316	0.542
A+trigrams from {adj, adv, subst}	80	PMI	Ratio	RB+I2	360	0.240	0.623
B	80	PMI	Ratio	RB+I2	360	0.316	0.443
C	80	PMI	Ratio	RB+I2	**360**	**0.198**	**0.649**
A	120	PMI	Ratio	RB+I2	360	0.197	0.665
B	120	PMI	Ratio	RB+I2	360	0.267	0.486
C	120	PMI	Ratio	RB+I2	**360**	**0.157**	**0.687**
A+trigrams from {adj, adv, subst}	120	PMI	Cos	Bagglo+I2	360	0.305	0.433
A+trigrams from {adj, adv, subst}	120	PMI	Cos	RBR	360	0.287	0.431
C	120	PMI	Ratio	RB+I2	360 & 500	0.186	0.653
A+trigrams from {adj, adv, subst}	80	PMI	Ratio	RB+I2	212	0.288	0.498
C	80	PMI	Ratio	RB+I2	360 & 500	0.218	0.619

the clusters are. The purity measure is based on defining for each cluster its majority class and next calculating the percentage of texts belonging to this class. Purity for the whole clustering result is calculated as the average. Purity describes how well the resulting clusters resemble the manually defined classes.

In the case of the authorship attribution task the best results were achieved when using only selected lemmas and punctuation symbols as features. We can notice that with the increasing number of clusters the measure values are also increasing. In the case of purity this is natural, as smaller clusters make dominance of a class in a cluster easier. However, the increasing value of entropy shows that in this case with the increasing number of clusters they are becoming more coherent with respect to the included authors. The larger number of clusters also allows for discovering small individual clusters for authors of only several works. We can also note that the list of the selected 360 lemmas seems to be a good basis for authorship attribution. In the case of a smaller list and a larger one extracted automatically from a very large corpus the results are lower. As the selected list is not dependent on any particular corpus, initially originated from a very large corpus and was built in a way maximising its independence from particular topics of texts, it can be used for the authorship attribution of different types of texts and that is why it is suggested in the appropriate configuration of WebSty to users.

The clustering results are far from perfect because of the large number of authors with a small number of texts, as well as the different sizes of texts in *1kNC*. Manual inspec-

tion has revealed that in all settings there are 1-2 bigger clusters grouping a large number of texts of quite many authors. This phenomenon is caused by the problem of the clustering method in finding common features for some texts.

Features based on grammatical classes produced slightly worse results in authorship attribution, but we could observe that they express an ability to identify literary works of the same style of genre. This was later confirmed by the results of the application of WebSty to the analysis of literary styles of web blogs reported below.

WebSty (different versions) has already been applied also to several research tasks from the area of H&SS, as well as used in teaching. Among its applications it is worth mentioning literary analysis of web blogs from the point of view of the styles they represent, see [43]. Due to the flexibility in our system in selecting and combining features and processing methods, it was possible to find text descriptions that go beyond a typical setting in authorship attribution, but seem to approximate the notion of literary style of blogs, as it had been first implicitly expressed in the manually annotated golden standard data set. In fact, the analysis presented in [43] had been first made on a prototype from which the selected properties and techniques were later included into the official, public version of WebSty. The best clustering results in comparison to style classes defined by human annotators were achieved by combining lemmas from the 360 list together with grammatical classes and bigrams of grammatical classes. The entropy was 0.438 and purity 0.604. It is slightly lower than the values observed in the experiments on *1kNC*, but the task of style recognition is much harder and the inter-annotator agreement was quite low. As the experiments were performed on the whole blogs, we expect to obtain better results in experiments done in the future on single posts.

Applications of WebSty in teaching students of Polish philology and experience collected during CLARIN-PL training workshops showed that the idea of installation-free, web application for stylometry goes into the right direction. Moreover, we have also learned that we need to put efforts on the development of rich user guide and pre-defined application setting for the different tasks in H&SS. Here appears a problem that we must work on: collecting or preparing different golden datasets for different tasks in stylometry.

WebSty has so far been focused on unsupervised processing by clustering. We are working on an extended version offering support for using a supervised approach in which classifiers are trained by ML on the basis of manually annotated datasets, e.g. a set of texts annotated with authors.

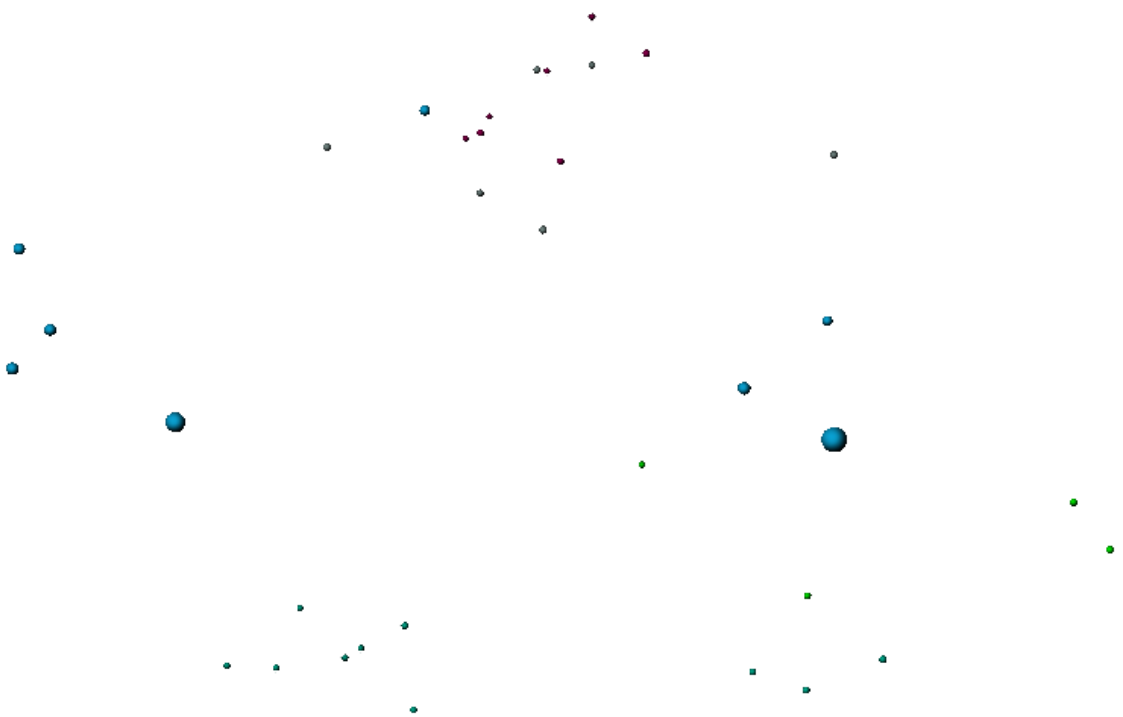


Fig. 8. Distance results in the form of 3D plot

We also plan to extend WebSty into a system enabling unsupervised and supervised semantic analysis of text data sets, e.g. identification of text fragments that are related to situations of specific types or to specific phenomena.

Topic analysis will be very soon included into WebSty as a tool by itself and also as a tool used for data preprocessing.

WebSty has already been expanded towards a multilingual version[19] with support for English, German and Hungarian. Further development towards the support for French, Slovenian and Spanish is planned. We intend to continue this process and modify WebSty architecture into a multilingual system.

Acknowledgements

Works funded by the Polish Ministry of Science and Higher Education within CLARIN-PL Research Infrastructure.

Appendix

A Definition of weighting methods

1. notation
   (a) *t*- feature, *d* - document
   (b) *f(t,d)* – raw frequencies of feature *t* in *d*-document
   (c) *N* – number of documents
2. TF
$$tf(t,d) = \frac{f(t,d)}{\max(f(t',d) \text{ for all } t' \text{ in } d)}$$
3. TF-IDF
$$\text{tfidf}(t,d) = \text{tf}(t,d) \cdot \log \frac{N}{\text{count}(f(t,d') > 0 \text{ for } d')}$$
4. Normalize
$$\text{norm}(t,d) = \frac{f(t,d)}{\sqrt{\sum_{t'} (f(t',d) f(t',d))}}$$
5. Mi-simple
$$\text{mis}'(t,d) = \log \frac{f(t,d) \cdot \sum_{t',d'} f(t',d')}{\sum_{t'} f(t',d) \cdot \sum_{d'} f(t,d')}$$
$$\text{discount}(t,d) = \frac{f(t,d) m(t,d)}{(m(t,d)+1)\,(f(t,d)+1)}$$
$$\text{mis}(t,d) = \text{mis}'(t,d) \cdot \text{discount}(t,d)$$
$$m(t,d) = \min\left(\sum_{t'} f(t',d), \sum_{d'} f(t,d')\right)$$

B Definition of similarity measures

1. notation
   (a) N – size of feature vector
   (b) $a,\ b$ – feature vectors,
2. cosine
$$s(a,b) = \frac{\sum_{i=1}^{N} a_i b_i}{\sqrt{\sum_{i=1}^{N} a_i a_i}\sqrt{\sum_{i=1}^{N} b_i b_i}}$$
3. ratio
$$s(a,b) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{(a_i + b_i)}{\max(a_i, b_i)} - 1\right)$$
4. dice
$$s(a,b) = \frac{2\sum_{i=1}^{N} a_i b_i}{\sum_{i=1}^{N} a_i a_i \cdot \sum_{i=1}^{N} b_i b_i}$$
5. Jaccard
$$s(a,b) = \frac{\sum_{i=1}^{N} a_i b_i}{\sum_{i=1}^{N} a_i a_i + \sum_{i=1}^{N} b_i b_i - \sum_{i=1}^{N} a_i b_i}$$
6. SHD
$$\text{prec}(a,b) = \frac{\sum_{i=1}^{N} |a_i - b_i|}{\sum_{i=1}^{N} a_i}$$
$$s(a,b) = 1 - \frac{1}{\alpha/\text{prec}(a,b) + (1-\alpha)\text{prec}(b,a)}$$

References

[1] P. Juola Authorship attribution. Foundations and Trends in Information Retrieval **1**(3), 233–334 (2006).

[2] M. Koppel, J. Schler, S. Argamon Computational methods in authorship attribution. Journal of the American Society for Information Science and Technology, **60**(1), 9–26 (2009).

[3] E. Stamatatos A survey of modern authorship attribution methods. Journal of the American Society for Information Science and Technology **60**(3), 538–556 (2009).

[4] Le, X., I. Lancashire, G. Hirst, R. Jokel Longitudinal detection of dementia through lexical and syntactic changes in writing: a case study of three British novelists. Literary and Linguistic Computing, 26(4): 435–461 (2011).

[5] *Signature* Stylometric System (access Apr. 2017). Web Page of the system. URL: http://www.philocomp.net/humanities/signature.htm.

[19] http://websty.clarin-pl.eu

[6] Maurer, Leon (access Apr. 2017) Web page of the StyleTool programURL:https://github.com/lnmaurer/StyleTool.

[7] S. Bird, E. Klein, E. Loper, (2009) Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit. O'Reilly Media, URL: http://www.nltk.org/book_1ed/.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011).

[9] JGAAP (accees Apr. 2017) Web page of the application. URL: https://github.com/evllabs/JGAAP.

[10] A. McDonald, S. Afroz, A. Caliskan, A. Stolerman, R. Greenstadt, Rachel (2012) Use Fewer Instances of the Letter "i": Toward Writing Style Anonymization. PETS 2012.

[11] I.H. Witten, Ian H., Frank, Eibe, Hall, Mark A., Christopher J. Pal Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufman (2017).

[12] S. Sinclair, Geoffrey Rockwell and the Voyant Tools Team (2012) Voyant Tools (web application). URL: http://docs.voyant-tools.org.

[13] C.D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S.J.Bethard, D. McClosky, The Stanford CoreNLP Natural Language Processing Toolkit. Association for Computational Linguistics (ACL) 2014 – System Demonstrations, ACL (2014).

[14] A.K. McCallum, MALLET: A Machine Learning for Language Toolkit. Web page of the system. URL: http://mallet.cs.umass.edu (2002) .

[15] LATtice - Application for Visualizing Linguistic Variation (access Apr. 2017) WEb page of the application URL: http://winedarksea.org/?p=1285

[16] M. Eder, J. Rybicki, M. Kestemont Stylometry with R: a package for computational text analysis. R Journal, 8(1): 107–121, http://journal.r-project.org/archive/2016-1/eder-rybicki-kestemont.pdf (2016).

[17] P. Wittenburg, et al. Resource and Service Centres as the Backbone for a Sustainable Service Infrastructure. In: Proceedings of the International Conference on Language Resources and Evaluation, pp. 60–63. European Language Resources Association (2010).

[18] M. Marcińczuk, J. Kocoń, M. Janicki, Liner2 - A Customizable Framework for Proper Names Recognition for Polish. Studies in Computational Intelligence, vol. 467, pp. 231–253 (2013).

[19] E. Wolff Microservices: Flexible Software Architectures, Addison-Wesley (2016).

[20] M. Bell SOA Modeling Patterns for Service-Oriented Discovery and Analysis. Wiley & Sons. (2010).

[21] T. Walkowiak Language Processing Modelling Notation – orchestration of NLP microservices. In: Advances in Dependability Engineering of Complex Systems: Proceedings of the Twelfth International Conference on Dependability and Complex Systems DepCoS-RELCOMEX, 2017, Springer International Publishing, pp. 464-473 (2017).

[22] C. Peltz, Web services orchestration and choreography. Computer **36**(10), 46–52 (2013).

[23] T. Parr, K. Fisher LL(*): the foundation of the ANTLR parser generator. ACM SIGPLAN Notices **46**(6), 425–436 (2011).

[24] T. Walkowiak, M. Pol The impact of administrator working hours on the reliability of the Centre of Language Technology. Journal of Polish Safety and Reliability Association **6**(1), 167–174 (2017).

[25] A. Radziszewski A tiered CRF tagger for Polish, Intelligent Tools for Building a Scientific Information Platform. Studies in Computational Intelligence **467**, 215–230 (2013).

[26] G. Salton, McM. Gill J. Introduction to modern information retrieval. McGraw-Hill. ISBN 978-0070544840, 1986.

[27] P. Pantel & D. Ravichandran (2004) Automatically Labeling Semantic Classes. In Susan D. Dumais M. & S. Roukos (Eds.) HLT-NAACL 2004: Main Proceedings , Association for Computational Linguistics, 2004, pp. 321-328.

[28] T. Hastie, R. Tibshirani, J. Friedman The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer, New York, 2009.

[29] T. Landauer & S. Dumais (1997) A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition. Psychological Review, 1997, 104, pp. 211-240.

[30] M. Piasecki; S. Szpakowicz & B. Broda (2009) A Wordnet from the Ground Up. Oficyna Wydawnicza Politechniki Wrocławskiej. URL: http://www.dbc.wroc.pl/Content/4220/Piasecki_Wordnet.pdf

[31] M. Eder, J. Rybicki, K. Młynarczyk, M. Oleksy, R. Borys, M. Maryl, M. Piasecki, *1000 Novels Corpus*, CLARIN-PL digital repository, http://hdl.handle.net/11321/312.

[32] L.R. Dice, "Measures of the Amount of Ecologic Association Between Species". Ecology. **26**(3), 297–302 (1945).

[33] T. Sřrensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons". Kongelige Danske Videnskabernes Selskab. **5**(4), 1–34 (1948).

[34] M. Eder Taking stylometry to the limits: benchmark study on 5,281 texts from Patrologia Latina. In: Digital Humanities 2015: Conference Abstracts http://dh2015.org/abstracts (2015).

[35] J.F. Burrows, "Delta": a measure of stylistic difference and a guide to likely authorship. Literary and Linguistic Computing **17**(3), 267–287 (2002).

[36] S. Argamon Interpreting Burrows's Delta: geometric and probabilistic foundations. Literary and Linguistic Computing **23**(2), 131–147 (2008).

[37] P. Gärdenfors, Conceptual Spaces – The Geometry of Thought. The MIT Press, (2000).

[38] Y. Zhao and G. Karypis Hierarchical Clustering Algorithms for Document Datasets. Data Mining and Knowledge Discovery, **10**(2), 1 (2005).

[39] I. Borg, P. Groenen Modern Multidimensional Scaling – Theory and Applications, Springer Series in Statistics, 1997.

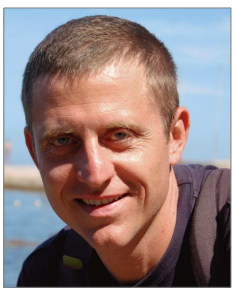
[40] J.P van der L. Maaten, G. Hinton Visualizing data using t-SNE. Journal of Machine Learning Research **9**(Nov), 2431–2456 (2008).

[41] M. Belkin, P. Niyogi Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. Neural Computation **15**(6), 1373–1396 (2003).

[42] A. Przepiórkowski, M. Bańko, R.L. Górski, B. Lewandowska-Tomaszczyk, (2012) editors. Narodowy Korpus Języka Polskiego [Eng.: National Corpus of Polish]. Wydawnictwo Naukowe PWN.

[43] M. Maryl; M. Piasecki, K. Młynarczyk, (2016) Where Close and Distant Readings Meet: Text Clustering Methods in Literary Analysis of Weblog Genres. In M. Eder & J. Rybicki (Eds.) Digital Humanities 2016 Conference Abstracts, Jagiellonian University and Pedagogical University, pp. 273-275.

Maciej Piasecki is Assistant Professor at the Department of Computational Intelligence, Faculty of Computer Science and Management, Wroclaw University of Science and Technology, Poland. He is a coordinator of the G4.19 Language Technology and Computational Linguistics Research Group. Maciej is also the Polish National Coordinator in the CLARIN ERIC (www.clarin.eu). He holds a PhD in Computer Science for work on Natural Language Processing. Maciej has also been a leader of the Polish wordnet project since its very beginning in 2005 till now. His main research areas are: Computational Linguistics, Natural Language Engineering and Human Language Technology, and with the main focus on the topics of automated extraction of the lexico- semantic knowledge from text corpora, semi-automatic wordnet, expansion Distributional Semantics, relational lexical semantics and shallow semantic processing of text. Maciej has also been working on morpho-syntactic processing of Polish (co-author of the first publicly available morpho-syntactic tagger of Polish called TaKIPI that received many applications), Information Extraction, Open Domain Question Answering, formal semantics and Machine Translation. Maciej has also background in software engineering and Human Computer Interaction. Maciej is a member of the Global WordNet Association Board. He has been author and co-author of one book and almost 300 research papers printed in: international journals, books chapters and proceedings of conferences.



Tomasz Walkowiak is Assistant Professor at the Department of Computer Engineering, Faculty of Electronics, Wroclaw University of Science and Technology, Poland. He is editor of 9 monograph books, author and co-author of over 190 papers printed in international journals, books chapters and proceedings of conferences. His research interests include dependability analysis and modelling of complex systems, computer simulation, web based systems, pattern recognition, text mining and open set classification. He is now involved in CLARIN (Common Language Resources and Technology Infrastructure) project that aims to create pan-European research infrastructure intended for the humanities and social sciences. It facilitates work with very large collections of natural language texts.

Maciej Eder is the director of the Institute of Polish Language, Polish Academy of Sciences, and associate professor at the Pedagogical University, Krakow, Poland. He is interested in European literature of the Renaissance and the Baroque, classical heritage in early modern literature, and quantitative approaches to literary works. These include measuring style using statistical methods, authorship attribution based on quantitative measures, as well as "distant reading" methods to analyse dozens (or hundreds) of literary works at a time. His recent work is focused on methodological issues of stylometry, including the problem of a minimal sample size for reliable authorship attribution, the impact of systematic noise on attribution, and so on. Eder is the principal author of the stylometric package Stylo.