COGNITIVE STUDIES | ÉTUDES COGNITIVES, 17
Warsaw 2017

Article No.: 1430
DOI: 10.11649/cs.1430

Citation: Eder, M., Piasecki, M., & Walkowiak, T. (2017). An open stylometric system based on multilevel text analysis. *Cognitive Studies | Études cognitives*, *2017*(17). https://doi.org/10.11649/cs.1430

MACIEJ EDER[1,A], **MACIEJ PIASECKI**[2,B], & **TOMASZ WALKOWIAK**[3,C]

[1] Institute of Polish Language PAS & Pedagogical University of Krakow, Poland
[2] G4.19 Research Group, Wroclaw University of Science & Technology, Poland
[3] Department of Computer Engineering, Wroclaw University of Science & Technology, Poland
[A] maciej.eder@ijp.pan.pl ; [B] maciej.piasecki@pwr.edu.pl ; [C] tomasz.walkowiak@pwr.edu.pl

AN OPEN STYLOMETRIC SYSTEM BASED ON MULTILEVEL TEXT ANALYSIS

Abstract

Stylometric techniques are usually applied to a limited number of typical tasks, such as authorship attribution, genre analysis, or gender studies. However, they could be applied to several tasks beyond this canonical set, if only stylometric tools were more accessible to users from different areas of the humanities and social sciences. This paper presents a general idea, followed by a fully functional prototype of an open stylometric system that facilitates its wide use through to two aspects: technical and research flexibility. The system relies on a server installation combined with a web-based user interface. This frees the user from the necessity of installing any additional software. At the same time, the system offers a variety of ways in which the input texts can be analysed: they include not only the usual lexical level, but also deep-level linguistic features. This enables a range of possible applications, from typical stylometric tasks to the semantic analysis of text documents. The internal architecture of the system relies on several well-known software packages: a collection of language tools (for text pre-processing), Stylo (for stylometric analysis) and Cluto (for text clustering). The paper presents: (1) The idea behind the system from the user's perspective. (2) The architecture of the system, with a focus on data processing. (3) Features for text description. (4) The use of analytical systems such as Stylo and Cluto. The presentation is illustrated with example applications.
Keywords: stylometry; Polish; CLARIN-PL; research infrastructure; language technology

1 Introduction

Stylometry, or the statistical analysis of writing style, aims to investigate text-to-text similarity on different linguistic levels. Originally developed to verify the authorship of anonymous literary works, it was later extended and generalised to assess style differentiation, chronology, genre, author's gender etc. It relies on the assumption that authors have their unique writing habits — sometimes referred to as their "stylistic fingerprint" — that can be pinpointed using, e.g. machine-learning approaches. Classical stylometric approaches are usually focused on very simple linguistic

features that can be automatically retrieved from text documents. These include the frequencies of the most frequent words, occurrences of punctuation marks, the average sentence length, the average word length etc. However, since these style-markers are very effective for authorship attribution and style recognition, they are not suitable for more semantically sensitive analysis. On theoretical grounds, multilevel features based on Natural Language Engineering (NLE) should be more efficient in this area.

Stylometric techniques are known for their high accuracy of text classification, but at the same time they are usually quite difficult to use for an average literary scholar. Presumably, stylometry would be routinely applied in many research tasks in the Humanities, if it were more accessible to researchers with no programming skills. It would seem that implementing some of these methods in an out-of-the-box tool might overcome this drawback. The goal of our work was twofold. Firstly, we wanted to develop a web-based system for stylometry aimed at scholars in the humanities, which does not require installing any software onto local machines, and which makes use of the high-performance capabilities of the server. Secondly, we planned to enlarge the set of standard stylometric features with style-markers referring to various levels of the natural language description and based on NLE methods.

Computing word frequencies is simple in English, but in the case of highly inflected languages, characterised by a large number of possible word forms, one faces the problem of data sparseness. Thus, it might be better first to map the inflected word forms to lemmas, and next to calculate the frequencies of lemmas. The mapping can be performed by using a morpho-syntactic tagger. The tagger tries to automatically recognise the grammatical attributes of the analysed words (e.g. case, gender). Such attributes can be also used as elements of the text document description, e.g. higher frequency of the first person can signal a personal style of writing. Moreover, the documents can be further processed and enriched with the identification of Proper Names, or even with disambiguated word senses (e.g. as recorded in a semantic lexicon). The present paper will analyse the applicability of the aforementioned language tools to document description, for the needs of stylometry and semantic content-based clustering of documents. One needs to be aware, however, that using NLE tools sometimes does not guarantee better classification precision, e.g. syntactic parsers for Polish do not improve the results in authorship recognition tasks, probably because the authors have less freedom of choice with respect to the syntax structures than of vocabulary (lexical level), or because parser errors introduce too much noise to the data.

The workflow supported by our web-based system is as follows. Input documents are processed in parallel. Since the uploaded documents might be in different formats (doc, docx, pdf, html, rtf, txt using various codepages), they are converted to a uniform text format. Next, each text is analysed by a part-of-speech tagger — we use WCRFT2 for Polish (Radziszewski, 2013) — and then it is piped to a name entity recogniser — in this case Liner2 (Marcińczuk, Kocoń, & Janicki, 2013). When the annotation phase has been completed for all the texts, the feature extraction module is launched — using the tool Fextor (Broda et al., 2013). It creates a matrix of features, which is then normalised, weighted or transformed. Finally, the R package Stylo (Eder, Kestemont, & Rybicki, 2013) is used to perform an explanatory analysis, e.g. multidimensional scaling. The results obtained in a graphical format are displayed by the web browser (see Fig. 1). The web interface allows the uploading of input documents from a local machine or from a public repository, provides some options for selecting linguistic features, and options for selecting a grouping algorithm. Apart from the standard procedure, one might want to use Cluto (Zhao, Karypis, & Fayyad, 2005), a well-known clustering tool, to perform the final steps of the analysis. In this case, Cluto replaces the R package Stylo in the text processing workflow and expands the set of clustering methods that can be in used in the analysis. The system in its original form is designed to process Polish. English texts are analysed on the level of word forms only. However, as the feature representation is almost language independent, we plan to extend the workflow with language tools for other languages. Firstly, full support for English will be introduced; other languages will be added successively.

The fully-functional system offers a variety of possible features combined with the rich functi-

onality of the clustering modules. As a result, it can be used as a research tool in stylometric analysis but also for discovering semantic classes in large document collections. Possible further developments of the system will be discussed, e.g. extraction of the descriptive features from text clusters.

The goal of our work is to facilitate broader applications of stylometric methods by:

- constructing an open stylometry system that is also accessible for non-technical users via web-based user interface (henceforth UI),
- equipped with a rich set of features for the extended description of text documents.

The web based interface and the lack of technical requirements facilitates the application of text clustering methods beyond the typical scope of stylometry, e.g. analysis of different types of blogs (Maryl, 2012), recognition of the corpus internal structure, analysis of subgroups and subcultures, etc.

The proposed Open Web-based Stylometric (*WebSty*) system, as well as the enhanced methods for text description discussed below, is focused on processing documents in Polish, as the system has been developed as one of the results of the CLARIN-PL[1] project (Piasecki, 2014), which is a part of the European research infrastructure CLARIN ERIC.[2] However, we aim to separate the text processing modules from the feature extraction and data analysis elements.

2 Barriers and opportunities for stylometry

2.1 Features

Stylometry, also known as computational stylistics, has been used for decades to infer the authorship of anonymous or disputed texts. It relies on the assumption that each author has their unique writing habits, which are subconscious and thus beyond any authorial control. Such a unique authorial profile is usually referred to as "stylistic fingerprint". It is fairly counterintuitive that these authorial fingerprints can be traced in linguistic units rarely associated with style, which include the usage of letter pairs, letter triplets, the co-occurrence of certain syllables, or parts of speech (Stamatatos, 2009; Houvardas & Stamatatos, 2006; Kjell, 1994). The most classical solution, however, introduced by Mosteller and Wallace in their seminal study on the authorship of the *Federalist Papers* (Mosteller & Wallace, 1964), is measuring the usage of a few dozen function words (grammatical words), such as "the", "or", "in", "a", "of", and so forth. Since the function words are at the same time the most frequent tokens in a corpus — no matter which language is taken into consideration — relying on top frequency lexemes became a robust, time-proven, and relatively easy extractable type of style-markers.

The attractiveness of the aforementioned classical solution — i.e. relying on the frequencies of the most frequent words — can be easily explained by the fact that extracting these features is straightforward and computationally very cheap. The whole pre-processing procedure can be completed using a single regular expression applied to the input corpus. Arguably, however, there are many other types of linguistic features that might prove equally effective as robust style-markers. They include: syntax structures, lemmatized words, sequences of parts of speech, named entities (i.e. the usage of proper nouns), the frequencies of particular grammatical categories, and many other similar features that involve sophisticated NLE tools and techniques. Despite their potentially strong discriminative power, however, the NLE-based features are not easy to extract from input texts. They require reasonable computational resources, some additional training data, and, in most cases, advanced programming skills in the users.

The main reason for undertaking the relatively difficult and costly NLE pre-processing tasks for stylometry is a theoretically well-justified assumption that the accuracy of authorial recognition

[1] `www.clarin-pl.eu`
[2] `www.clarin.eu`

will increase significantly (Hirst & Feiguina, 2007). Moreover, in stylometry tasks which go beyond authorship attribution and which aim to trace high-level stylistic layers in input texts, such as genre, gender, register, chronology, and so forth, an extended selection of style-markers might lead to a substantial increase in text classification accuracy. Thus, the aim of the system discussed in this paper is to introduce a variety of different NLE-based features to be used separately or in combination with the classical type of style-markers, namely the most frequent words.

2.2 Multidimensional methods

Particular stylistic profiles, as represented by the frequencies of features, are compared using a variety of multidimensional methods. The reason for their value in text classification is the fact that they aggregate the impact of many features of individually weak discriminating strength. Multidimensional methods can be divided into two groups: explanatory (or unsupervised) machine-learning techniques, supplemented by simple visualizations such as dendrograms or scatterplots, and supervised techniques, said to be very accurate, albeit counterintuitive. The former group includes Principal Components Analysis, Multidimensional Scaling, or Cluster Analysis (Hoover, 2003), while the latter group includes Support Vector Machines (Koppel, Schler, & Argamon, 2009), Naive Bayes Classification (Schaalje, Blades, & Funai, 2013), Nearest Shrunken Centroids (Jockers, Witten, & Criddle, 2008), and so on.

Explanatory or unsupervised methods allow the data to "speak for themselves" in their entirety. An algorithm is used to accommodate the combined differences between the samples into a single plot; the assumption is that in this way, relevant groupings and/or separations are likely to emerge. Such techniques rely on the concept of "distance" between particular stylistic profiles. The complex set of individual frequency differences is transformed into a compact measure of similarity between the samples. There are many possible mathematical transformations that can be used as distance measures (Moisl, 2014); stylometry uses a dozen or so of them.

Machine-learning, or supervised methods, involve two steps of analysis. In the first step, the goal is to divide the input dataset into two subsets:

- a training set containing samples representative for each class,
- a test set containing all of the remaining samples.

The differences between the profiles of the samples in the training set are used to produce a classifier, i.e., a set of rules or an automaton[3] for discriminating stylistic profiles. In the second step, this classifier is used to assign other samples to the classes established in the first step, thus evaluating its accuracy. The entire procedure is repeated several times with different texts selected randomly as the training and the test subsets in order to neutralise any local anomalies in the training data. This procedure is known as k-fold cross-validation, where k stands for the number of parts into which the data set is randomly divided, as well as the number of repetitions. It is routinely used in most supervised classification techniques.

2.3 Limitations

Stylometric methods can be applied to many tasks in the humanities which require grouping of documents according to their properties, finding similarities and differences between single documents and groups of documents, as well as tracing annotated documents over a timeline. Stylometric techniques share many characteristics with methods of semantic text classification that can be used as a basis for semi-automated semantic tagging. This can be very useful in sociology, for example.

The main obstacles to the wider use of stylometry methods are: insufficient programming

[3]They are usually based on statistic analysis.

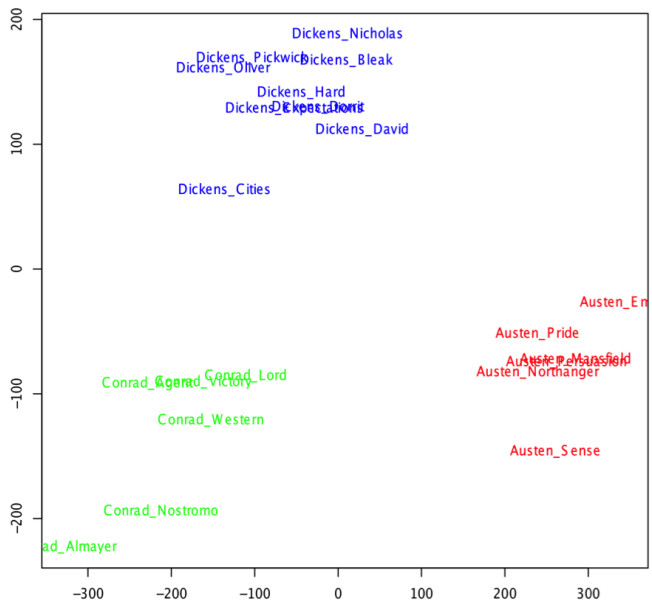

Figure 1: Example of two-dimensional visualisation of texts-clusters.

skills and knowledge of Computational Linguistics (CL), as well as NLE.[4] Concerning the former, many stylometric toolkits require some form of installation and often proficiency in writing simple programs (or scripts). Moreover, they depend on some external packages, e.g. R programming language (R Core Team, 2015) that also have to be installed.

With regards to the natural limitations of the users' knowledge, not every linguistic phenomenon can be expressed formally by CL models, and not every stylometric feature which can be theoretically defined can be extracted by NLE tools in a sufficiently robust way (in terms of precision, coverage, error distribution, processing efficiency etc.). It is also difficult to predict the behaviour of a given feature in a stylometric setup. For instance, perfectly recognised elements of a semantic structure can all belong to a specific class, while other types of elements are neglected by the recognition tool. This causes considerable bias in the statistical data obtained.

Additionally, stylometric systems often assume that all NLE preprocessing is done before a dedicated stylometric software is run. NLE language tools are usually much harder to install and run than most stylometric systems. Combining several language tools into one processing chain is not an easy task at all, due to their interdependencies, e.g. they require an order of application, and depend on the compatibility of data formats. Stylometric systems can also place strong limitations on the meta-data formats produced by language tools. This already complicated picture can become even more complex, if one goes deeper into the details. The main conclusion, however, is already clear. The burden put on non-technical users is too large.

3 Text processing

The basic design of the WebSty system assumes the scenario of an unsupervised authorship attribution, according to which the input documents are automatically clustered into groups that should include texts by the same author, provided that the applied method works as expected. As the methods depend on many hyperparameters and clustering algorithms, we have also considered

[4] Traditionally referred to as Natural Language Processing. However, NLE puts the emphasis on robustness of the developed methods, and their applicability for large-scale tasks.

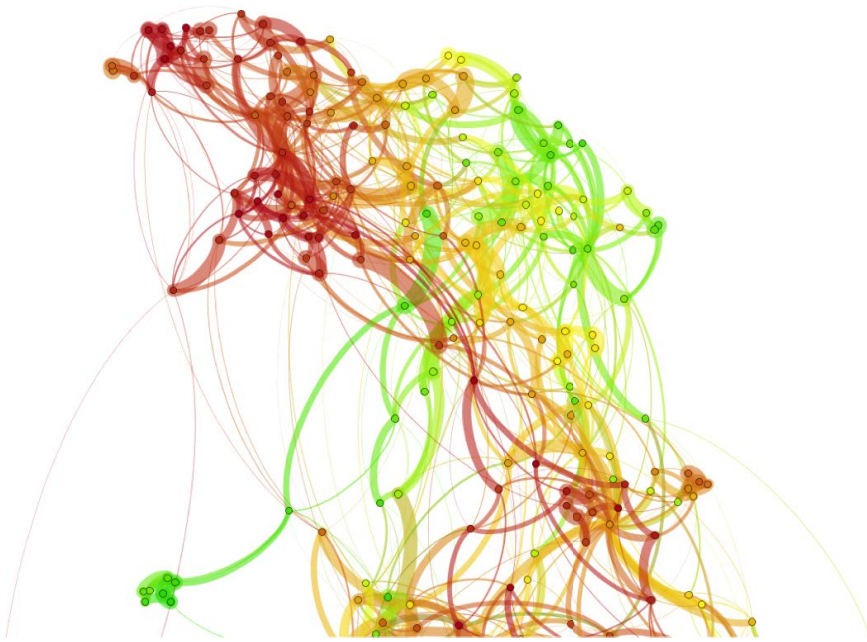

Figure 2: Example of similarity graph for a subset of English texts.

a supervised version of this basic scenario, in which documents of known authorship are processed in order to test the system, select the best method and tune its parameters. Users can experiment with different parameters in order to test how these settings influence the final clusters. This can be done by comparing the obtained clustering with the expected text groups (according to authorship attribution). The supervised version has an educational advantage — it illustrates the basic ideas of stylometry.

This section of the paper describes the text processing pipeline, which may slightly vary depending on the chosen testing scenario. Next, we discuss the means for the implementation of the text processing scheme. The finally selected set is presented and we provide justifications for the decisions that have been made, especially in the area of the NLE: the types of features implemented and those that we decided not to implement. In addition, we try to briefly analyse the research tasks for which the proposed features and processing techniques may prove to be effective.

3.1 Scheme of processing

According to the assumed basic usage scenario, one can distinguish the following main processing steps.

1. Uploading a corpus of documents together with meta-data.
2. Choosing features for the description of documents.
3. Setting the parameters of the analysis.
4. Pre-processing the texts using the chosen language tools.
5. Extracting the features from the pre-processed texts.
6. Calculating feature values.
7. Filtering the features on the basis of their values: this is often combined with transforming the values of features (or even the features themselves) into new abstract features.
8. Performing the main stylometric analysis: clustering or classification.
9. Presenting the results: visualisation or export of data.

3.1.1 Uploading a corpus

Uploading document files into the system is the obvious first step. Several steps have to be taken:

1. Uploading document files in formats delivered by the user.
2. Extracting and decoding the meta-data describing the documents.
3. Converting the content of documents into plain text.

Relevant information about the documents, such as the author, the language, the author's gender, the date of creation, and the place of writing etc. should be expressed in a commonly-accepted meta-data format. However, a common practice is to use only descriptive document file names instead of meta-data, e.g. Stylo (Eder et al., 2013) suggests that the file name should start with a class identifier (an author name or another relevant class label).[5] We have adopted this practice as a simple alternative to proper meta-data.

Nevertheless, as any meta-data format can be much more expressive than simple labels embedded in file names, proper meta-data are preferred. There is a plethora of meta-data formats for documents, hence it is hard to support even the most common ones. Instead, we decided to limit the support to the CMDI[6] (Component Metadata Infrastructure) meta-data format (Broeder et al., 2009; Broeder, Windhouwer, van Uytvanck, Trippel, & Goosen, 2012) which is a standard required by the CLARIN-PL and CLARIN ERIC research infrastructure in general. Fortunately, CMDI is a very flexible format and CLARIN undertook considerable efforts to provide converters from other meta-data formats to CMDI. Moreover, a single CMDI meta-data record itself can consist of a number of components that can express meta-data in other formats. Thus, in a sense, CMDI encompasses most of the existing meta-data formats and it is relatively easy to write new converters if needed.

The assumed web-based UI makes uploading the input files challenging, as web technology is not well suited for the transfer of large data volumes, see Sec. 6.2. This is why only small sets of documents can be uploaded directly via the web-based UI (e.g. up to 30 files and 30 MB in total). For larger data sets, a connection between WebSty and the CLARIN-PL repository, based on the DSpace system, was built. The data set must first be deposited in the repository[7], and only then they can be selected for processing in WebSty, see Sec. 6.2.

Source documents can be written in many different formats, e.g. MS Word files, PDF, HTML, open editor formats, XML, plain texts etc. In order to free users from the technological burden as much as possible, the system provides automated content extraction from different media formats, cf Sec. 6.2. However, automated conversion is always the last resort, as the ways of encoding the text content depend on the file format, and even the way in which it is applied. For instance, in the case of PDF files, text fragments can be physically stored in a very unstructured, or even random, sequence of elements inside the file. A common problem for many formats is that they do not separate text and meta-text, so that most headers, footers, watermarks, page numbers etc. are not filtered out in the converted version. Thus, it is always better to deliver plain text files without any unwanted additions.

3.1.2 Choosing descriptive features and setting up processing parameters

The selection of features describing a document is crucial for the whole processing, and is done by the user. In the case of the classification scenario, the choice can be made automatically by a feature selection algorithm, assuming that enough training data is provided.

A wide range of possible features for describing documents has been defined and implemented in WebSty, see Sec. 3.2. The number of feature types makes the choice rather difficult. Some advice, especially concerning the limitations of the available language tools, is provided in Sec. 3.2. As

[5] The author name can be a class by itself, as can be his/her gender, represented genre or any other type of meta-data.

[6] `http://www.clarin.eu/content/component-metadata`

[7] The user can choose any kind of license, even a very restrictive one.

support for users, we have introduced pre-defined feature sets that are designed to be a reasonable choice for particular types of tasks.

The initial feature values, as acquired from documents, are frequencies.[8] As such data mostly include large amounts of statistical noise (i.e. accidental occurrences) and are dependent on the document length, several algorithms for transforming the values of features were implemented (see Sec. 3.6). However, it was possible to propose relatively good default choices to users. Feature filtering removes unwanted features from the analysis, and sometimes, as a consequence, also the documents which do not contain the relevant features in question. Filters of the type: "minimum 5 occurrences" are quite intuitive, and some good default values can also be proposed.

With regards to the parameters of processing, the main scheme of processing should firstly be based on either *clustering* or *classification*. However, WebSty so far only supports the former. For the clustering process, we use the ready-to-use toolkits Stylo and Cluto. Several more are planned to be included in the future. Clustering parameters are the sum of all the parameters of the toolkits used, and they are discussed in Sec. 4. As the goal of clustering is to find groups of texts with very similar features, the parameters focus on the similarity method used, and the exact way of determining groups of similar documents. As a decision about the clustering parameters requires knowledge of the methods, WebSty offers pre-defined parameter sets for typical tasks. Users can experiment first with the document sets of the known meta-data, in order to choose the parameter setting that seems to work best for the collection of the considered type.

In the case of the planned classification scenario, training-testing data must be provided and the set-up for the learning process must be decided on. As this requires substantial knowledge of algorithms, the only possible option is to provide predefined choices for users, and to make the detailed setting accessible but hidden by default from average users.

3.1.3 Pre-processing texts and calculating feature values

The goal of this step is to convert plain text into an XML format in which single tokens and token groups are annotated with meta-data describing their linguistic properties. The results then form the basis for the extraction of features referring to the linguistic properties of the text (see the detailed description in Sec. 3.2).

Depending on the features selected, a slightly different set of language tools must be run in a sequence. However, this is done fully automatically. User decisions or control are not required.

The features selected by the user are expressed in the form of expressions interpreted by the Fextor system (Broda et al., 2013) for feature extraction. More complex features are expressed with the help of the WCCL language of linguistic constraints (Radziszewski, Wardyński, & Śniatowski, 2011), which describes the expected properties and dependencies between tokens in texts. In order to calculate feature values, Fextor goes sequentially across the linguistically annotated document, runs the implemented features, and for each feature counts how many times a text token or a group of text tokens matches the constraints expressed in the features. Fextor has large expressive power, but this comes at the cost of processing, which is much more complex than simply counting the number of different words. As a result, processing is slower than in the case of collecting trivial features.

3.1.4 Filtering and transforming features

The features' values extracted from texts are organised into a two-dimensional matrix in which rows correspond to documents and columns to different features. A matrix cell $\mathbf{M}[f_i, d_j]$ stores the value of the feature f_i for the document d_j. Initially, the value is simply the frequency of f_i in the document d_j, i.e. how many tokens or token sequences (or even groups) from d_j matched the constraint defined in f_i, e.g. if f_i is "the token is a noun", then the initial value of f_i is the number of nouns in the document.

[8] The number of occurrences of different elements in a document, see Sec. 3.2

Additional filters remove features, i.e. set values of individual features to zero or remove whole columns corresponding to selected features from the matrix. In the case of documents, whole rows are removed, e.g. for documents that are described by too little information. Such filters are applied very rarely.

Transformations change the individual values of features, e.g. Pointwise Mutual Information changes each feature value from a frequency into a value showing the amount of information delivered by the given feature about the document. Transformations which change the whole matrix are also applied, e.g. SVD (Single Value Decomposition) is used to compute new dimensions for the matrix in which the number of columns is much smaller than the original, and to transform the whole matrix into a matrix of the reduced dimensions, e.g. from the initial 50,000 columns (and features) into 200 columns. As a result, a set of new abstract features is created. This type of transformation creates a generalisation over the raw data and can reduce the influence of the noise to some extent.

Filtering and transformations are performed automatically on the basis of the settings provided earlier by the user.

3.1.5 Performing the core stylometric analysis and presenting the results

For the core analysis, the system applies already existing tools, especially Stylo, as well as existing systems for data clustering, e.g. Cluto, as the core processing modules. In the future, ready-to-use classification tools will also be added. Some of these tools provide visualisation of results, e.g. Stylo and Cluto to some extent, while others return data in numerical form and the visualisation must then be implemented in the Web-based UI. Stylo is also a specialised system for clustering and classifying text data. Data clustering systems have different input and output data formats, as well as parameters and ways of setting them. Therefore, each tool for core data processing must be packed into a dedicated module, but the rest of WebSty remains unchanged.

3.2 Features for text description

The frequencies of the most frequent words in the given language are in some mysterious way the most effective features in authorship attribution, according to many research works (Koppel et al., 2009; Stamatatos, 2009; Eder, 2011). Some of the other most reliable style markers are: the frequency of punctuation marks (Baayen, Van Halteren, Neijt, & Tweedie, 2002), the average sentence length, the average word length etc. However, the number of possible features that can be used in stylometry is very large. More advanced feature types require some complex text processing routines, e.g. syntactic parsing. Appropriate tools are not available for all languages and the feature frequencies produced with the help of such tools can be distorted or biased, not only by statistical noise, but also by some systematic errors produced by the language tool, due to its limited accuracy.

Polish is a language with fairly rich inflection and weakly constrained word order. The large number of words forms[9] means that features based on word frequencies can be very misleading, e.g. different forms of the same adjective, noun or verb are counted as separate words. This effect is less visible in the case of the most frequent words, e.g. the 100–500 most frequent, as this set is dominated by non-inflected grammatical classes such as conjuncts and adverb-particles. However, in the case of grammatical classes, counting word forms can generate strong statistical noise. The mapping of word forms onto some abstract representation of word classes, e.g. represented by lemmas — basic morphological forms — is necessary.

We also wanted to explore the space of features beyond word-based ones, i.e. to try to apply more syntactically and semantically informed analysis, based on grammatical classes, structures, lexical meanings, semantic classes etc. However, syntactic analysis is more difficult than in English due to the weakly constrained word order of Polish.

[9]For instance, there are more than 100 possible word forms for most adjectives.

3.3 Morphological

Morphological features refer to the directly observable properties of texts and words, and are the simplest ones to calculate:

- *length* of: documents, paragraphs or sentences,
- *frequency* of:
  - word forms or tokens,
  - punctuation marks,
  - pseudo-suffixes and pseudo-prefixes (last or first several letters).

Extraction of these features only requires a tool for segmenting text into sentences and tokens. Paragraphs can only be read from the meta-data of the document if they are described by annotation. Automated segmentation of text into paragraphs generally expresses too little accuracy (typically up to 50%) and its performance is too domain-dependedng to be seriously considered as a basis for stylometric features. Segmentation can be very rudimentary and based on punctuation marks and blank spaces, but we used much more accurate tools: MACA (Radziszewski & Śniatowski, 2011), for segmenting text into sentences and tokens, and *Morfeusz* (Woliński, 2006), for recognising word form tokens, e.g. an ad-adjectival adjective *biało* 'white' in expressions such as *biało-czerwony* 'white-red'.

The last several letters in Polish word forms contain a great deal of information about the grammatical properties of the form, as well as its possible derivation.[10] Pseudo-prefixes, i.e. the first several letters, mostly express information concerning derivation.

Prefixes and suffixes, as defined in any model of Polish morphology, can be extracted with the help of a morphological analyser, e.g. Morfeusz. This, however, creates ambiguity and in order to solve it, it is necessary to refer to morpho-syntactic tagging, which is the basis for the next group of features.

In a similar fashion, features based on lemmas[11] have been included in the next group, as a proper disambiguation of lemmas provided by the morphological analyser requires the application of the morpho-syntactic tagger.

3.4 Grammatical

The group of grammatical features encompasses features based on the grammatical properties of words and structures. The former can be analysed with a morphosyntactic tagger (henceforth tagger), the latter requires some form of parsing.

In our system, we utilised the WCRFT morpho-syntactic tagger (Radziszewski, 2013) in the version WCRFT2, which displays slightly worse accuracy but is much faster. WCRFT is a robust tool with good accuracy and coverage for practical applications. The features based on tagging encompass the frequency of:

- *lemmas* — assigned to words by the morphological tagger,
- *grammatical classes*,
- *Parts of Speech* — based on grouping grammatical classes into traditional classes,
- combinations of grammatical classes and categories.

Lemmas are included in tags — meta-data elements — assigned to text words by the morphological analyser. WCRFT selects both an appropriate tag and a lemma for a word.

[10] Statistical analysis of the pseudo-suffixes can be found in Piasecki and Radziszewski (2008), the use of pseudosuffixes and prefixes in the recognition of derivational relations we studied in Piasecki, Ramocki, and Maziarz (2012a, 2012b); Piasecki, Ramocki, and Minda (2012).

[11] A lemma is here understood as a selected basic morphological form that represents a set of word forms that differ only in the values of the grammatical categories.

In the case of grammatical classes, we follow the tagset of the Polish National Corpus (Przepiórkowski, Bańko, Górski, & Lewandowska-Tomaszczyk, 2012). The classes express fine-grained division motivated by morphological and distributional properties, e.g. pseudo-past participle (praet), non-past form (fin), ad-adjectival adjective (adja), etc.

The combinations of grammatical classes and categories are more fine-grained subclasses defined over parts of the complex positional tags, e.g. *personal verb uses* can be defined as verbs (several grammatical classes) in the 1st or 2nd person. Such a feature can be used as a signal of the personal character of some writing.

Individual co-occurrences of lemmas and tags can be merged together into n-element sequences. Features based on sequences can provide some limited information about the text structure:

- *Lemma sequences* — representing expressions, potential collocations.
- *Sequences of grammatical classes* — providing information about syntactic structures.

Lemma sequences can show some more frequent and more fixed language expressions, including potential collocations. However, on the level of lemma sequences we do not check if they only represent proper expressions. The morphosyntactic compatibility of words from the occurrences of a given sequence is not tested.

Sequences of grammatical classes express some information about grammatical structures. Such information is very partial, as it is based only on grammatical classes, and many sequences cross the boundaries between different constituents of syntactic structures.

In theory n can be any number, but in practice the number of the observed distinct sequences increases so quickly with the growing n, that only $n = 2$ or $n = 3$ are used in practice:

- $n = 2$ — so called *bigrams*,
- and $n = 3$ — *trigrams*.

Potential syntactic features based on the constituent structure, such as the frequency of some constituent types or sequences, as well as on dependency structures, such as the frequency of selected lexicalised dependency relations, have not been implemented in WebSty. The available parsers for Polish do not provide disambiguation of the possible syntactic analysis, they are not robust enough in terms of accuracy and coverage, nor are they efficient enough to process larger volumes of text.

Lemmatisation based on a morphosyntactic tagger for Polish is reliable enough to use lemmas instead of word forms. Many lemmas (and word forms too) may be too specific for different thematic domains, and features based on them may result in clusters motivated by topics shared by the documents. However, the most frequent lemmas and other grammatical features should express author-depended signal.

3.5 Semantic

Due to the lack of a robust syntactic parser, it is difficult to perform semantic analysis of sentences and longer expressions, as it is usually based on the results of deeper syntactic analysis. Therefore, we concentrated only on features based on the lexical semantics:

- semantic *Proper Name classes*,
- *lexical meanings* (word senses),
- *generalised lexical meanings*,
- *formalised concepts*,
- thematic domains, e.g. from the WordNet Domains set

Proper Names (PN) are very important for the automated extraction of information from texts as they anchor texts in the context of interpretation. However, individual PN occurrences are too specific to express information common for documents of the same style or author.[12] Instead of

[12]However, PNs can be very important for clustering documents sharing the same topics.

counting PN occurrences, we can count the frequencies of PN classes such as first names, surnames, places, and organisations, etc. PN occurrences can be recognised in text and classified with the help of the Named Entity Recogniser, Liner2 (Marcińczuk et al., 2013). Liner2 utilises a very extensive dictionary of PNs as one of the contextual features, but its work and accuracy does not depend on the dictionary.

A quasi-standard representation of lexical meanings in NLE is a wordnet. For processing Polish, one can use plWordNet 3.0 emo, which is a very large wordnet.[13] It provides descriptions for more than 178,000 lemmas with the help of more than 259,000 lexical units[14] grouped into more than 184,000 synsets.[15]

The main means of description are more than 40 lexico-semantic relations for which more than 600,000 relation links have been created. Thus, plWordNet 3.0 represents a near-comprehensive description of the Polish lexical system.

Distinct lexical meanings are represented in plWordNet by synsets. Finding all the synsets per lemma is a straightforward operation, but considering that almost 40% of text words correspond to polysemous lemmas, there are two options: to calculate the frequencies of all the synsets per lemma as separate features, or to identify the appropriate synsets per lemma occurrences. The former solution can create statistical noise, but it can be useful in text classification. The latter depends on the use of the Word Sense Disambiguation (WSD) language tool that maps text words to their word sense, in our case to wordnet synsets. We used an WSD tool called WoSeDon (Kędzia, Piasecki, & Orlińska, 2015) that works on the basis of plWordNet. It provides an accuracy of $\approx 50\%$ for polysemous words in a more difficult test (based on selected polysemous words), and $\approx 64\%$ of accuracy on all polysemous words in randomly selected text samples. This accuracy may seem limited, but it is similar to the typical accuracy achieved for English, and even an inferior version of this WSD tool has already proved to be helpful in improving text semantic clustering (Kędzia, Piasecki, Kocoń, & Indyka-Piasecka, 2014).

Features based on disambiguated text word meanings have been implemented as frequencies of the synsets assigned to text words by the WoSeDon tool. Such features characterize the semantics of documents, but can be also used to look for idiosyncratic tendencies in the use of lexical meanings, or for particular semantic fields that dominate in a document.

Such associations become even more visible if one changes the perspective from individual meanings to some form of generalisation. Wordnet relations of hypernymy and hyponymy, together with similar relations (e.g. type/instance), define a hierarchy of synsets in which hypernyms are more general and hyponyms are more specific. With the help of this hierarchy, one can map individual synsets assigned to text words onto their hypernyms of the n levels up. Thus, n specifies the level of generalisation introduced. In this way, more specific senses are grouped into more general classes and the calculation of the frequencies is done on a more general level. Data sparsity is reduced.

plWordNet hypernymy synsets do not form a single rooted hierarchy, but rather a set of more than one hundred separated individual subhierarchies. This structure is derived from the lexical material according to the linguistic model assumed for plWordNet. However, plWordNet synsets have been mapped onto the formalised Suggested Upper Merged Ontology (SUMO) (Niles & Pease, 2001; Pease, 2011) which takes the form of a single root tree graph. SUMO concepts can be interpreted as abstract semantic classes. By mapping from synsets assigned to text words to SUMO concepts, we can introduce features describing texts on a more general and abstract level. Their values are concept frequencies.

The lexical meanings of different Parts of Speech are described in a wordnet in separate sub-databases. Therefore, there is no direct connection between verbs and nouns from the same semantic domain, such as *mountaineering*. This is a well-known drawback of wordnets. English synsets

[13] The largest world language resource of this type. It is mapped to the Princeton WordNet of English.
[14] I.e. pairs: lemma plus sense identifier.
[15] I.e. sets of near synonyms.

in Princeton WordNets have been automatically grouped into WordNet Domains[16] (Bentivogli, Forner, Magnini, & Pianta, 2004) on the basis of a large corpus. This grouping can be mapped on plWordNet with the help of the inter-lingual links (Rudnicka, Maziarz, Piasecki, & Szpakowicz, 2012; Maziarz, Piasecki, Rudnicka, & Szpakowicz, 2013) and used as a basis for features describing the frequency of the semantic domains in texts. However, the amount of statistical noise included in such features can be significant as the original domains have been defined automatically, and their assignment to text words is mitigated by an WSD tool (i.e. WoSeDon).

3.6 Data processing

The initial feature values are raw frequencies collected from documents. In the case of very frequent lemmas (e.g. those occurring more than 1000 times across many documents) such data can be a reliable basis for further processing. Infrequent features (e.g. those occurring less than 100 times) or features with very skewed distributions (e.g. those with almost all occurrences located in a few documents) can generate accidental associations between documents. The worst combination which can occur is a large number of rare features distributed across many documents (i.e. with a few occurrences per document) together with a minority of reliable, frequent features. In such a case, the majority of rare features results in a blurring of the landscape of the inter-document similarity, with many accidental associations.

Apart from the aforementioned issue, raw frequencies depend on document length and the average frequency of a given word in a given language. Thus, in the 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 which express the relative importance of the occurrences of the given feature. This mapping from the initial values to values expressing the relative significance of feature values is called *weighting*. Several popular weighting methods have been implemented in WebSty. They are based on solutions implemented in Stylo and the SuperMatrix system (Broda & Piasecki, 2008, 2013), which is also used for data transformation and filtering:

- Statistical association measures: χ^2, student test, relative frequencies, Z-score, log likelihood.
- Information Theory: PointWise Mutual Information (PMI) (different versions).
- Heuristics: entropy normalisation, tf.idf (Salton & McGill, 1986), Eder's delta (Eder et al., 2013).

Some weighting methods have an intrinsic ability to filter out non-informative features. For instance, in the case of PMI, all values smaller than or equal to zero are discarded from further processing, as they indicate a lack of association between a given feature and the document in which it has occurred. For most weighting methods, some thresholds can be defined experimentally for the minimal values that seem to be sufficiently informative. However, in the case of very infrequent features most weighting methods do not produce satisfactory results. For example, PMI overestimates the importance of infrequent co-occurrences of features, which usually leads to a statistical error that is unacceptably large. Thus, in WebSty the weighting methods are supplemented with several simple filters on:

- the minimal number of occurrences of a feature in the whole collection,
- the minimal number of occurrences of a feature in the given document,
- the minimal number of co-occurrences of a document and a feature.

Filters can also be applied to eliminate particular types of features, e.g. specified lemmas (a stop word list), selected punctuation marks, or sequences of grammatical classes (e.g. involving Proper Names).

Finally, several transformations of the whole space of feature vectors are proposed in the literature. They do not change individual values, but instead transform the whole matrix of documents vs. features into a new space, in which the number of the matrix columns (i.e. new features) is

[16] `http://wndomains.fbk.eu/index.html`

smaller. For instance, Single Value Decomposition (SVD) is used to calculate a new set of features, the number of which is radically reduced, e.g. from 50,000 to 200. The new vectors provide some kind of generalisation in a way that emphasises the most important similarities and differences between documents. Accidental associations are usually reduced after SVD application. Consequently, the level of statistical noise should be lower. However, SVD introduces a kind of generalisation of the document description that goes too far for many tasks, and does so in such a way that some distinctions that are of interest disappear. Additionally, the new features calculated by SVD are abstract and do not have any intuitive interpretation.

The matrix transformation methods are usually used in combination with feature values transformations and filtering.

4 Analysis

After the feature values have been collected, transformed and filtered, the core processing can begin. In WebSty we tried to use already existing solutions for this purpose, namely toolkits for stylometry — Stylo (Eder et al., 2013), and data clustering — Cluto (Zhao et al., 2005). According to the assumed basic scenario, the analysis starts with the calculation of the similarities between documents represented by feature vectors. Next, the vectors are grouped according to their similarity, and finally the identified clusters are presented.

4.1 Similarity measures

There are dozens of different methods for calculating the similarity between data vectors. For WebSty, we selected those that are sufficiently good for textual data (Zhao et al., 2005; Broda &Piasecki, 2008; Eder et al., 2013):

- Similarity measures: Jacquard, Dice, Cosine measure, correlation coefficient from CLUTO (Karypis, 2003).
- Distance measures: Euclidean measure, Manhattan measure, Canberra Measure.
- Delta measure in its different varieties, including Burrows's Delta (Burrows, 2002), Argamon's Linear Delta (Argamon, 2008), and Eder's Delta (Eder et al., 2013).

The Dice and Jacquard measures are based on calculating the ratio of the features weights that are common for two document profiles in relation to the joint set of features from both profiles.

In the Cosine measure, the cosine of the angle between two vectors is calculated. Its main advantage is the fact that the angle is taken into account, not the lengths of the vector. Thus, the cosine measure provides some kind of data normalisation in relation to the documents. This simple measure performs surprisingly well in many applications.

Both the Manhattan and Euclidean measures are well-known and widely-used methods of computing distance between vectors of numerical values. Despite being very sensitive to the length of input vectors and — particularly — to the imbalance between frequent and rare features, they can be very accurate when applied to normalised datasets. Burrows's Delta distance metric (Burrows, 2002), which has attracted a good deal of attention among stylometric researchers (Hoover, 2004a, 2004b; Argamon, 2008), relies on the Manhattan distance combined with a Z-score normalisation of the input dataset.

Eder's Delta is a modification of the aforementioned Burrows's measure. It slightly increases the weights of frequent features and rescales less frequent ones in order to suppress the discriminative strength of some random infrequent features. It was designed to be used with highly inflected languages.

A systematic comparison of all the provided measures, however, might lead to some new conclusions about their performance.

4.2 Clustering

There are two basic schemes of clustering: agglomerative and flat clustering. In addition, Cluto enables the combination of flat clustering with a hierarchy of clusters built in an agglomerative way.

In *agglomerative clustering*, in each step the two most similar clusters are found and merged into a new cluster. This "bottom-up" process starts with an initial set of single clusters including one document each, so that initially 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 placed at the bottom, and all of the clustering decisions between documents and clusters are represented by links (see the example in Fig. 13). Methods of agglomerative clustering differ in the way that they represent clusters and calculate similarities between clusters.

Flat clustering can be based on partitioning larger clusters of documents into smaller ones, or grouping documents around a selected number of documents or some abstract points in the space. Cluto supports the first approach and implements several variants, including mixed approaches in which the initial flat partitioning-based clustering is used as an input to the agglomerative clustering, or in which the results of the initial clustering are used to enhance the information in the profiles of the documents.

The clustering process is controlled by two parameters: the distance/similarity measure, and the clustering criterion function. Clustering criterion functions are different for different clustering methods. For example, in agglomerative clustering they define the way of computing similarities of clusters on the basis of the similarity measure for documents. In the case of flat clustering, the clustering criterion functions define the properties of the clusters that should be preferred by the algorithm, e.g. the minimal distances between documents inside clusters, the maximal distances between clusters, etc.

Selecting the clustering method and setting its parameters is a complicated issue. Mostly it is done by tuning a method on a training set. WebSty provides some ready to use defaults in the UI that should provide reasonably good results for typical tasks.

4.3 Presentation of the results

The algorithm assigns documents to their clusters, which can be interpreted as identifying relations between documents, such as whether they have been written by the same author. The results can be downloaded in a numerical form or visualised in the web-based UI.

In the first case, Cluto generates text output files that include information about the clustering method used and the input data, the assignment of objects to clusters (including hierarchical clusters presented in a textual form) and the values of various metrics calculated for the clusters, such as purity or entropy (cf. Karypis, 2003).

Stylo does not really differ from Cluto, since it produces a number of variables that are saved into output files, as well as a final plot. In its current version, Stylo produces a matrix of features' frequencies, a list of features used, and some of the parameters that were applied. In newer versions, a rich set of variables will be saved to output files.

The visualisation of the results depends to a very large extent on the toolkits used for the core processing, i.e. Stylo and Cluto. The former tool provides a rich set of means for visualisation: dendrograms (for Cluster Analysis), and different types of scatterplots (for Principal Components Analysis and Multidi-mensional Scaling).

Cluto produces dendrograms, matrix representations of clusters, and combinations of both. In the matrix representation of flat clusters, documents belonging to one cluster are placed in adjacent rows, while the columns correspond to the features that have been identified by the clustering algorithm as the most significant for defining the clusters. The matrix cells are filled with colour whose intensity represents the specificity of the given feature for the given cluster. However, the method used for identifying characteristic features and calculating their importance

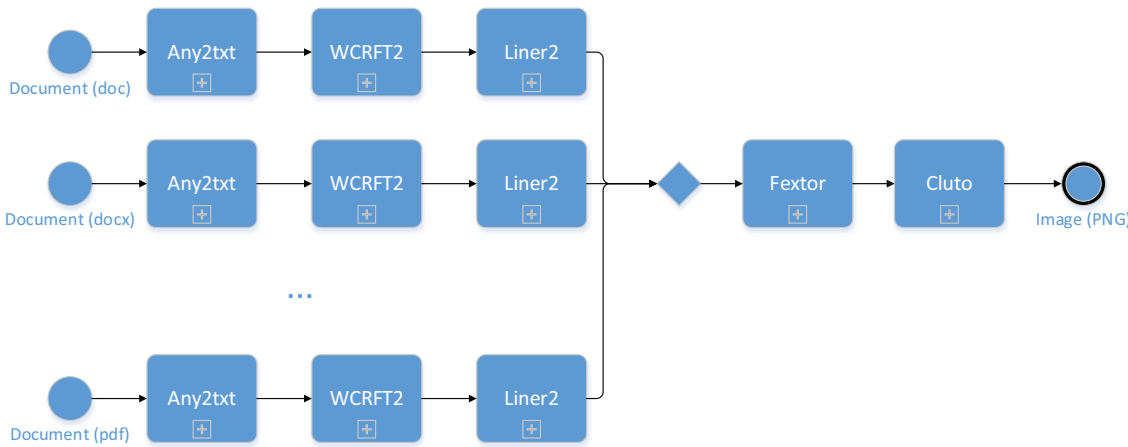

Figure 3: Stylometry workflow.

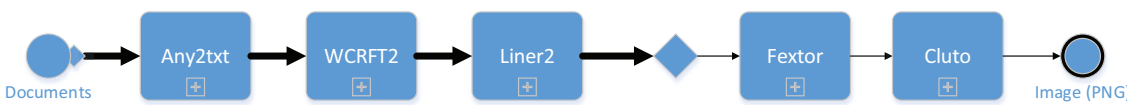

Figure 4: Stylometry workflow in simplified form.

do not seem to be well suited to the needs of stylometry.

5 Language processing workflow

A text analysis task usually requires running a sequence of language tools. For simple applications, such as counting the number of word or PN occurrences, a single sequence of tools is enough. However, for more sophisticated tasks, like text clustering, the process requires complex workflows.

The graphical representation of the workflow of the stylometry system discussed here is presented in Fig. 3. Since the uploaded documents might be in different formats (doc, docx, pdf, html, rtf, txt using various codepages), they are converted to a uniform text format. This step

```
<lpmn>
  <source id="1">
    <file name="E.Chesterton The invisible man 1926.txt">...</file>
  ...
  </source>
  <activity name="any2txt" id="2" source="1"/>
  <activity name="wcrft2" id="3" source="2" />
  <activity name="liner2" id="4" source="3" options="{'model':'5nam'}"/>
  <agregate name="cluto.png" type="dir" id="5" source="4"/>
  <activity name="fextor2" id="6" source="5"  options='{"features":"..."}'/>
  <activity name="cluto" id="7" source="61"  options='...'/>
  <output id="output" source="7"/>
</lpmn>
```

Figure 5: Stylometry workflow in LPMN.

in the text processing is denoted in Fig. 3 by the any2txt boxes, and is done with the use of the Apache Tika[17] toolkit. Next, each text is analysed by a part-of-speech tagger — we use WCRFT2 (Radziszewski, 2013) for Polish — and then it is piped to a Named Entity Recogniser — in this case Liner2 (Marcińczuk et al., 2013). When the annotation phase is completed for all the texts, the feature extraction module is run using the Fextor tool (Broda et al., 2013)). Finally, the Cluto (Zhao et al., 2005) or Stylo (Eder et al., 2013) package is used to perform the data clustering.

A part of the text analysis process could be done in parallel. Each input document could be processed by any2txt, WCRFT2 and Liner2 in parallel, as can be seen in Fig. 3. Of course, other types of text analysis (e.g. supervised learning) may require different workflows. However, some common elements in different language processing workflows can be identified. Therefore, a special Language Processing Modelling Notation (LPMN) was developed for defining workflows in a way accessible to designers of language processing applications. LPMN was inspired by the Business Processing Modelling Notation (BPMN)[18] (Allweyer, 2010) used in modelling information systems. LPMN enables the definition of the functionality of complex language tools by combining simple ones.

The language processing workflow defined in the LPMN consists of: sources, activities, outputs, gate-ways and sequence flows. A source (represented with a circle in the graphical notation, see Fig. 3) denotes an input file to the language processing system. It could be a text input for a language tool, or a set of parameters (for example a list of lexemes). An activity (a rounded-corner rectangle in the graphical notation) represents a langue processing tool. It is defined by a tool name and a set of parameters. Each activity has one input and one output (it could be a single file or a folder). A gateway is represented by a diamond shape and determines the forking and merging of processing paths. There are two kinds of gateways: parallel (used to create parallel paths, with one input and many outputs) and aggregate (used to join parallel paths with many inputs and one output). An output represents the result of a process. It is represented by a circle with a bold border. A sequence flow is encoded by a solid line and arrowhead, and shows in which order the activities are performed. It connects the other elements of the workflow.

To simplify the notation and to generalise the workflow for any number of input files, it is possible to merge identical parallel paths into one path, as presented in Fig. 4. The additional diagonal box after the source indicates that it will start parallel processing, and the thick lines for a sequence flow indicate that the processing is done for a set of input files and that it can be done in parallel.

The LPMN can be represented in XML format to allow for automated processing of the workflow. An example of an XML file with a workflow for a stylometry task is listed in Fig. 5.

LPMN has not been used in WebSty yet, but we plan to apply it as a solution for the easy adaptation of the system to new tasks and new types of features. This facility can also be accessed by more advanced users.

6 Web-based stylometric system

6.1 Infrastructure for natural language processing

The practical usage of LPMN requires an engine that enables the automated execution of the workflow. This is why we have developed an open access, scalable, and highly available infrastructure with various types of software interfaces. The aim of the infrastructure was to build a set of language processing applications dedicated to research in the Humanities and Social Sciences.

In order to guarantee the high usability of the proposed solution, the Web-Oriented Architecture (WOA) (Thies & Vossen, 2008) paradigm was used, whose aim is to build systems consisting

[17] `http://tika.apache.org/`
[18] `http://www.bpmn.org/`

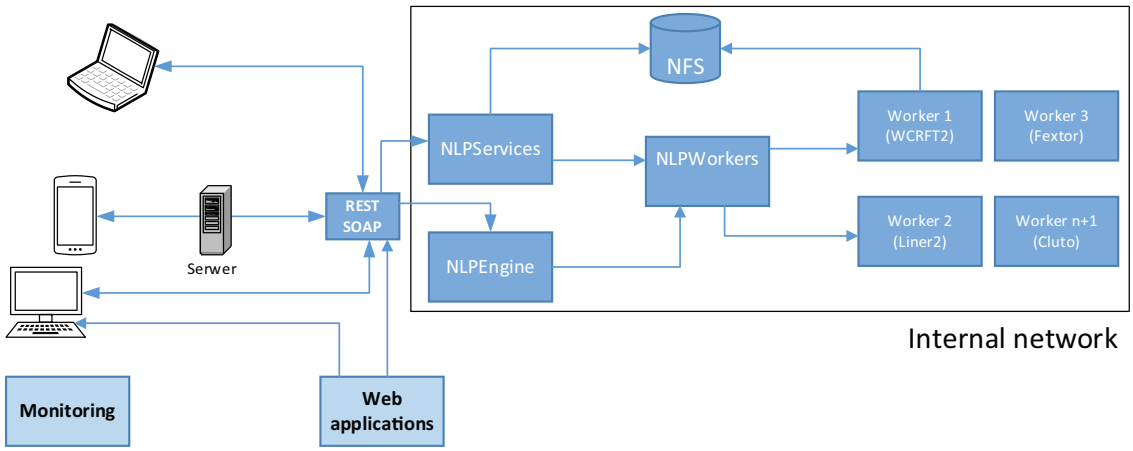

Figure 6: System architecture.

of modular, distributable, sharable and loosely coupled components. WOA follows the SOA (Service Oriented Architecture) paradigm (Josuttis, 2007) and views the entire system as consisting of resources accessed via the representational state transfer (REST) (Richardson & Ruby, 2007). The system architecture is presented in Fig. 6.

The core of the system (NLPTasker) consists of a simple asynchronous REST service, task queues, data storage and a set of workers. The workers run the language, machine learning and visualisation tools. Each worker collects a task from a queue, loads data from the data storage, processes them and returns the results to the data storage. The workers and the queue system allow for the effective scaling of the infrastructure. Additional service (NLPServices) grants access from the Internet. It works as a proxy for the core system, delivering a large set of different APIs. Different techniques for accessing the infrastructure, including synchronous and asynchronous services, SOAP and REST, as well as XML and JSON, are available. Such an approach facilitates easy integration with almost any kind of application. Moreover, the engine for running workflows described in LPMN was developed. It enables the processing of a large corpus of text in a batch-like mode.

To achieve the high availability requirements, the system was deployed on a scalable hardware and software architecture that can be easily optimised to deliver high performance. The hardware consists of eight Cisco UCS B-Series Blade Servers based on Intel® Xeon® processors E5 product families. The servers are connected by a fast fibre channel connection with highly scalable midrange virtual storage, designed to consolidate workloads into a single system for simplicity of management (the IBM Storwize V7000). XEN Citrix, creating a private cloud, controls each server. It makes the virtual infrastructure management more convenient and efficient, since the operating systems are independent of the hardware. Each language tool is deployed on a separate virtual machine. Therefore, it is easy to scale up the system simply by duplicating the virtual machines as a reaction to a high number of requests for a given type of language tool.

The infrastructure is monitored on different levels, starting from hardware monitoring, through to virtual machines, queues and the processing time of each worker.

The first version of the system presented here is focused on the Polish language, but it is flexible enough to be extended in the future to other languages as well.

6.2 Web-based application

Language tools are often hard to install and integrate, since they are developed with different technologies. In addition, the processing of large texts requires huge computational power. The-

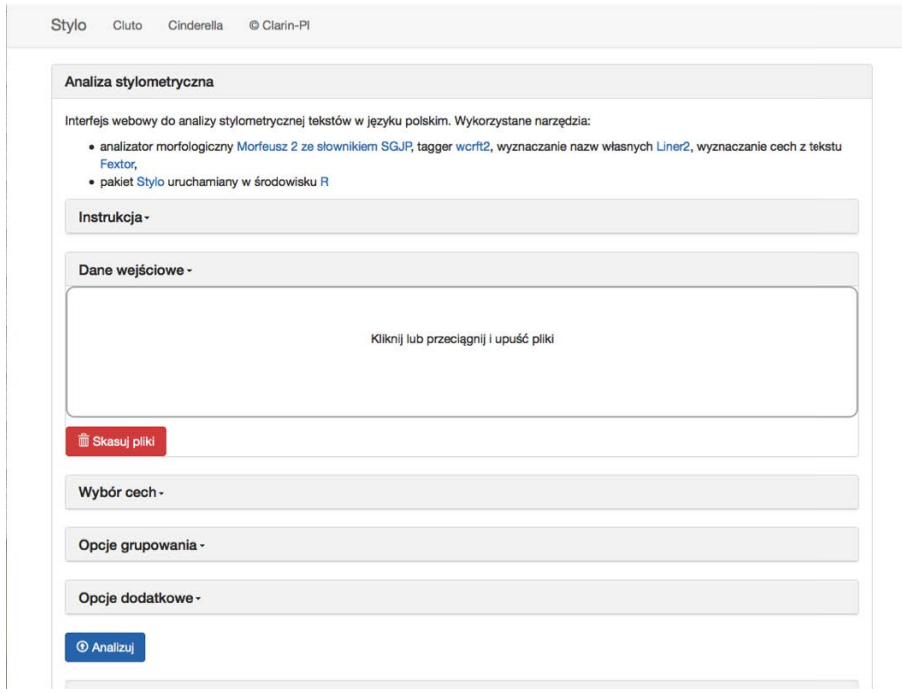

Figure 7: User interface of the open web-based stylometric system of CLARIN-PL.

refore, the infrastructure described in the previous section was used as the basis for building web-based applications for stylometry targeted at scholars in the Humanities and Social Sciences.

These applications were developed in pure HTML5 and JavaScript technology, using REST web service to run and control the language processing workflow on the server side. Two kinds of clustering tools, used to perform the final steps of the analysis, can be used: The R package Stylo[19] (Eder et al., 2013) or Cluto[20] (Zhao et al., 2005).

The graphical UI is presented in Fig. 7. Firstly, a user must select documents for further processing. This can be done by uploading input documents from a local machine (Fig. 8) or by selecting a corpus (Fig. 9) from the CLARIN-PL public repository based on the DSpace system.[21]

Next, the user selects a feature set by putting marks into checkboxes in the feature selection tab (Fig. 10). In the next tab, the user can decide on the parameters for the selected clustering algorithm (Fig. 11 options for Stylo). To help the user to understand the consequences of the decisions, a set of predefined options for different analysis types (classical and extended authorship, analysis of grammatical style and semantic likelihood) is provided (Fig. 12). Finally, the clustering results are displayed in a graphical form (Fig. 13). The Web-based interface, the lack of requirement to install any software on local machines, and access to a computing cluster make for a useful and accessible tool.

7 Applications

The WebSty system can be applied to all stylometric tasks that fit into the clustering scenario. For the input set of documents of known authors, one can use different clustering algorithms which search for groups of similar and dissimilar documents. By testing different sets of selected features,

[19] `http://ws.clarin-pl.eu/demo/stylo2.html`
[20] `http://ws.clarin-pl.eu/demo/cluto2.html`
[21] `http://clarin-pl.eu/dspace`

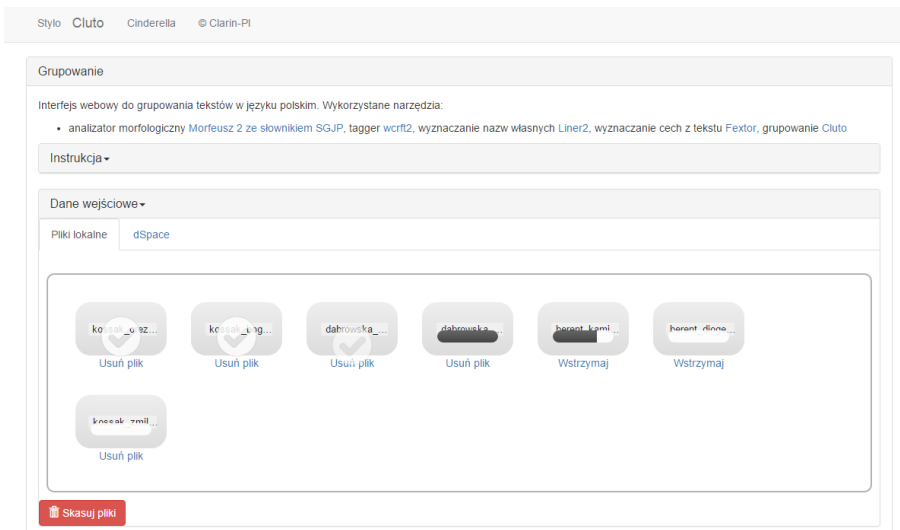

Figure 8: User interface — upload from local machine.

Figure 9: User interface — selecting a corpus from public repository.

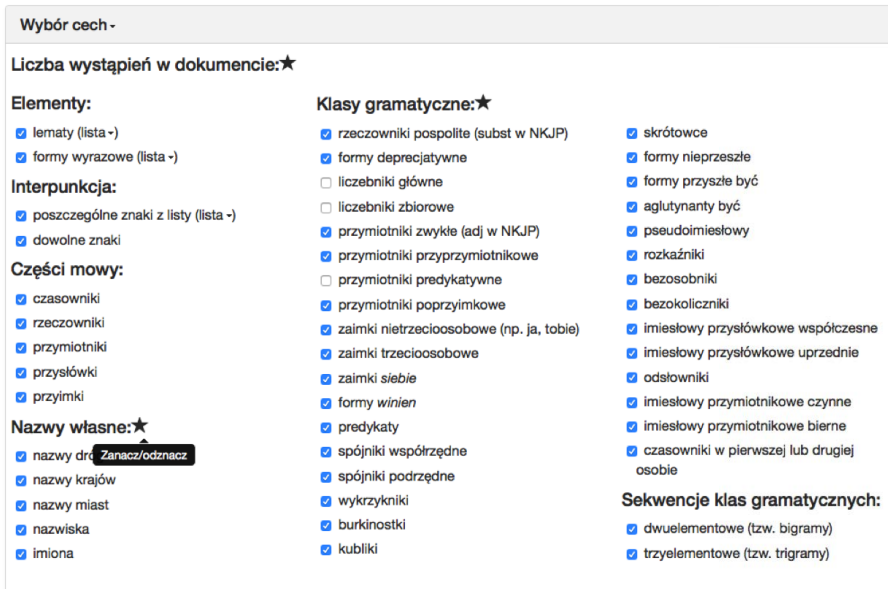

Figure 10: Selection of features in the open web-based stylometric system of CLARIN-PL.

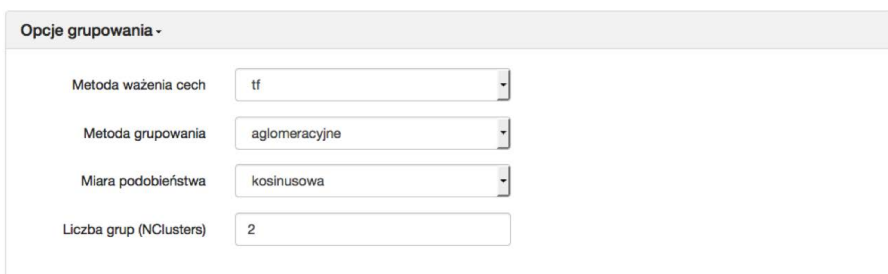

Figure 11: Choice of the analysis parameters in the open web-based stylometric system of CLARIN-PL.

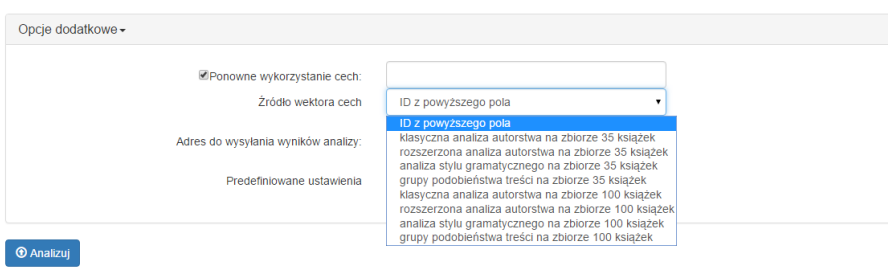

Figure 12: User interface — selecting of predefinied analysis types.

one can analyse the nature of the similarities and differences between documents and groups of documents. In order to facilitate such analysis, we plan to expand WebSty with functions for extracting features that are characteristic of the identified clusters of documents. The extracted characteristic features can illustrate the clusters.

Similar analysis can be applied to documents described not (or not only) by authors, but by other meta-data attributes such as origin, gender, age etc.

In the case of documents of unknown authorship or unknown meta-data characteristics, WebSty can be used to study possible similarities between documents and groups of documents. Feature settings tested on example sets of documents with known meta-data characteristics can be used for this purpose. Automatically identified groups of documents can later be used as sub-corpora for further research. Both types of experiments have been performed on a collection of blogs studied in Maryl (2012).

Similar analysis can be performed on sections of a large book, when it is suspected that different parts were written by different authors or in different periods.

Clustering-based methods can also be used to trace changes in language over time. The extraction of characteristic features from the identified clusters can be especially interesting. Researchers in "stylochronometry", for instance, might be interested in clustering the texts according to their dates of composition, on the basis of stylistic aspects (Stamou, 2008; Juola, 2007).

Perhaps even more interesting, particularly from the point of view of literary studies, is the analysis of large amounts of textual data at one time. This can range from a few dozen novels to thousands of literary texts, assessed in the theoretical framework of "distant reading", or "macroanalysis" (Jockers, 2013). The assumption behind such a large-scale approach is that a massive analysis might reveal literary phenomena that have been overlooked by traditional critical studies. Our system is particularly suitable for performing such computation-intensive tasks, as it can outperform any desktop stylometric system.

In the Social Sciences, the system can be used to search for similarities and differences between texts from a more semantic-oriented perspective. In some applications, it could be very useful to look into specific meanings and concepts that are characteristic for particular authors, groups of authors, or which are used to describe or to refer to particular persons or situations.

The results of clustering, as well as the comparison of the results of several different clustering algorithms, can be used to look for outliers, i.e. documents that are significantly different from the others.

8 Further development and research

WebSty is a fully functional system and it proves the feasibility of web-based, open stylometric systems. Nevertheless, there are many unresolved issues which are open to further research.

There are still options for Stylo that cannot be set via the WebSty UI. More flexibility in setting features, especially for Stylo, should be introduced. Fextor enables the definition of complex features based on lexico-morpho-syntactic patterns, but this possibility is still not open to users.

Support for processing large collections of documents has been introduced. The collections can be described by meta-data in CMDI format, but the flexibility of the use of meta-date needs improving.

A lot remains to be done in improving the efficiency of the system, and the use of distributed and parallel processing. In the present version, values calculated for sets of features can be stored and re-used with different clustering algorithms.[22] However, if a single feature is removed or added to this set, the whole costly feature extraction process must be repeated. A more flexible mechanism of caching feature values is required.

[22] CLARIN-PL repository enables also storing the results of pre-processing of documents with the help of language tools.

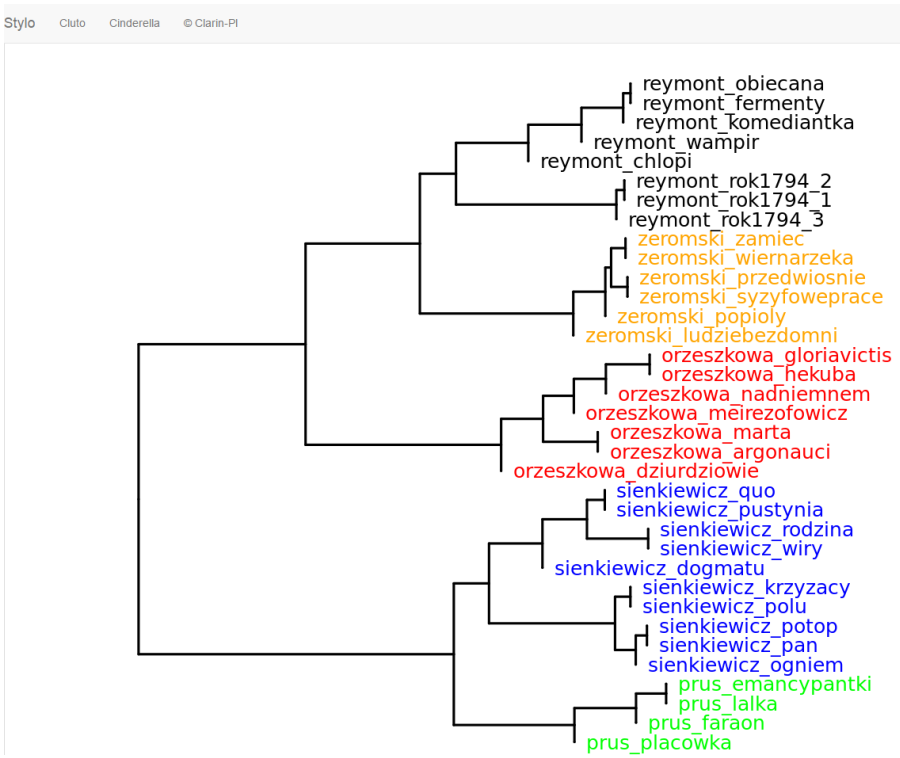

Figure 13: An example of the application of the system to the authorship attribution analysis.

In addition to the clustering-based analysis, a process based on classification should be added. In a classification scheme, users provide a testing-training collection of documents described by meta-data, e.g. including authorship. The WebSty system automatically selects a feature set and trains classifiers on the basis of the provided data set. For example, a classifier recognises a pair of documents as sharing some meta-data attribute, e.g. the author. Finally, the trained classifier is used to process and automatically describe, with meta-data, documents of unknown description.

The classification-based processing scheme can also be applied to semantic text analysis, also known as semantic tagging. As a result, documents or document fragments are automatically described by the user defined tags provided in the training data.

References

Allweyer, T. (2010). *BPMN 2.0: Introduction to the standard for business process modelling*. Norderstedt: Books on Demand.

Argamon, S. (2008). Interpreting Burrows's Delta: Geometric and probabilistic foundations. *Literary and Linguistic Computing*, *23*(2), 131–147. `https://doi.org/10.1093/llc/fqn003`

Baayen, H., Van Halteren, H., Neijt, A., & Tweedie, F. (2002). An experiment in authorship attribution. In *Proceedings of JADT 2002* (pp. 29–37). St. Malo: University de Rennes.

Bentivogli, L., Forner, P., Magnini, B., & Pianta, E. (2004). Revising wordnet domains hierarchy: Semantics, coverage, and balancing. In *COLING 2004 Workshop on Multilingual Linguistic Resources Geneva, Switzerland, August 28* (pp. 101–108). `https://doi.org/10.3115/1706238.1706254`

Broda, B., & Piasecki, M. (2008). SuperMatrix: A general tool for lexical semantic knowledge acquisition. In G. Demenko, K. Jassem, & S. Szpakowicz (Eds.), *Speech and language technology* (Vol. 11, pp. 239–254). Polish Phonetics Assocation. (The first version was published in the Proceedings of the International Multiconference on Computer Science and Information Technology — 3rd International

Symposium Advances in Artificial Intelligence and Applications (AAIA'08)).

Broda, B., & Piasecki, M. (2013). Parallel, massive processing in SuperMatrix: A general tool for distributional semantic analysis of corpora. *International Journal of Data Mining, Modelling and Management*, *5*(1), 1–19. https://doi.org/10.1504/IJDMMM.2013.051924

Broda, B., Kędzia, P., Marcińczuk, M., Radziszewski, A., Ramocki, R., & Wardyński, A. (2013). Fextor: A feature extraction framework for natural language processing: A case study in word sense disambiguation, relation recognition and anaphora resolution. In A. Przepiórkowski, M. Piasecki, K. Jassem, & P. Fuglewicz (Eds.), *Computational linguistics: Applications* (pp. 41–62). Berlin: Springer. (*Studies in Computational Intelligence*, 458).

Broeder, D., Gaiffe, B., Gavrilidou, M., Hinrichs, E., Lemnitzer, L., van Uytvanck, D., Witt, A., & Wittenburg, P. (2009). Registry requirements metadata infrastructure for language resources and technology. Technical Report CLARIN-2008-5, Consortium CLARIN. http://hdl.handle.net/1839/00-DOCS.CLARIN.EU-33

Broeder, D., Windhouwer, M., van Uytvanck, D., Trippel, T., & Goosen, T. (2012). CMDI: A component metadata infrastructure. In V. Arranz, D. Broeder, B. Gaiffe, M. Gavrilidou, M. Monachini, & T. Trippel (Eds.), *Proceedings of the Workshop on Describing LRs with Metadata: Towards Flexibility and Interoperability in the Documentation of LR. LREC'2012* (pp. 1–4).

Burrows, J. (2002). 'Delta': A measure of stylistic difference and a guide to likely authorship. *Literary and Linguistic Computing*, *17*(3), 267–287. https://doi.org/10.1093/llc/17.3.267

Eder, M. (2011). Style-markers in authorship attribution: A cross-language study of the authorial fingerprint. *Studies in Polish Linguistics*, *6*, 99–114.

Eder, M., Kestemont, M., & Rybicki, J. (2013). Stylometry with R: A suite of tools. In *Digital Humanities 2013: Conference abstracts* (pp. 487–89). Lincoln, NE: University of Nebraska-Lincoln.

Hirst, G., & Feiguina, O. (2007). Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*, *22*(4), 405–417. https://doi.org/10.1093/llc/fqm023

Hoover, D. L. (2003). Multivariate analysis and the study of style variation. *Literary and Linguistic Computing*, *18*(4), 341–360. https://doi.org/10.1093/llc/18.4.341

Hoover, D. L. (2004a). Delta Prime? *Literary and Linguistic Computing*, *19*(4), 477–495. https://doi.org/10.1093/llc/19.4.477

Hoover, D. L. (2004b). Testing Burrows's Delta. *Literary and Linguistic Computing*, *19*(4), 453–475. https://doi.org/10.1093/llc/19.4.453

Houvardas, J., & Stamatatos, E. (2006). N-gram feature selection for authorship identification. In J. Euzenat & J. Domingue (Eds.), *Artificial intelligence: Methodology, systems, and applications* (pp. 77–86). Berlin: Springer. (*Lecture Notes in Computer Science*, 4183).

Jockers, M. L. (2013). *Macroanalysis: Digital methods and literary history*. Urbana: University of Illinois Press. (*Topics in the Digital Humanities*).

Jockers, M. L., Witten, D. M., & Criddle, C. S. (2008). Reassessing authorship of the Book of Mormon using delta and nearest shrunken centroid classification. *Literary and Linguistic Computing*, *23*(4), 465–491. https://doi.org/10.1093/llc/fqn040

Josuttis, N. M. (2007). *SOA in practice: The art of distributed system design*. Beijing: O'Reilly Media.

Juola, P. (2007). Becoming Jack London. *Journal of Quantitative Linguistics*, *14*(2–3), 145–147. https://doi.org/10.1080/09296170701378957

Karypis, G. (2003). Cluto — a clustering toolkit release 2.1.1. Technical Report 02-017, University of Minnesota, Department of Computer Science, Minneapolis, MN 55455, USA, November 28.

Kędzia, P., Piasecki, M., & Orlińska, M. (2015). Word sense disambiguation based on large scale Polish CLARIN heterogeneous lexical resources. *Cognitive Studies | Études cognitives*, *2015*(15), 269–292. https://doi.org/10.11649/cs.2015.019

Kędzia, P., Piasecki, M., Kocoń, J., & Indyka-Piasecka, A. (2014). Distributionally extended network-based word sense disambiguation in semantic clustering of Polish texts. *IERI Procedia*, *10*, 38–44. https://doi.org/10.1016/j.ieri.2014.09.073

Kjell, B. (1994). Discrimination of authorship using visualization. *Information Processing & Management*, *30*(1), 141–150. https://doi.org/10.1016/0306-4573(94)90029-9

Koppel, M., Schler, J., & Argamon, S. (2009). Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, *60*(1), 9–26. https://doi.org/10.1002/asi.20961

Marcińczuk, M., Kocoń, J., & Janicki, M. (2013). Liner2 — a customizable framework for proper names recognition for Polish. In *Intelligent tools for building a scientific information platform* (pp. 231–253). Berlin: Springer. (*Studies in Computational Intelligence*, 467).

Maryl, M. (2012). Kim jest pisarz (w internecie?). *Teksty Drugie*, *2012*(6), 77–100.

Maziarz, M., Piasecki, M., Rudnicka, E., & Szpakowicz, S. (2013). Beyond the transfer-and-merge Wordnet construction: plWordNet and a comparison with WordNet. In *Proc. of the International Conference Recent Advances in Natural Language Processing RANLP 2013* (pp. 443–452). Hissar, Bulgaria. INCOMA Ltd.

Moisl, H. (2014). *Cluster analysis for corpus linguistics*. Berlin: Mouton de Gruyter.

Mosteller, F., & Wallace, D. (1964). *Inference and disputed authorship: The Federalist*. Stanford: CSLI Publications.

Niles, I., & Pease, A. (2001). Towards a standard upper ontology. In *Proceedings of the International Conference on Formal Ontology in Information Systems — Volume 2001*, FOIS '01 (pp. 2–9). New York, NY: ACM.

Pease, A. (2011). *Ontology: A practical guide*. Angwin, CA: Articulate Software Press.

Piasecki, M. (2014). User-driven language technology infrastructure — the case of Clarin-PL. In *Proceedings of the Ninth Language Technologies Conference*. Ljubljana, Slovenia.

Piasecki, M., & Radziszewski, A. (2008). Morphological prediction for Polish by a statistical *A Tergo* index. *Systems Science*, *34*(4), 7–17.

Piasecki, M., Ramocki, R., & Maziarz, M. (2012a). Automated generation of derivative relations in the Wordnet expansion perspective. In C. Fellbaum & P. Vossen (Eds.), *Proceedings of 6th International Global Wordnet Conference* (pp. 273–280). Matsue, Japan: The Global WordNet Association.

Piasecki, M., Ramocki, R., & Maziarz, M. (2012b). Recognition of Polish derivational relations based on supervised learning scheme. In N. Calzolari, K. Choukri, T. Declerck, M. Uğur Doğan, B. Maegaard, J. Mariani, J. Odijk, & S. Piperidis (Eds.), *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)* (pp. 916–922). Istanbul, Turkey: European Language Resources Association (ELRA).

Piasecki, M., Ramocki, R., & Minda, P. (2012). Corpus-based semantic filtering in discovering derivational relations. In A. Ramsay & G. Agre (Eds.), *Artificial intelligence: Methodology, systems, and applications* (pp. 14–22). Berlin: Springer. (*Lecture Notes in Computer Science*, 7557).

Przepiórkowski, A., Bańko, M., Górski, R. L., & Lewandowska-Tomaszczyk, B. (Eds.). (2012). *Narodowy Korpus Języka Polskiego*. Warszawa: Wydawnictwo Naukowe PWN.

R Core Team. (2015). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing.

Radziszewski, A. (2013). A tiered CRF tagger for Polish. In *Intelligent tools for building a scientific information platform* (pp. 215–230). Berlin: Springer. (*Studies in Computational Intelligence*, 467).

Radziszewski, A., & Śniatowski, T. (2011). Maca — a configurable tool to integrate Polish morphological data. In *Proceedings of FreeRBMT11*. Software available at `http://nlp.pwr.wroc.pl/redmine/projects/libpltagger/wiki`

Radziszewski, A., Wardyński, A., & Śniatowski, T. (2011). WCCL: A morpho-syntactic feature toolkit. In I. Habernal & V. Matoušek (Eds.), *Text, speech and dialogue* (pp. 434–441). Berlin: Springer. (*Lecture Notes in Computer Science*, 6836).

Richardson, L., & Ruby, S. (2007). *RESTful web services* (1st ed.). Farnham: O'Reilly.

Rudnicka, E., Maziarz, M., Piasecki, M., & Szpakowicz, S. (2012). A strategy of mapping Polish WordNet onto Princeton WordNet. In *Proceedings of the 24th International Conference on Computational Linguistics COLING* (pp. 1039–1048).

Salton, G., & McGill, M. J. (1986). *Introduction to modern information retrieval*. New York, NY: McGraw-Hill, Inc.

Schaalje, G. B., Blades, N. J., & Funai, T. (2013). An open-set size-adjusted Bayesian classifier for authorship attribution. *Journal of the American Society for Information Science and Technology*, *64*(9), 1815–1825. `https://doi.org/10.1002/asi.22877`

Stamatatos, E. (2009). A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, *60*(3), 538–556. `https://doi.org/10.1002/asi.21001`

Stamou, C. (2008). Stylochronometry: Stylistic development, sequence of composition, and relative dating. *Literary and Linguistic Computing*, *23*(2), 181–199. `https://doi.org/10.1093/llc/fqm029`

Thies, G., & Vossen, G. (2008). Web-oriented architectures: On the impact of web2.0 on service-oriented architectures. In *Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference (APSCC)* (pp. 1075–1082). Yilan, Taiwan.

Woliński, M. (2006). Morfeusz — a practical tool for the morphological analysis of Polish. In M. A. Kłopotek, S. T. Wierzchoń, & K. Trojanowski (Eds.), *Intelligent information processing and web mining: Proceedings of the International IIS: IIPWM '06 Conference held in Ustroń, Poland, June 19-22, 2006* (pp. 511–520). Berlin: Springer. (*Advances in Soft Computing*).

Zhao, Y., Karypis, G., & Fayyad, U. (2005). Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery*, *10*(2), 141–168. https://doi.org/10.1007/s10618-005-0361-3

Acknowledgment

This work was co-financed:

- by the Polish Ministry of Science and Higher Education, a program in support of scientific units involved in the development of a European research infrastructure for the humanities and social sciences in the scope of the consortia CLARIN ERIC (www.clarin-pl.eu) and ESS-ERIC, 2015-2016.,
- and as a part of the investment in the CLARIN-PL research infrastructure (www.clarin-pl.eu) funded by the Polish Ministry of Science and Higher Education.

The authors declare that they have no competing interests.

The authors' contribution was as follows: concept of the study: ME, MP, TW; data analyses: ME, MP, TW; the writing: ME, MP, TW.

Publisher: Institute of Slavic Studies, Polish Academy of Sciences, University of Silesia & The Slavic Foundation